Learning Actions from Human-Robot Dialogues

Rehj Cantrell* and Paul Schermerhorn* and Matthias Scheutz**
Human-Robot Interaction Laboratory
*Indiana University and **Tufts University, USA
{rcantrel,pscherme}@indiana.edu, mscheutz@cs.tufts.edu

***Abstract*— Natural language interactions between humans and robots are currently limited by many factors, most notably by the robot's concept representations and action repertoires. We propose a novel algorithm for learning meanings of action verbs through dialogue-based natural language descriptions. This functionality is deeply integrated in the robot's natural language subsystem and allows it to perform the actions associated with the learned verb meanings right away without any additional help or learning trials. We demonstrate the effectiveness of the algorithm in a scenario where a human explains to a robot the meaning of an action verb unknown to the robot and the robot is subsequently able to carry out the instructions involving this verb.**

I. Introduction

In human-robot interaction contexts, it is important for robots to allow humans to tell them what to do. However, current robots are limited in their capability to interact with humans in spoken natural language, largely due to two main types of limitations: those stemming from natural language understanding systems and those stemming from cognitive robotic architectures. As a result, today's robots understand a limited number of concepts, and can thus carry out a limited number of commands that use these concepts. For example, a robot that does not understand the concept of "accompanying a person to the door" in a reception setting will not be able to perform the action. And even if it understands the concept, this does not automatically imply that it will know how to do it (e.g., because it might not know how to drive side-by-side with the person or how to lead the person through a crowded room).

While research in speech recognition and computational linguistics has produced impressive algorithms for processing syntactic aspects of natural language (e.g., [1]), progress in processing semantic and pragmatic aspects of natural language—prerequisites for understanding and carrying out spoken commands—is much more restricted and modest. Part of the problem is the way in which meaning representations are typically constructed (e.g., being based on only a limited number of semantically annotated lexical items). Another, possibly even more important, restriction is the limited number of concepts that are explicitly represented in robotic cognitive architectures. The combination of both limitations severely constrains what current robots can understand and do.

Additional complications arise in human-robot interaction domains, where the goal is for robots to interact with humans in natural ways [2]. In such contexts, it is impossible to know all the concepts the robot might need ahead of time, because it might not be clear or known how humans will phrase requests or which requests they will formulate. Hence, manually adding new semantic representations and concepts at design time alone is insufficient for run-time performance. Rather, robots will need to acquire new natural language constructions at run-time and semantically ground them within their knowledge base. Moreover, because robots will likely encounter novel situations not anticipated by the designer, robot behaviors cannot exclusively be pre-programmed. Robots must be able to learn and produce new behaviors at runtime. Only by meeting both requirements will robots be capable of being tasked in natural ways.

To our knowledge, there is currently no robotic architecture that allows operators to talk with robots directly and interactively tell them what to do and how to do it—that can both (1) learn new meanings of words based on natural language explanations and (2) apply them directly in different contexts without any additional learning, programming, adaptation, or other help from the instructor. Specifically, there is no robotic system that has demonstrated the capability of learning new action verb meanings based on natural language descriptions and then immediately being able—without additional training—to perform the described action.

In this paper, we describe such a system and demonstrate its functionality in real-time human-robot dialogue interactions where a human teaches a robot the meanings of action verbs. We start by further elaborating our motivation and discussing the state-of-the-art in robotic systems that can be taught in natural language. Then, in Section III, we describe our proposed learning algorithm for how meaning representations are formed as part of an integrated incremental syntactic-semantic analysis. We then demonstrate an episode in which a human interactively explains an action to a robot, allowing it to carry out a command that it previously did not know how to do. Though space does not allow a full evaluation alongside the necessary description of this system, this provides background for a future evaluation using human-subject interaction studies. Finally, we summarize our contribution and discuss some of its limitations.

II. Motivation and Background

Humans communicate most naturally and effectively using spoken language. Hence, *natural human-robot interaction* will require robots to understand and respond appropriately

to *natural* spoken language—any solution that requires humans to master a grammatically-limited control language for robot instruction will ultimately take away naturalness and likely limit the effectiveness of the interaction. While there are implementations of robotic systems that can understand sequences of natural language instructions (e.g., [3]), they are typically not able to learn new words and thus new actions based on the meanings of those words. And while there are robotic systems that can learn new actions, they usually do not do so from natural language instructions (e.g., [4]).

In general, current robotic architectures are implemented with very limited semantic knowledge, as programming enough semantic knowledge for robots to respond acceptably well to natural language instructions is prohibitive. Additionally, robots are equipped to handle only a very limited number of domains, as changing contexts might require a robot to handle very different vocabulary and concepts. To overcome these limitations, it is essential that robotic NLU algorithms be self-adapting, i.e., be able to both learn the syntactic and semantic structures of new words, and to ground those words semantically to real-world actions, objects, and concepts.

Several projects have attempted to address this need for online learning of meanings in the context of human-robot interaction.[1] [6], for example, demonstrate an instruction system for motion learning in robots using a structured control language. However, this work relies on an very limited, hand-crafted fragment of English (containing approximately 10 grammar rules, compared to the hundreds of rules typically necessary to even approximate a natural human-like grammar).

[7], [8] demonstrate a system for word learning in robots; however they only learn nouns, pronouns, and prepositions (limited to perceptions), not verbs and actions.

[9] describe a system that learns to perform action sequences from natural language instructions; however, the sequences are situation-specific, rather than being adaptable to many specific situations.

[10] describe a system that can learn new actions based on natural language descriptions. However, the descriptions have to follow a very rigid structure that maps easily onto a procedural semantics for action execution.

Similarly, [11] translate natural language instructions into sequences of goals for which a planner then generates corresponding action sequences. They employ a flexible dependency parser trained on real-world language, rather than relying on a small set of grammar rules. As a result, their system is able to handle several types of disfluencies robustly, which none of the above systems can handle. However, while they are also able to fill in missing steps in instructions, their system cannot handle unknown words.

What is needed is a systematic way of handling new words semantically on the natural language side and procedurally in the case of action verbs on the action execution side. In the next section, we will specifically propose an algorithm that can learn the procedural semantics of verbs together with control laws that run in real-time on a real robot to perform the action. Moreover, scalability is implicit in the design, as every component is general: the grammar is trainable, not hand-written; definitions for unknown words of any type are learned automatically; and learning is not a separate process, but uses syntactic/semantic parsing procedures already instantiated for understanding.

[1]Note that we are not focussing on work here that attempts to learn action verb meanings from observations (e.g., [5]). Moreover, for space reasons, we cannot review past work on general word learning that is not specific to action verb learning using natural language instructions on robots.

III. Learning Meanings of Action Verbs

Learning the meanings of novel action verbs via natural language interactions requires a robot to identify action expressions, to extract their *syntactic requirements* and to generate *procedural semantics* that can be associated with the syntactic form. In particular, it is both necessary to understand the syntactic constraints in natural language, and possible to exploit those constraints for the generation of semantic forms.

Take, for example, the following uses of the word **wait**: (1) "I want you to wait," (2) "to wait means to stay here", (3) "wait", and (4) "Paul waits". While all uses of **wait** must ultimately reveal a grammatical subject denoting an actor who waits, only in some of the utterances is the subject actually expressed (namely examples 1 and 4). Furthermore, in examples 1 and 2, the word **wait** takes a syntactic argument, **to**, that the others do not require.

In order to represent these different ways in which action verbs might occur, the algorithm for determining meaning representations must have two properties: (1) it must be able to insert grammatical subjects that are not overtly expressed; and (2) certain words that serve only syntactic roles (without any particular meaningful semantic interpretation) must not be considered arguments to the action verb. We now introduce a process for the generation of semantic forms that satisfies both requirements.

A. Representing semantic roles

The syntactic analysis is performed using an incremental dependency parsing algorithm (based on the shift-reduce parsing algorithm [12]) that creates a dependency graph by successively choosing one of four parsing actions based on the current input word. The first two actions, *shift* and *reduce*, are the same as in the shift-reduce algorithm: to *reduce a node* means to decide that processing of the node has completed and the node is subsequently set aside; to *shift* (to a new node) means to start working on the next input node. The other two actions are *left-arc*—connect a (syntactic) "head" node (or parent node) to a child node on its left—and *right-arc*—connect a head node to a child node on its right. The parser chooses among the four parsing actions based on a model learned from a pre-parsed corpus via the k-nearest-neighbor machine learning algorithm. The edges in the dependency graph are labeled according to the type of relationship that exists between the two words represented by the connected nodes where the set of possible arc labels

is based on the set used in the most common dependency grammar Penn Treebank ([13]). For example, if a subject (SBJ) arc exists between two words, it indicates that the child is the subject of the head.

The creation of a semantic arc by the dependency parser indicates a predicate-argument relationship. For example, if a SBJ arc is created between two nodes, it indicates that the head node is a predicate with the child being its subject argument, which can, in turn, be expressed in λ-theoretic semantics. The dependency structure for the node **want** with three arguments for the subject (SBJ), the object (OBJ), and the action(ACT), would be expressed as:

$$\lambda x_{SBJ}.\lambda y_{OBJ}.\lambda z_{ACT}.want(x_{SBJ}, y_{OBJ}, z_{ACT})$$

indicating that the *subject* **want**s the *object* to perform a particular *action*. Note that the variables bound by the $lambda$ operators are typed, thus requiring arguments of a particular type for subsequent $lambda$-conversions to succeed. The idea is to explicitly represent semantic restrictions on the types of entities to which a given action applies (e.g., usually autonomous agents or *actors* such as people and robots are taken as being able to perform actions such as wanting, while certain entity types might be legitimate arguments for the objects an action like following operates on). Alternatively, we can also use untyped variables, but make the type restrictions explicit as part of the meaning representation:

$$\lambda x.\lambda y.\lambda z.want(x, y, z)\&actor(x)\&actor(y)\&action(z)$$

Type restrictions for overt arguments are taken directly from the corresponding argument in the dependency node, and restrictions for created arguments (e.g., an implicit subject) are deduced from the requirements that led to their creation.

Note that either of the above renditions of **want** in $lambda$ form only explicates the argument types of wanting, but does not capture what it means for someone to want something as in "I want to sleep" or for someone to want somebody else to do something as in "I want you to follow me."

The latter case might express the human speaker's request to the robot to follow the speaker. Hence, the meaning of "want" needs to be represented as a request or command that the robot perform the "follow-speaker" action. This could be expressed as $\lambda y_{OBJ}.\lambda z_{ACT}.received\_orders(y_{OBJ}, z_{ACT})$.[2] The order, in turn, can be expressed (and should be understood by the robot) as a goal that the robot needs to achieve. To indicate that the semantic forms of action verbs that denote actions to be performed by the robot are really goal state descriptions, we will use the past-tense form of the action verb expressing the requested action. So, for example, the command to pick up a block would be expressed as $picked\text{-}up(robot, block)$, meaning that the robot should arrive at a state where it, the actor, has completed the action of picking up the block.

[2]Changing the predicate in "x wants y to do z" to "y receives-order z" is, of course, quite specific to this particular example of the three-place wanting-action that can be construed as an "order".

```
function determineVerbMeaning()
  DG = action = args = meaning = ∅
  while ¬(end-of-utterance ∨ action=REDUCE) do
    word := readNextInput()
    action := getParserAction(word,DG)
    DG := updateDependencyGraph(word,DG,action)
    if ¬isVerb(word) then
      args := args ∪{word}
      meaning := update-λ(meaning,args)
    else
      match := retrieveMatches(word,args)
      if ¬empty(match) then
        meaning := fill(match,DG)
      else
        meaning := makeLambda(word,args,DG)
      end if
    end if
  end while
```

Fig. 1. The algorithm for translating a syntactic dependency structure into a corresponding λ-expression.

Another example, which this time includes a *syntactic* arc label, is that of **follow** in the utterance "to follow somebody". It also has three dependents: a SBJ, a TO, and an OBJ. However, since TO is purely syntactic (in other words, it comes into the sentence from a rule-based syntax and does not add anything semantically), it is not considered an argument. For that reason, the semantic form of "follow" would be expressed as $\lambda x.\lambda y.follow(x, y)\&actor(x)\&actor(y)$, indicating that the actor x follows the actor y.

We next show how to generate meaning representations for new semantic forms of action verbs from semantic forms representing known meanings.

B. Generating semantic forms and action representations

The basic idea of the proposed algorithm for generating meaning representations for new semantic forms is simple: we start by parsing a sentence word-by-word and attempt to retrieve an appropriate $lambda$-expression from a dictionary of known expressions. In order for an expression to be appropriate for use, its arguments must "match" those of the dependency node for the word under consideration, i.e., it must have the same number and type of semantic arguments as the dependency node. When a matching expression is found, we retrieve it and fill in arguments wherever possible. When no matching expression can be found, a new one is created as described above based on the unknown word token and its argument structure as presented. As words continue to arrive, the argument structure may need to be updated. In that case, previously-created expressions that no longer fit are discarded and new meaning expressions are either created or retrieved. This process continues until the dependency parser performs a *reduce* action or reaches the end of the utterance. We now know that we have found all arguments for a verb. A pseudo-code summary of this algorithm is shown in Figure 1.

If a procedural representation of the verb's action meaning exists in the knowledge base, we simply retrieve it and associate it with the meaning representation. Otherwise we examine the argument structure to determine whether the action is required of the robot, in which case the robot will

have to find a way to obtain it (e.g., by asking a human to define the meaning of the action verb or possibly by using a problem solver to determine what action sequences might lead to the desired goal state expressed by the past-tense form of the action verb—note that for novel verbs the problem solver will most likely not recognize the goal state representation and will thus not be able to remedy the situation, leaving only human help as an option). We now walk through an example to illustrate the various steps.

Consider the utterance **"To follow someone means to stay within one meter of them,"** which defines the verb **to follow**. Figure 2, referenced throughout this discussion, shows the parse graph at various stages. Upon recognition of the first three words, **To follow someone**, the parser produces the graph shown in Figure 2(a). We have no expression for **follow**, but from the parse we know that the predicate follow has one child connected by a syntactic arc labeled as a VMOD ("verb modifier") and one child connected by an arc labeled OBJ ("object"). The λ-expression is created according to the algorithm in Figure 1. In this case, we get

$$\lambda x_{SBJ}.y_{OBJ}.followed(x, y)$$

giving us for the full utterance:

$$\lambda x_{SBJ}.followed(x, someone)$$

"followed" appears to be a two-place predicate taking a subject and an object. The robot does not have a definition for this word, so it creates a new one ("CL" means "current lambda expression"):

```
Created meaning expression for follow:
#SBJ.#OBJ.followed(SBJ,OBJ)
CL = #SBJ.followed(SBJ,someone)
```

Note that we are not yet resolving the reference of "someone", nor are we introducing a variable instead, for reasons that will become clear shortly.

The graph structure after recognizing **means** is shown in Figure 2(b). The word **means** takes **follow** as its argument with the arc label **SBJ**, so **means** appears to be a one-place predicate. The node structure of **follow** is unchanged. The only expression we have for **means** takes two arguments, SBJ and OBJ, so a new expression with an overt subject is created:

$$\lambda x_{SBJ}.means(x)$$

and the current λ-representation, including two unknown words (**follow** and the hypothesized single-argument form of **means**) is:

$$means(\lambda x_{SBJ}.followed(x, someone)).$$

The computer produces the output:

```
Created meaning expression for means:
#SBJ.meant(SBJ)
CL = meant(#SBJ.followed(SBJ,someone))
```

After **to stay** is recognized (Figure 2(c)), the parser finds a matching expression for **means**, $\lambda x_{SBJ}.\lambda y_{OBJ}.assoc\_meaning(x, y)$, and has the following λ representation:

$$assoc\_meaning(\lambda x_{SBJ}.followed(x, someone), \lambda x_{SBJ}.stayed(x)).$$

producing the following output ("AM" means "associated meaning"):

```
Created meaning expression for stay:
#SBJ.stayed(SBJ)
CL = AM(#SBJ.followed(SBJ,someone),
  #SBJ.stayed(SBJ))
```

As depicted in Figure 2(d), once **within** is received, **stay** has two arguments, syntactic VMOD and LOC, matching the dictionary definition:

$$\lambda x_{SBJ}.\lambda y_{LOC}.maintained(x, y)$$

giving us the λ-representation:

$$assoc\_meaning(\lambda x_{SBJ}.followed(x, someone), \lambda x_{SBJ}.maintained(x, within))$$

A new expression must be created for **within**:

```
Created meaning expression for within:
within
CL = AM(#SBJ.followed(SBJ,someone,
     #SBJ.maintained(SBJ,within))
```

The addition of **one meter**, which attaches to **within**, results in the λ-expression:

$$assoc\_meaning(\lambda x_{SBJ}.followed(x, someone), \lambda x_{SBJ}.maintained(x, <(one\_meter))$$

This expression is replaced with the arrival of **one**:

```
Created meaning expression for:
#NUM.within(NUM)
CL = AM(#SBJ.followed(SBJ,someone,
     #SBJ.maintained(SBJ,within(1))
```

The dictionary expression for **within** is matched when *meter* is recognized: it is a two-place predicate, indicating that one measurement (the first argument) should be less than the second argument:

```
Created meaning expression for meter:
meter
CL = AM(#SBJ.followed(SBJ,someone),
     #SBJ.maintained(SBJ,<(meter,1)))
```

Finally, the addition of **of them** completes the expression:

$$assoc\_meaning(\lambda x_{SBJ}.followed(x, someone), \lambda x_{SBJ}.maintained(x, <(distance\_from(them), 1))$$

Upon receiving **of**, the structure for **meter** begins to emerge– it refers to the distance from the subject to its single argument. The final form of the dependency graph is generated (Figure 2(e)) and the λ-free semantic representation is

```
CL = AM(#SBJ.followed(SBJ,someone,
     #SBJ.maintained(SBJ,
     <(distance_from(them),1)))
```

The final argument structure of **to follow** is:

$$\lambda x_{SBJ}.\lambda y_{OBJ}.followed(x, y)$$

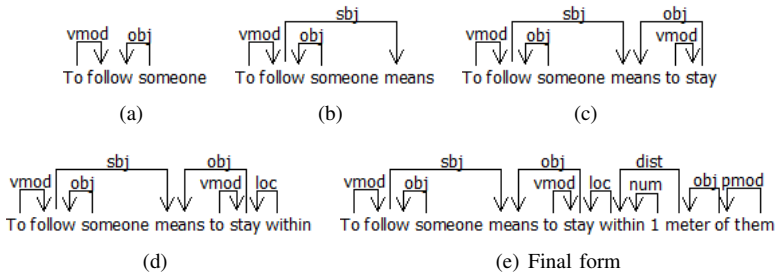

Fig. 2. Dependency graph progression for "to follow someone means to remain within 1 meter of them."

The anaphor **them** must be resolved, using *anaphora resolution*, a subset of coreference resolution in which a spoken anaphor (e.g., a pronoun) is resolved to its antecedent within the utterance. The following procedure is used for anaphora resolution: the current focus of attention (e.g., the unknown word) is maintained, along with a list of the discourse entities for each utterance. The most likely antecedent of a given anaphor is selected from these discoursal and focal entities based on cues such as sentence structure. By consulting our semantic knowledge base, we learn that **them** must refer to a noun. The only noun in this utterance, and thus the only possible antecedent, is **someone**.

$$assoc\_meaning(\lambda x_{SBJ}.followed(x,y)\&actor(y), \lambda x_{SBJ}.maintained(x, < (dist\_from(y),1))$$

We further assume that a verb's subject is an actor:

$$assoc\_meaning(\lambda x_{SBJ}.followed(x,y)\&actor(x) \&actor(y), \lambda x_{SBJ}.maintained(x, < (dist\_from(y),1))$$

This structure is then sent to the goal manager:

```
AM(["followed", [x,actor], [y,actor]],
   [maintain(<(distance_from(y),1))])
```

and from that point on the robot will respond correctly to requests that it follow someone.

IV. Validation

For validation, the algorithm described above was integrated into a robotic architecture which includes components for robot control, vision processing, planning, sensing, and others, in addition to various components for natural language processing. The entry point to the natural language subsystem is the CMU Sphinx speech recognizer. Recognized words are sent to the next NLU component, the incremental, deterministic data-driven dependency parser Mink, which is an incremental version of MaltParser [14]. The parser incorporates the part of speech tagger by first checking for a tag in the dictionary; if a tag is not found there, it uses a trigram model to assign one. Performing in tandem with the dependency parsing is a λ converter which produces semantic representations of the input, either by using meaning expressions found in a dictionary or, if such expressions are either not existent or not applicable, by creating new expressions consistent with the dependency structure. Finally, anaphora resolution and other post-processing are performed, still in the Mink component.

If unresolved references or unrecognized words are left, a dialogue component can generate verbal requests for additional information or clarification. Once a complete valid action expression has been obtained, the goal representation is sent to the goal manager component component for translation into an action script (preserving any restrictions on the argument types). The goal manager performs all action selection, goal management, and action execution tasks. Action scripts, which specify the sequence of actions to achieve a goal, constitute the "procedural knowledge base" of the goal manager. Multiple goals can be pursued concurrently (i.e., multiple action scripts can be executing concurrently); when resource conflicts arise, the goal manager resolves them based on goal priorities.

The enhanced robotic architecture allows a human operator to interact with a robot to explain (in natural language) action words it had not previously known, and allows the robot to both perform the new action and to store the procedural knowledge for future use. Specifically, in this demonstration a human interlocutor **H** asks the robot **R** to follow him, but the robot does not know the concept of "following someone". Hence, it asks for clarification, which the human provides. From the clarification, the robot learns the meaning of "following someone" and is able to immediately perform the associated action.

H I want you to follow me.
R I don't know how to follow.
H it means that you should stay within 1 m. of me
R okay

The demonstration was performed using an implementation of the algorithm on a Segway robot equipped with two USB web cameras (calibrated to be able to determine distance from stereo), a Hokuyu laser range finder (for obstacle detection), a microphone (for speech input), and a speaker (for speech output). The parser had access to the dictionary shown in Table I. A short video of this interaction can be viewed at `http://tiny.cc/segwaywalk`.

V. Discussion

The example demonstrates that the proposed algorithm allowed the robot to autonomously learn new verb meanings

TABLE I

MEANING EXPRESSIONS USED IN THE VALIDATION.

Word	meaning expressions
I	$speaker$
want	$\lambda y_{OBJ}.\lambda z_{ACT}.receive\_orders(y_{OBJ}, z_{ACT})$
you	$robot$
to	to
me	$speaker$
it	it
means	$\lambda x_{SBJ}.\lambda y_{OBJ}.assoc\_meaning(x_{SBJ}, y_{OBJ})$
that	$\lambda x_{PROP}.x_{PROP}$
should	$(\lambda x_{SBJ}.\lambda y_{VC}.y_{VC}(x_{SBJ})$
stay	$\lambda x_{LOC}.maintain(x_{LOC})$
within	$\lambda x_{NUM}.\lambda y_{DIST}.<(y_{DIST}, x_{NUM})$
of	$\lambda x_{OBJ}.of(x_{OBJ})$
one	1
meter	$\lambda z_{OBJ}.distance\_from(z_{OBJ})$

and immediately carry out the associated actions without any human intervention. And it can learn complex sequences of actions containing not-yet-known actions, continuing to ask for further specification until all action instructions in the sequence are fully defined. This flexibility is critical for scaling up to larger, more complex robotic systems that need to interact with humans in natural ways. However, since successful learning of a large number of action concepts depends on having a varied and flexible set of action building blocks from which to build more actions, it is important to select a good set of initial primitives from which compound meanings and complex actions can be derived. Also note that while many action verbs can be learned through instruction, some are more awkward to describe than others, so the results of instruction may vary. Understanding which types of verbs should be described in language versus taught using other means (e.g., visual demonstrations though gestures, etc.) is an interesting problem for future work. Ideally, an integrated learning system would be able to handle both verbal and non-verbal aspects of teaching (e.g., linguistic expressions with accompanying gestures fixing some part of the meaning description). Another important open question is how inter-instructor variability will affect the successful acquisition of verb meanings and the learning of control laws (e.g., instructors that skip important steps or give instructions that are ambiguous). This is clearly an empirical question that can be answered in user studies where different definitions for action verbs are collected from subjects. Such an evaluation is beyond what the space in the paper allows for, will ultimately aid in verifying that the proposed algorithm is sufficiently flexible to cope with different ways of expression verb meanings.

VI. CONCLUSIONS

We introduced a new algorithm that naturally integrates the natural language understanding system with the production of new action scripts from spoken instructions and demonstrated its effectiveness on a robot in the context of spoken dialogue-based human interactions using a scenario in which the robot successfully learned operational definitions for a previously unknown action verb. Different from other approaches to word learning where learning algorithms are external to natural language understanding, learning here is deeply integrated in the understanding process and only limited by the availability of known definitions from which to build new action meanings. Moreover, by virtue of being incremental, it is possible to detect lack of understanding (e.g., missing action verb definitions) early on, which is critical for a human-robot interaction contexts where rapid feedback is expected by human interlocutors. Given that the integrated learning system is robust and scalable due to its grounding in data-driven dependency parsing, the proposed approach is not only suitable for online learning in human-robot interaction scenarios, but also for large-scale learning of λ expressions from corpora. This will be evaluated in future work, in addition to evaluations in human subject experiments.

VII. ACKNOWLEDGMENTS

This work was in part funded by ONR MURI grant #N00014-07-1-1049 to the third author.

REFERENCES

[1] M. Marcus, B. Santorini, and M. Marcinkiewicz, "Building a large annotated corpus of english: the penn treebank," *Corpus Linguistics: Readings in a Widening Discipline (eds. G. Sampson and D. McCarthy)*, 2004.

[2] M. Scheutz, P. Schermerhorn, J. Kramer, and D. Anderson, "First steps toward natural human-like HRI," *Autonomous Robots*, vol. 22, no. 4, pp. 411–423, 2007.

[3] S. Lauria, T. Kyriacou, G. Bugmann, J. Bos, and E. Klein, "Converting natural language route instructions into robot executable procedures," in *Proc. of Ro-Man*, 2002, pp. 223–228.

[4] D. Hasegawa, R. Rzepka, and K. Araki, "A method for acquiring body movement verbs for a humanoid robot through physical interaction with humans," in *Proc. of AAAI*, 2009.

[5] J. Siskind, "Grounding lexical semantics of verbs in visual perception using force dynamics and event logic," *Journal of AI Research*, vol. 15, pp. 31– 90, 2001.

[6] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas, "From structured english to robot motion." in *Proc. of IEEE/RSJ IROS*, San Diego, 2007.

[7] K. Gold, M. Doniec, and B. Scassellati, "Learning grounded semantics with word trees: Prepositions and pronouns," in *Proc. of 6th IEEE ICDL*, 2007.

[8] K. Gold and B. Scassellati, "A robot that uses existing vocabulary to infer non-visual word meanings from observation," in *Proc. of AAAI*, 2007.

[9] S. Huffman and J. Laird., "Flexibly instructable agents." *Journal of Artificial Intelligence Research*, vol. 3, pp. 271–324, 1995.

[10] P. E. Rybski, J. Stolarz, K. Yoon, and M. Veloso, "Using dialog and human observations to dictate tasks to a learning robot assistant," *Journal of Intelligent Service Robots*, 2008.

[11] R. Cantrell, M. Scheutz, P. Schermerhorn, and X. Wu, "Robust spoken instruction understanding for HRI," in *Proc. of ACM/IEEE HRI*, 2010.

[12] J. Nivre, "Incrementality in deterministic dependency parsing," *Incremental Parsing: Bringing Engineering and Cognition Together. Workshop at ACL*, 2004.

[13] R. Johansson and P. Nugues, "Extended constituent-to-dependency conversion for english," in *Proc. of Nordic Conference on Computational Linguistics (NoDaLiDa) 2007*, 2007.

[14] J. Nivre, J. Hall, J. Nilsson, A. Chanev, G. Eryiğit, S. Kübler, S. Marinov, and E. Marsi, "MaltParser: A language-independent system for data-driven dependency parsing," *Natural Language Engineering*, vol. 13, no. 2, pp. 95–135, 2007.