Generating Grammars for Natural Language Understanding from Knowledge about Actions and Objects

Alexander Perzylo[1] and Sascha Griffiths[2] and Reinhard Lafrenz[3] and Alois Knoll[3]

***Abstract*— Many applications in the fields of Service Robotics and Industrial Human-Robot Collaboration, require interaction with a human in a potentially unstructured environment. In many cases, a natural language interface can be helpful, but it requires powerful means of knowledge representation and processing, e.g., using ontologies and reasoning.**

In this paper we present a framework for the automatic generation of natural language grammars from ontological descriptions of robot tasks and interaction objects, and their use in a natural language interface. Robots can use it locally or even share this interface component through the RoboEarth framework in order to benefit from features such as referent grounding, ambiguity resolution, task identification, and task assignment.

I. Introduction

The following contribution discusses the design and implementation of a human-machine interface for a cloud robotics [1] system. The main advantage of relying on cloud computing in robotics is that it removes the robots' limitations with respect to *"onboard computation, memory or programming"* [2]. In the growing area of service robotics these limitations, and hence their removal, will be crucial as robots encounter unstructured, only partially observable and non-deterministic environments and are especially exposed to contact with naïve human users. Fig. 1 shows such a use-case staged in a hospital environment within the RoboEarth project.

Modern service robots thus depend on interaction with humans in a natural environment. Several modalities can be used for this, but in many cases, a natural language interface is a requirement. For instance, if the human needs to use his or her hands to manipulate objects in a physical interaction scenario, or in the case of service robots for elderly or disabled people, language might be the only way to communicate with the robot.

As Steels [3] notes, natural communication with robots is no longer an issue held back by the robot systems not having the right capabilities, as was the case a few decades ago, but a matter of developing natural language processing methods which are suitable for use in robotics. However, such developments are now hindered because *"many roboticists regard a speech-enabled interface as a somewhat independent, bolt-on goody"* [4].

Fig. 1. In the depicted scene a mock-up hospital use-case is shown. The Amigo robot has been instructed by a human patient to serve a drink.

Human-robot interaction via natural language requires a powerful knowledge representation mechanism and strong background knowledge. This includes the semantic description of objects, environments, possible actions, etc. An explicit representation of their functional and non-functional properties based on a formal specification can be used to automatically process the knowledge and to allow for state-of-the-art reasoning mechanisms.

In cloud-robotics systems, robots will share such knowledge. However, once their knowledge base changes, the interaction capabilities with respect to the robots' language need to also be amended. Therefore, it is desirable to have mechanisms by which the linguistic abilities of robots can be adjusted accordingly and also shared. We present a system in which we address this problem. Our solution is a system which fully automatically generates grammars based on the current state of the knowledge base. New task descriptions and all robots which share these semantic task descriptions can also make use of the natural language interface once a new grammar has been generated. The specific system is not designed to cover all aspects of language but is specific to the requirements of commanding robotic systems. However, it is flexible as the grammar updates as soon as new knowledge is acquired.

The remainder of the paper is organised as follows: After a description of related work, a system overview explaining the configuration and run-time phases is given. Then, the implementation and experiments justifying the validation of the concept are described. Finally, the results are summarised and an outlook is offered.

[1]Alexander Perzylo is with fortiss GmbH, Guerikestr. 25, 80805 München, Germany perzylo@fortiss.org

[2]Sascha Griffiths is with the Cognitive Science Research Group, School of Electronic Engineering and Computer Science, Queen Mary University of London, 10 Godward Square, London E1 4FZ, United Kingdom sascha.griffiths@qmul.ac.uk

[3]Reinhard Lafrenz and Alois Knoll are with the Institute for Informatics VI, Technische Universiät München, Boltzmannstr. 3, 85748 Garching, Germany lafrenz@in.tum.de | knoll@in.tum.de

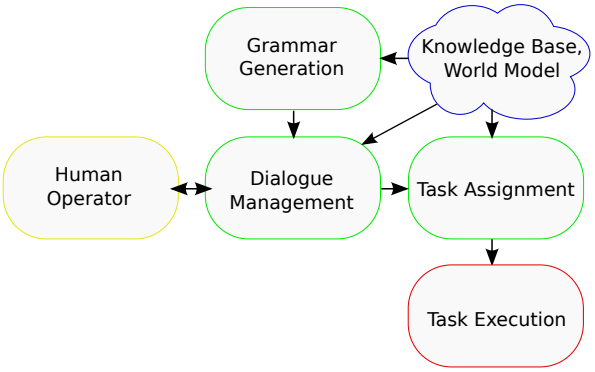

Fig. 2. Schematic overview showing how natural language commands given by a human operator are processed by the system, so that they can be mapped to known tasks. These identified tasks can then be assigned to and executed on available and compatible robots.

II. Related work

Ontologies are a common way to represent knowledge and can be shared easily with other suitably equipped systems, for instance as a component of cloud robotics applications, such as in the RoboEarth project [5].

The goal of the RoboEarth initiative [9] is the effective sharing of knowledge [10], data [11], and processing resources [12] among robots. This is often referred to as cloud robotics, and has established advantages regarding memory and processing limits. Additionally, models acquired by one robot can be re-used by another one.

While this kind of sharing of knowledge is an effective means of communication for robots, it is impossible to access a human's brain directly [13]. Therefore, an effective system requires interfaces which translate knowledge into communication channels which humans can access, thus enabling robots to map the information contained in human commands onto an appropriate representation.

Language is the primary means of information transmission both in human-human and human-machine interaction [14]. Thus, language is often mentioned by experts in surveys as one area of robotics research in which more work is required [15].

The system presented here uses the OpenCCG[1] library. This is based on the *combinatory categorical grammar* (CCG) formalism [16]. OpenCCG has already been successfully applied to robotics [17]. In particular, the CCG formalism has been shown to be useful for the use in task descriptions [18], [19].

III. System Overview

While a complete system with different physical robots has been implemented in RoboEarth, we will focus on its natural language interface in this paper, which can also be used independently from RoboEarth. This user interface consists of two main components. One component is the dialogue management component which handles speech input and output at run time, and the other component is the grammar generation component which generates grammars at configuration time.

Accordingly, two different phases can be distinguished. First, there is the *configuration phase*, in which a human expert annotates the system's action/object/environment ontologies [10] with links to concepts in the Wordnet ontology. Upon updating the natural language annotations, the grammar generation process is carried out once, resulting in an OpenCCG grammar that serves as an input for the dialogue component. Through the RoboEarth framework, all robots connected to the system can reuse this grammar.
We employ the web ontology language (OWL) [6] for encoding knowledge and annotations. OWL distinguishes between classes, individuals (instances of classes), and properties defined for classes or individuals. For persistently storing OWL-based descriptions we use a Sesame repository [7] which can evaluate SeRQL queries [8].

Next, during the *runtime phase* the generated grammar is used to map spoken (or written) commands directed at a robot back to concepts in the system's ontologies and then to interpret these correctly. This phase includes ambiguity resolution as well as possible additional interactions with the user preceding the execution of a task.
Fig. 2 shows a schematic overview of the system, with a human operator providing spoken commands to a robot. The dialogue component processes the input based on an automatically generated grammar. If the given command can directly be mapped to a task description available to the robot, the respective task identifier is sent to a task assignment component, which keeps track of robots and their working states. Otherwise, the command is considered being ambiguous or unknown, and further dialogue is initiated to find a valid mapping. The physical and software-related requirements of a task are modelled in an action recipe which includes the required capabilities of the robots [10]. The system can assign the identified task to a compatible robot, which then starts to execute the action.

IV. Configuration phase

A. Annotation of action and object ontologies

Natural languages, unlike formal languages, are full of ambiguities. A naïve mapping of an action the robot can perform to just one single verb would fail, if the user decided to use a different verb for the action. For instance, the same action can be described by the verbs *fetch* and *bring*. Ideally,

```
Declaration(Class(re:ServeBeverage))
Declaration(AnnotationProperty(re:linkedSynset))
AnnotationAssertion(re:linkedSynset re:
    ServeBeverage <http://www.w3.org/2006/03/wn/
    wn20/instances#synset-serve-verb-6>)
```

Fig. 3. Excerpt of annotated action ontology in OWL functional syntax. Task class *ServeBeverage* is linked with Wordnet's synonym set individual *synset-serve-verb-6*

[1]OpenCCG: The OpenNLP CCG Library – http://openccg.sourceforge.net/

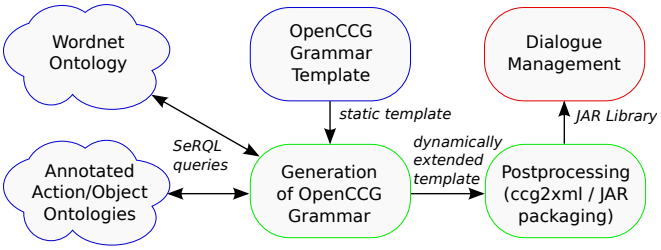

Fig. 4. Schematic overview of the grammar generation process during the configuration phase. The static OpenCCG grammar template is dynamically extended based on the annotations in the action and object ontologies and the corresponding Wordnet synonym sets. A post-processing step transforms the generated OpenCCG grammar from the DotCCG format to its XML based representation and packages the created files into a JAR library.

one would link to all synonyms as an appropriate identifier for an action or object.

To generate lexical entries from the knowledge base, a mapping from the classes and individuals in the knowledge base to words in natural language must be provided. This mapping can be created in two ways: either by adding explicit information about the mapping to the ontology, or by giving a fixed set of rules, for instance by using the class names as lexical forms in the grammar.

The second approach, however, has several disadvantages. First of all, it is assumed that the ontology creator adhered to a set of implicit rules, which can lead to hard-to-spot errors during grammar generation if the creator violated one of the rules. Furthermore, it can be hard to decide on the word's syntactic category using the name of an entity in the ontology alone. On the other hand, if the ontology requires explicit annotations for the mapping, entities in the ontology lacking these annotations can be automatically skipped without breaking the system.

For these reasons the classes in the ontology are annotated explicitly with Wordnet synonym sets (synset), which capture the classes' meaning. Fig. 3 lists an excerpt of the action ontology, which is comprised of the declaration of OWL class *ServeBeverage*, the annotation property *linkedSynset* and an annotation assertion which links the *ServeBeverage* class with Wordnet's synset *synset-serve-verb-6*. While this approach causes additional work for the ontology maintainer to annotate all classes correctly, it suffers from none of the shortcomings described above.

Consequently, classes that describe actions are annotated with verb synsets; classes that describe physical objects are annotated with noun synsets; and classes that only serve as fillers for other classes' properties, e.g., colour or temperature, are annotated with adjective synsets.

B. Automatic generation of OpenCCG grammar

The actual grammar used by the dialogue component is created using content from the knowledge base and a static grammar template.

The grammar template is encoded in the DotCCG format. This allows for the prior definition of the grammar's static part beforehand in a template file, in the form of functions,

```
SELECT Concept, Synset FROM
{Concept} re:linkedSynset {Synset}
USING NAMESPACE
re = <http://www.roboearth.org/kb/roboearth.owl#>
```

Fig. 5. SeRQL query for extracting all natural language annotations from the action and object ontologies

```
<?xml version='1.0' encoding='UTF-8'?>
<sparql xmlns='http://www.w3.org/2005/sparql-
    results#'>
 <head>
  <variable name='Concept'/>
  <variable name='Synset'/>
 </head>
 <results>
  <result>
   <binding name='Concept'>
    <uri>http://www.roboearth.org/kb/roboearth.owl#
        Cold</uri>
   </binding>
   <binding name='Synset'>
    <uri>http://www.w3.org/2006/03/wn/wn20/
        instances/synset-cold-adjective-1</uri>
   </binding>
  </result>
  ...
 </results>
</sparql>
```

Fig. 6. List of found pairs of Ontology concepts and their natural language annotation

macros and category definitions. These functions and macros can then be used during the generation of the dynamic part of the grammar. For example, when a verb is defined, there has to be a definition for the verb in the singular, singular third person and plural form. Since the building of the verb's singular third person and plural form follows a strict rule, this can be automated by creating a function. Also, definitions of frequently used words and functional words that are not explicitly specified in the knowledge base, like "a", "the", "and" or personal pronouns like "you", "me" or "ours", are stored in the template. Additionally, the template holds general information about which features are available for words of different classes, like case, number, etc. The complete template can be found online[2].

In a first step, the annotations described in Section IV-A are searched for in the knowledge base. Every annotation points to a Wordnet synset ID. Thus, the Wordnet database is queried for the lexical form of all words which belong to the specified synsets. We distinguish between *nouns*, *verbs* and *adjectives* for all words, so that the correct representation for each of the lexical items can be generated. Fig. 5 lists the query in the SeRQL query language. The evaluation of the query against the knowledge base results in a set of ontology concepts and their Wordnet synset annotations, as can be seen in Fig. 6.

The lexical categories for nouns are annotated with the Uniform Resource Identifiers (URI) of their corresponding

[2]http://www6.in.tum.de/~perzylo/template.ccg

```
SELECT DISTINCT Individual FROM
{Individual} rdf:type {re:bottle}; kr:describedInMap {Map},
{Map} rdf:type {kr:SemanticEnvironmentMap}, {Room} kr:describedInMap {Map},
{U} kr:properPhysicalParts {V} kr:properPhysicalParts {W} kr:properPhysicalParts
{X} kr:properPhysicalParts {Room},
{U} rdfs:label {City}, {V} rdfs:label {Street},
{W} kr:streetNumber {StreetNo}, {X} kr:floorNumber {FloorNo}, {Room} kr:roomNumber {RoomNo}
WHERE
City="Munich"@en AND Street="Guerickestrasse"@en AND StreetNo="25" AND FloorNo="1" AND RoomNo="109"
USING NAMESPACE
kr=<http://ias.cs.tum.edu/kb/knowrob.owl#>, re=<http://www.roboearth.org/kb/roboearth.owl#>
```

Fig. 7. SeRQL query for searching for all instances of the bottle concept in a specified environment. An environment is specified in a hierarchical manner using room and floor numbers, street and city names.

classes in the ontology. This is possible, since every class in the ontology that describes physical objects, can be mapped to a certain meaning of a noun. Consequently, there is a one-to-one mapping between classes and synsets. The same is applicable for adjectives.

For verbs, however, there is no such one-to-one mapping, because there may be many different tasks which are associated with the same verb sense. For example, the verb *bring to* can refer to a task which describes how to serve a drink, as well as to a task dealing with a spatial displacement in any other context. This issue is tackled during runtime.

The extracted information is used to extend the static grammar template. As a necessary post-processing step, OpenCCG's tool *ccg2xml* is invoked to generate an XML representation that can be used by the OpenCCG parser. As a final step, the generated XML files are packed into a JAR file which serves as an input to the dialogue component. The whole process is visualised in Fig. 4.

V. Runtime Phase

A. Language parsing

A natural language interface requires that the robot can understand the commands given to it by the user. For this, the natural language sentences are parsed into a logical form, which represents the meaning of a sentence before any further processing takes place. A natural language parser has to account for the fact that parts of the sentence may not appear in their canonical position in the sentence. Moreover, sentence elements that build a logical unit can be scattered across the sentence.

The generated logical form is used to analyse the sentence's structure and how the different parts are semantically related to each other, e.g., which noun is the subject of which verb. This information is important for grounding it in the knowledge base's concepts in order to interpret the meaning of the sentence.

B. Grounding the referents

Once a command is parsed into its logical form, the system must identify a corresponding task to be executed by a robot. In order to accomplish this mapping, the sentence's referents must be grounded in the system's knowledge base.

This can be done by exploiting the sentence structure and the annotations in the system's ontologies. Every verb phrase in the sentence is treated as a command to the robot, and thus corresponds to a task it shall perform. Each of these phrases has either one, two or three attached noun or prepositional phrases, depending on whether the verb is intransitive, transitive or ditransitive. These phrases are the arguments to the verb. They specify the objects on which the task must be carried out. Therefore, every argument must be grounded to an individual in the knowledge base.

Every noun is mapped to a certain class in the knowledge base. Thus all individuals in the knowledge base that are instances of this class or instances of one of its subclasses are possible referents of this noun phrase. An exemplary SeRQL query used to find all bottles in a particular environment is shown in Fig. 7. There may be further restrictions, on which individuals are possible referents for the noun. An adjective, that modifies a noun phrase, describes additional requirements the referents must fulfil. Like nouns, every adjective is mapped to exactly one class in the knowledge base. If an adjective is attached to a noun, then all possible referents must be not only instances of the noun's class or subclasses, but they must also be linked to another individual that is an instance of the adjective's class.

For example, the noun "bottle" in the noun phrase "the red bottle" is grounded to individuals, which are instances of the class `Bottle` and are linked to an individual, which is an instance of the class `Red`.
Prepositional phrases attached to a noun are treated in a similar manner, the only difference is, that the constituent of the prepositional phrase might in turn have attached adjectives or prepositional phrases. In such cases, the candidate individual, to which the noun phrase's referent could be mapped, must be restricted by certain further individuals itself.

For verbs there is no such one-to-one mapping to classes in

```
SELECT ActionRecipe FROM
{ActionRecipe} re:linkedSynset {C}
WHERE
C in (wn:synset-bring_to-verb-1)
USING NAMESPACE
re = <http://www.roboearth.org/kb/roboearth.owl#>
wn = <http://www.w3.org/2006/03/wn/wn20/instances#>
```

Fig. 8. SeRQL query for searching for all action recipes that are associated with the given verb's synonym sets

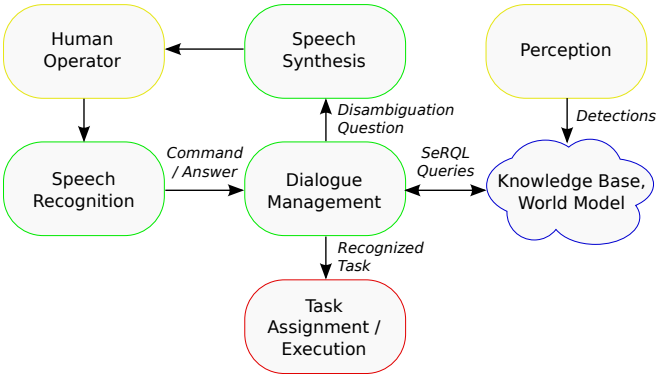

Fig. 9. Schematic overview of the run time phase. The human operator provides a spoken command to the system. The dialogue manager utilises the speech recognition component to receive the command in a textual representation. Using the OpenCCG grammar generated during the configuration phase and queries to the knowledge base the dialogue manager interprets the command. Perception modules might constantly update the world state. Unresolvable ambiguity is handled through posing questions to the operator, which are rendered by the speech synthesizer. Finally, the recognised task is sent to the task assignment component.

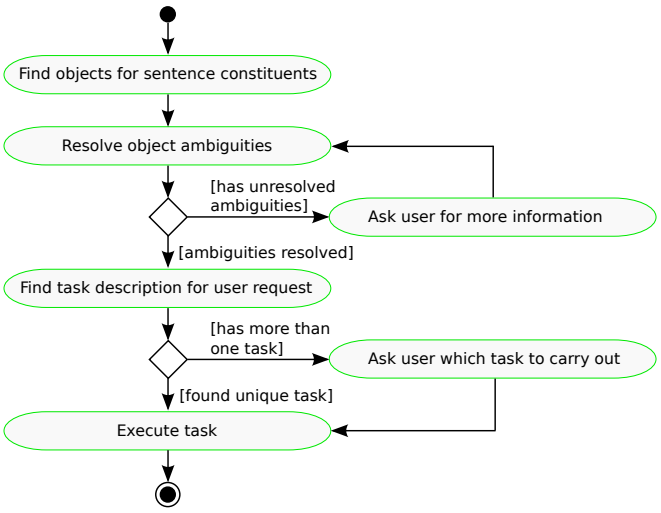

Fig. 11. Activity diagram of the ambiguity resolution process for mapping a human operator's command to a known task description.

the knowledge base. Instead, for every verb in the sentence, the Wordnet ontology is searched for all synsets that contain a word sense associated with this verb. Then the knowledge base is searched for task descriptions, which are annotated with at least one of the previously found synsets. Fig. 8 shows a SeRQL query used to extract all task descriptions associated with the *bring-to* verb synset. Disambiguation of task descriptions uses the grounded noun phrases and is explained in more detail in Section V-C.

C. Ambiguity resolution

For every noun and verb phrase in a parsed sentence, a matching entity from the knowledge base is searched for. But there might be more than one individual acting as a candidate for a possible match. A robust natural language interface has to be capable of dealing with these kinds of ambiguities and resolving them correctly. However, they cannot be dealt with on a purely syntactical level; instead the robot's knowledge about the domain must be taken into account.

Nouns specify the class of the matching individual, and the adjectives and prepositional phrases attached to noun phrases describe further restrictions the individuals must fulfill. These requirements are translated into a query to the knowledge base. But it is not guaranteed that this query will yield

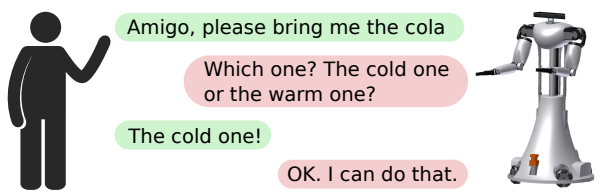

Fig. 10. Example dialogue between human operator and the Amigo robot. Given that there are two instances of a cola bottle in the robot's world model – a warm one on the table and a cold one in the refrigerator – the robot uses this information to resolve the ambiguity for the requested object.

only one individual, since there may be several matches complying with the given description.

For every candidate object class in the ontology which could be a referent of the noun, the knowledge base is searched for task descriptions, that are annotated with a Wordnet synset including the sentence's verb operating on objects of that class. Entities that cannot be used in such a way are discarded. If there is still more than one candidate after this elimination process, the user is asked for more information about the object to which they are referring. In order to speed up this process, the most discriminative feature is computed and used to pose a question to the user, as visualised in Fig. 10.

Similarly, a verb is likely to refer to several task descriptions. This is due to the fact that verbs can have several distinct meanings depending on the context in which they are used. Thus it is necessary to incorporate the context into the process of resolving ambiguities.

Once all nouns in a command are unambiguously grounded to individuals from the knowledge base, the correct task must be found. As described above, the context in which a verb is used determines which semantic meaning the speaker intended to express. Hence, it is necessary to generate a grounding of the context, i.e. the arguments of the verb, first, before an attempt to ground the verb itself can be made.

For every task description in question, the knowledge base is queried about the types of objects it operates on. This yields a list of classes for every task description. These lists can be matched with the individuals' classes, that were chosen as the noun phrases' referents. Task descriptions which differ in arity or act on objects of irrelevant classes are discarded. If this information is not enough to find a distinct task description, the user is asked explicitly which task should be executed. An activity diagram of the complete ambiguity resolution process is shown in Figure 11. The whole run time phase is summarised in Fig. 9.

VI. Implementation

The software components described in this paper have been implemented as ROS modules and integrated with the RoboEarth framework. They have been successfully showcased at the final demonstration of the RoboEarth project in Eindhoven, Netherlands, in January 2014. The natural language interface was used in a mock-up hospital environment, as seen on Fig. 1, to command various robots to assist human patients.

A video introducing the vision of RoboEarth and showing footage of the final demonstration can be found online.[3]

VII. Conclusions & Outlook

The natural language interface presented in this paper shows a simple but flexible approach to dialogue handling for cognition-enabled robots. The assignment of tasks to robots and the local response by a given robot are leveraged in our approach by moving resources intensive elements of language processing and those elements which are useful for other robots in the system into the cloud. This is a deliberate choice, efficiently allowing a central system to react as well as enabling short response times for a given robot in an application context.

Theoretical approaches to dialogue handling emphasise the role of embodiment in cognitive tasks for robots [20]. Despite being simplistic, the central component handles dialogues in a dis-embodied fashion. However, the task execution is, in a sense, grounded in the knowledge the system has about the robots and their perceived environment.

The dialogue system itself is designed to be flexible, but in order to increase its effectiveness, a better mapping between speech recognition and grammar generation needs to be realised. One such approach, especially in the field of robotics, is based on real-time spoken phrase recognition [21]. A simpler solution would be a transformation between the OpenCCG framework and the speech recognition grammar. Currently, the missing interface between the open-ended grammar generation and the finite speech recognition grammar is the main limitation of our system.

Acknowledgements

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 248942 in the project RoboEarth, and grant agreement no. 287787 in the project SMErobotics, and FET grant agreement no. 611733 in the project ConCreTe.

The authors would like to thank Stephen McGregor for comments on a draft of this paper.

References

[1] B. Kehoe, S. Patil, P. Abbeel, and K. Goldberg, "A survey of research on cloud robotics and automation," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 2, pp. 398–409, 2015.

[2] K. Goldberg and B. Kehoe, "Cloud robotics and automation: A survey of related work," EECS Department, University of California, Berkeley, Berkeley, California, Tech. Rep. UCB/EECS-2013-5, 2013.

[3] L. Steels, "Grounding language through evolutionary language games," in *Language Grounding in Robots*. Berlin: Springer Verlag, 2012, pp. 1–22.

[4] R. K. Moore, "From talking and listening robots to intelligent communicative machines," in *Robots that Talk and Listen – Technology and Social Impact*. Boston, MA: De Gruyter, 2014, pp. 317 – 336.

[5] O. Zweigle, R. van de Molengraft, R. d'Andrea, and K. Häussermann, "Roboearth: connecting robots worldwide," in *Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human*. ACM, 2009, pp. 184–191.

[6] P. Hitzler, M. Krötzsch, B. Parsia, P. F. Patel-Schneider, and S. Rudolph, Eds., *OWL 2 Web Ontology Language: Primer*. W3C Recommendation, 27 October 2009, available at http://www.w3.org/TR/owl2-primer/.

[7] J. Broekstra, A. Kampman, and F. Van Harmelen, "SeSame: A generic architecture for storing and querying RDF and RDF schema," in *The Semantic WebISWC 2002*. Springer, 2002, pp. 54–68.

[8] J. Broekstra and A. Kampman, "SeRQL: An RDF query and transformation language," in *3rd International Semantic Web Conference, ISWC*, Hiroshima, Japan, 2004, pp. 23–39.

[9] M. Waibel, M. Beetz, J. Civera, R. D'Andrea, J. Elfring, D. Gálvez-López, K. Häussermann, R. Janssen, J. Montiel, A. Perzylo, B. Schieß le, M. Tenorth, O. Zweigle, and R. van de Molengraft, "RoboEarth," *IEEE Robotics & Automation Magazine*, vol. 18, no. 2, pp. 69–82, Jun. 2011.

[10] M. Tenorth, A. C. Perzylo, R. Lafrenz, and M. Beetz, "Representation and Exchange of Knowledge About Actions, Objects, and Environments in the RoboEarth Framework," *IEEE Transactions on Automation Science and Engineering*, vol. 10, no. 3, pp. 643–651, Jul. 2013.

[11] J. Elfring, S. van den Dries, M. J. G. van de Molengraft, and M. Steinbuch, "Semantic world modeling using probabilistic multiple hypothesis anchoring," *Robotics and Autonomous Systems*, vol. 61, no. 2, pp. 95–105, Dec. 2012.

[12] D. Hunziker, M. Gajamohan, M. Waibel, and R. D'Andrea, "Rapyuta: The RoboEarth Cloud Engine," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, May 2013, pp. 438–444.

[13] L. S. Lopes and T. Belpaeme, "Beyond the individual: new insights on language, cognition and robots," *Connection Science*, vol. 20, no. 4, pp. 231–237, Dec. 2008.

[14] J. P. de Ruiter, "On the primacy of language in multimodal communication," in *Workshop Proceedings on Multimodal Corpora: Models of Human Behaviour for the Specification and Evaluation of Multimodal Input and Output Interfaces.(LREC2004)*. Paris: ELRA - European Language Resources Association, 2004, pp. 38–41.

[15] S. Griffiths, C. Natale, R. Araújo, G. Veiga, P. Chiacchio, F. Röhrbein, S. Chiaverini, and R. Lafrenz, "The ECHORD Project: A General Perspective," in *Gearing up and accelerating cross-fertilization between academic and industrial robotics research in Europe:*, ser. Springer Tracts in Advanced Robotics, F. Röhrbein, G. Veiga, and C. Natale, Eds. Heidelberg: Springer International Publishing, 2014, vol. 94, pp. 1–24.

[16] M. Steedman, *The Syntactic Process*. Cambridge, MA / London: MIT Press, 2001.

[17] M. Rickert, M. E. Foster, M. Giuliani, T. By, G. Panin, and A. Knoll, "Integrating language, vision and action for human robot dialog systems," in *Proceedings of the International Conference on Universal Access in Human-Computer Interaction, HCI International*, ser. Lecture Notes in Computer Science, vol. 4555. Beijing, China: Springer, 2007, pp. 987–995.

[18] J. Dzifcak, M. Scheutz, C. Baral, and P. Schermerhorn, "What to do and how to do it: Translating natural language directives into temporal and dynamic logic representation for goal management and action execution," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, May 2009, pp. 4163–4168.

[19] M. Scheutz, R. Cantrell, and P. Schermerhorn, "Toward humanlike task-based dialogue processing for human robot interaction," *AI Magazine*, vol. 32, no. 4, pp. 77–84, 2011.

[20] R. Pfeifer and J. Bongard, *How the body shapes the way we think: a new view of intelligence*. Cambridge, MA: MIT press, 2007.

[21] R. Veale and M. Scheutz, "Neural circuits for any-time phrase recognition," in *Proceedings of the 34th Annual Conference of the Cognitive Science Society*, N. Miyake, D. Peebles, and R. P. Cooper, Eds. Austin, TX: Cognitive Science Society, 2012, pp. 1072–1077.

[3] https://www.youtube.com/watch?v=mgPQevfTWP8