DCG-UPUP-Away: Automatic Symbol Learning through Grounding to Unknowns

Mycal Tucker, Derya Aksaray, Rohan Paul, and Nicholas Roy

***Abstract*— This paper addresses the grounding problem whose objective is to understand the meaning of natural language commands within a robot's symbolic world. Existing techniques to solve the grounding problem typically assume that there is a fixed set of phrases or objects that the robot will encounter during deployment. However, the real world is full of confusing jargon and unique objects that are nearly impossible to anticipate and therefore train for. This paper introduces a model called the Distributed Correspondence Graph - Unknown Phrase, Unknown Percept - Away that enables to explicitly represent unknown phrases and objects as unknown, to reason about objects out of perception, and to learn new symbols in an online fashion. The effectiveness of the model is evaluated via simulations and real experiments in terms of grounding and learning new phrases.**

I. Introduction

Recently, there has been a great interest in human-robot teaming in civilian (e.g., at factories, hotels, hospitals, homes) and military (e.g., reconnaissance) applications. Communication plays an important role in effective teaming between humans and robots. One way of communication is via natural language, which provides a rich, intuitive, and flexible medium. Accordingly, the grounding problem in the literature addresses the question of how a robot can understand the meaning of a natural language command in the context of its world model (e.g., [1], [2]).

The existing methods to solve the grounding problem make two primary assumptions. First, they assume a fixed set of phrases that can constitute the commands and a fixed set of objects that exist in the world model. Thus, such methods typically fail to reason about unknown phrases or objects that have never been encountered (e.g., in the training process). Second, these methods often assume that the location of the object being grounded to is known (e.g., the phrases refer to the objects that are currently perceived or localized within a known map). As a result, a robot using these methods tends to pick the most likely perceived grounding rather than exploring its surroundings.

Note that the aforementioned assumptions do not typically reflect the reality. For example, humans tend to use context-specific lexicons that the general population does not recognize. Moreover, they often refer to objects whose locations may be unknown. To deal with such cases, training a robot to know the meaning of every possible word is infeasible and inefficient. Also, attempting to reason over the space of all possible maps is similarly computationally infeasible.

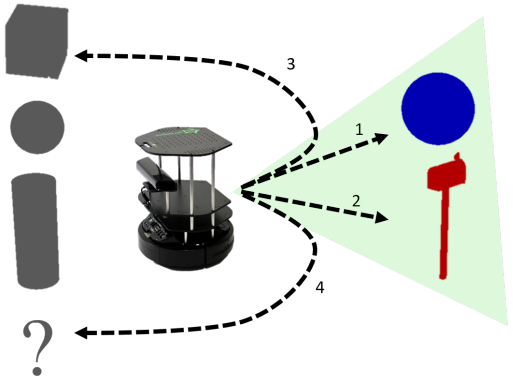


Fig. 1: An illustration of the proposed model DCG-UPUP-Away, which has been trained only with cubes, spheres, and cylinders; and it may ground phrases to known perceived objects (1), unknown perceived objects (2), known hypothetical objects (3), or unknown hypothetical objects (4).

This paper introduces a model called the Distributed Correspondence Graph - Unknown Phrase, Unknown Percept - Away (DCG-UPUP-Away), which relaxes two aforementioned assumptions by 1) explicitly modeling unknown phrases and percepts, and 2) creating hypothetical objects that can be out of perception. These two changes yield a model that can ground a large variety of phrases in complex environments, and they facilitate learning new words and objects in an online fashion. The proposed ideas are supported by a simulation study using commands generated by Amazon Mechanical Turk users. Also, the performance of the proposed model is evaluated via real experiments where a turtlebot is initially trained to recognize a small set of phrases and objects. The results demonstrate that the robot correctly grounds commands approximately 80% of the time while learning new concepts in an unsupervised manner.

The remainder of this paper is organized as follows: The preliminaries on probabilistic graphical models used for the grounding problem is introduced in Section II. The technical approach used in developing the DCG-UPUP-Away model is presented in Section III. The model is evaluated in Section IV. Existing research in natural language robotics and human-robot interaction that complements this work are reviewed in Section V. Finally, Sections VI concludes the paper by summarizing the contributions and future research.

All authors are with the Computer Science and Artificial Intelligence Laboratory (CSAIL), Massachusetts Institute of Technology, Cambridge, MA 02139, USA {mycal,daksaray,rohanp,nickroy}@csail.mit.edu

This work was partially supported by the Robotics Consortium of the U.S Army Research Laboratory under the Collaborative Technology Alliance Program.

II. Background

The work in this paper falls within the field of natural language grounding, which addresses the problem of correctly determining how phrases relate to the real world (e.g., the phrase "go to the cube" means approaching a physical cube). In the general formulation of the grounding problem, three sets must be considered: $\Gamma = \{\text{symbolic actions and objects}\}$ is the set of groundings, which represents what phrases may be grounded to; $\Lambda = \{\text{English phrases}\}$ is the set of phrases, which represents what phrases natural language sentences may be composed of; and $\Upsilon = \Gamma^O \times \{\text{object attributes}\}$ is the cartesian product of the set of objects (i.e., $\Gamma^O \subset \Gamma$) and the object attributes (e.g., color, location), which represents the world model in which the phrases may be grounded.

Given a natural language command $\boldsymbol{\lambda}$, which is a vector of phrases from the set Λ and has a length of $|\boldsymbol{\lambda}|$, the general grounding problem can be formulated as a probability maximization problem

$$\boldsymbol{\gamma}^* = \arg\max_{\boldsymbol{\gamma} \in \Gamma^{|\boldsymbol{\lambda}|}} p(\boldsymbol{\gamma} | \boldsymbol{\lambda}, \Upsilon), \tag{1}$$

where $\boldsymbol{\gamma} \in \Gamma^{|\boldsymbol{\lambda}|}$ is a vector of groundings with a length of $|\boldsymbol{\lambda}|$, and Υ denotes the world model. In this formulation, the optimal vector of groundings $\boldsymbol{\gamma}^*$ is the one with maximum likelihood, given a command $\boldsymbol{\lambda}$ and a world model Υ.

In practice, the domains of Γ, Λ, and Υ in (1) typically include elements from previously seen examples. For example, rather than allowing the set of phrases Λ to include all words in a dictionary, Λ is generally assumed to only contain words that have appeared in the training examples. Moreover, solving (1) is a hard combinatorial optimization problem due to the diversity in language and world. One way to tackle this issue is to construct probabilistic graphical models based on the linguistic structure of the commands. For example, the Generalized Grounding Graph (G3) model [1] is a factor graph that is trained from a corpus of labeled examples to ground language commands with objects, locations, and paths.

An alternative model is the Distributed Correspondence Graph (DCG) [2], which infers the most likely set of planning constraints from language commands (rather than grounding phrases to particular actions, objects, or paths as in G3 model). In particular, the DCG model has the following properties: First, the DCG decouples motion planning from the grounding problem. To this end, it discards the notion of grounding to specific actions or objects by instead only grounding to constraints relative to known objects (e.g. the area near a cube). Then, it passes the constraints to a motion planner. Second, the DCG allows only the phrases composed entirely of the words that have already been encountered in training. Third, the DCG only considers the perceived objects rather than a full model of the world including the unobserved objects. Fourth, ternary correspondence variables ϕ_{ij} are introduced to represent whether the i^{th} phrase λ_i from the overall command $\boldsymbol{\lambda}$ corresponds to a grounding γ_{ij}. For a given phrase λ_i, ϕ_{ij} is set to *Active* if λ_i refers to constraint γ_{ij} (e.g., the area near the cube), *Inverted* if λ_i refers to the opposite of γ_{ij} (e.g. the area far from the cube), or *Inactive* if λ_i has no bearing on γ_{ij} (e.g. the area near a sphere). Fifth, for the sake of computational efficiency, the overall inference is factored using conditional independence according to the structure of the parse tree of $\boldsymbol{\lambda}$.

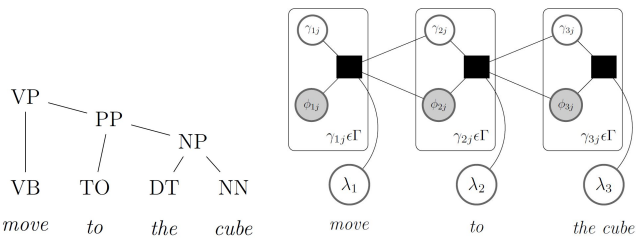
Accordingly, the optimization problem solved over a DCG model becomes

$$\boldsymbol{\phi}^* = \arg\max_{\phi_{ij} \in \boldsymbol{\phi}} \prod_{i}^{|\boldsymbol{\lambda}|} \prod_{j}^{|\Phi_i|} p(\phi_{ij} | \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP}), \tag{2}$$

where $\lambda_i \in \Lambda_{KN}$ is the i^{th} phrase in command $\boldsymbol{\lambda}$ and Λ_{KN} is the set of phrases with known (previously seen) words; Φ_i is the set of correspondence variables of λ_i, $\phi_{ij} \in \Phi_i$ is the j^{th} correspondence variable of λ_i; γ_{ij} is the j^{th} grounding of λ_i; $\Upsilon_{KP} = \{\text{object attributes}\} \times \Gamma_{KP}$ is the world model consisting of the object attributes and the set of known perceived symbolic objects Γ_{KP}; and $\Gamma_{c_{ij}}$ is the set of child groundings of γ_{ij}. Note that $\Gamma_{c_{ij}}$ is defined as the set of groundings for the immediate children phrases (leftmost descendants) of the parent phrase λ_i in the parse tree of the natural language command.

For example, consider Fig. 2a that shows the parse tree of a simple command, and Fig. 2b that shows the corresponding DCG graphical model. The child grounding of the phrase "move" is the grounding for the phrase "to." Similarly, the child grounding of the phrase "to" is the grounding for the phrase "the cube" (determiners such as "the" may be collapsed into their nouns). Thus, in this example, each grounding has exactly one child grounding, yielding the inter-plate structure in Fig. 2b. Examining the parse tree also reveals why the factorization in (2) is reasonable: the meaning "move" should be conditionally independent of the noun "cube" given the prepositional phrase. After all, the correct grounding of the word "move" is an action that does not depend on whether the target is a cube or a sphere, but it does depend on the position of the cube.

Finally, one must consider the factor function $\Psi : \Phi \times \Gamma \times \Lambda \times \Gamma \times \Upsilon \to \mathbb{R}$ within each plate that determines the most likely configuration of each $\phi_{ij} \in \Phi$ given $\gamma_{ij} \in \Gamma$, $\lambda_i \in \Lambda$, $\Gamma_{c_{ij}} \subset \Gamma$, and $\Upsilon_{KP} \subset \Upsilon$. Accordingly, one may rewrite (2)



(a) Parse tree for the command "move to the cube"

(b) The DCG graphical model for the parse tree in Fig. 2a

Fig. 2: An illustration of a parse tree and the corresponding DCG model.

as

$$\boldsymbol{\phi}^* = \underset{\phi_{ij} \in \boldsymbol{\phi}}{\arg\max} \prod_{i}^{|\boldsymbol{\lambda}|} \prod_{j}^{|\Phi_i|} \Psi(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP}), \tag{3}$$

where Ψ is a log-linear model (LLM) composed of a weighted combination of hand-coded binary functions, that is,

$$\Psi(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP}) = \frac{\exp\Big(\sum\limits_{f \epsilon F} \mu_f f(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP})\Big)}{\sum\limits_{\phi_{ij} \in \{-1,0,1\}} \exp\Big(\sum\limits_{f \epsilon F} \mu_f f(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP})\Big)}, \tag{4}$$

where each binary function f belongs to a set of hand-coded binary features that evaluate specific traits about a grounding (e.g., whether the word "cube" appears in $\boldsymbol{\lambda}$), and μ_f is the weighting of each f. In this work, the weights μ_f are learned in a training procedure via the Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm.

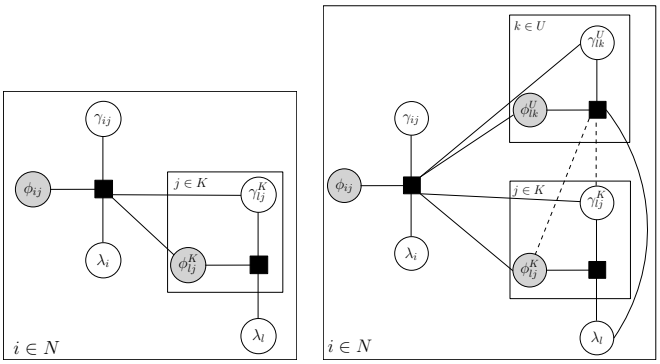
III. Technical Approach

In the previous section, it has been stated that the DCG model can be efficiently used in grounding problems where the objects and phrases are known and the phrases are grounded to only perceived objects. In this paper, the proposed model DCG-UPUP-Away enables the solution of a more generalized grounding problem such that 1) the phrases and objects may be known or unknown, and 2) the phrases may be grounded to objects that are out of perception. To this end, the following sections detail how to ground unknown phrases or objects, how to incrementally learn new objects and phrases, how to hypothesize groundings out of perception, and how to involve adjective attributes into grounding to avoid ambiguities.

A. Grounding Unknown Phrases or Objects

In this paper, the phrases (i.e., nouns) are defined as unknown if they have never grounded to a known object type in the training files. By maintaining a set of known nouns, one can easily determine whether or not a phrase is unknown. Similarly, the objects are defined as unknown if no object of the same class has ever appeared in the training data. Note that the DCG model does not explicitly represent the unknown symbols (i.e., objects and phrases) as in Fig. 3a, so it becomes computationally infeasible to ground them. An alternative model can be generated by decoupling the unknown symbols from the known symbols as in Fig. 3b where each known symbol is connected to each unknown symbol (as dashed edges). While such a model represents all possible correlations between the known and unknown symbols, solving the grounding problem over this graph becomes computationally expensive due to strong connectivity.

Motivated by the idea of decoupling the unknown symbols from the known ones, we propose the DCG-UPUP model which exhibits a less connected graph than the one in Fig. 3b (by removing the dashed edges). This simpler representation is mainly based on the following assumption:



(a) Without explicit unknown grounding variables.

(b) With explicit unknown grounding variables.

Fig. 3: Factor graph representations. Input instruction is parsed into N phrases where λ_l represents a child phrase of parent phrase λ_i. Superscripts K and U denote known and unknown variables, respectively.

given the language command $\boldsymbol{\lambda}$ and the world model Υ, the unknown groundings become conditionally independent from the known groundings because the variables $\boldsymbol{\lambda}$ and Υ are sufficient to predict the unknown groundings.

In light of going from (2) to (3), the factored objective function for the DCG-UPUP model can be written as in (5) where the the domain of the world model is extended to known and unknown perceived objects (i.e., $\Upsilon_{KP} \cup \Upsilon_{UP}$).

$$\boldsymbol{\phi}^* = \underset{\phi_{ij} \in \boldsymbol{\phi}}{\arg\max} \prod_{i}^{|\boldsymbol{\lambda}|} \prod_{j}^{|\Phi_i|} \Psi(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP}), \tag{5}$$

where each feature function is an LLM model as

$$\Psi(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP}) = \frac{A_U}{B_U}, \tag{6}$$

where

$$A_U = \exp\Big(\sum_{f \in F_{\text{DCG}}} \mu_f f(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP}) + \sum_{f' \in F_U} \mu_{f'} f'(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP})\Big),$$

$$B_U = \sum_{\phi_{ij} \in \{0,1\}} \exp\Big(\sum_{f \in F_{\text{DCG}}} \mu_f f(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP}) + \sum_{f' \in F_U} \mu_{f'} f'(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP})\Big),$$

F_{DCG} and F_U are the sets of hand-coded binary features used in the DCG model and for detecting unknown phrases or objects, respectively.

B. Incremental Unsupervised Learning

While the DCG-UPUP model can reason about unknown phrases and objects, it can also learn new symbols permanently. This is mainly achieved by the following unsupervised learning procedure: whenever an unknown phrase is

grounded to an unknown object, a new type is created based on the new phrase. Then, a new training example, in which the phrase grounds to the new type, is generated. Accordingly, the LLM models are retrained with the expanded set of training examples. Consequently, when the newly generated objects (or phrases) are encountered again, they become known symbols. Hence, the DCG-UPUP model can learn to associate a new phrase with a new type.

C. Hypothesized Groundings out of Perception

The previous section presented that the DCG-UPUP model can explicitly represent the unknown phrases and objects. However, its performance is limited since the robot can only ground to the perceived objects. As an extension of this model, we propose the DCG-UPUP-Away, which enables to ground phrases to objects out of perception.

The main process to include hypothetical objects to the model is as follows: after populating a world model by using the sensors of the robot, a single instance of every known object type, as well as one instance of an unknown object, are added to the world model and labeled as hypothetical objects. The resulting graphical model for the DCG-UPUP-Away is illustrated in Fig. 4c, where the nouns may ground to 1) known and perceived objects, 2) unknown and perceived objects, 3) known and hypothetical objects, and 4) unknown and hypothetical objects. As a comparison, Fig. 4b illustrates the DCG-UPUP model, where the nouns can be grounded to only known perceived and unknown perceived objects. Moreover, Fig. 4a presents the DCG model where the nouns can be grounded to only known perceived objects.

Similar to (5), the factored objective function for the DCG-UPUP-Away model can be written by extending the world model to known perceived, unknown perceived, known hypothetical, and unknown hypothetical objects (i.e., $\Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}$) as follows:

$$\boldsymbol{\phi}^* = \underset{\phi_{ij} \in \boldsymbol{\phi}}{\arg\max} \prod_{i}^{|\boldsymbol{\lambda}|} \prod_{j}^{|\Phi_i|} \Psi(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}), \tag{7}$$

where each feature function is an LLM model as:

$$\Psi(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}) = \frac{A_{UH}}{B_{UH}} \tag{8}$$

where

$$\begin{aligned} A_{UH} = \exp\Big(& \sum_{f \in F_{\text{DCG}}} \mu_f f(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}) \\ & + \sum_{f' \in F_U} \mu_{f'} f'(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}) \\ & + \sum_{f'' \in F_H} \mu_{f''} f''(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}) \Big), \end{aligned}$$

$$\begin{aligned} B_{UH} = \sum_{\phi_{ij} \in \{0,1\}} \exp\Big(& \sum_{f \in F_{\text{DCG}}} \mu_f f(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}) \\ & + \sum_{f' \in F_U} \mu_{f'} f'(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}) \\ & + \sum_{f'' \in F_H} \mu_{f''} f''(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}) \Big), \end{aligned}$$

and F_H is the set of hand-coded binary features to detect if an object is hypothetical.

D. Adjective-Attribute Heuristics

One way to improve the grounding performance of the DCG-UPUP-Away model is to allow the association between the natural language adjectives and the object properties. For example, if there exist two cube type objects in the world, one way to distinguish them from each other is to consider their properties such as color or size. In this section, we present how to include color information into the solution of grounding problem over the DCG-UPUP-Away. To this end, two additional features are introduced: the feature f_{word} detects whether the language command contains a color adjective, and the feature f_{color} checks the color property of an object.

Note that the factored objective function in this case has exactly the same form as in (7) where each feature function $\Psi(.)$ is defined as follows:

$$\Psi(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}) = \frac{A_{UHC}}{B_{UHC}} \tag{9}$$

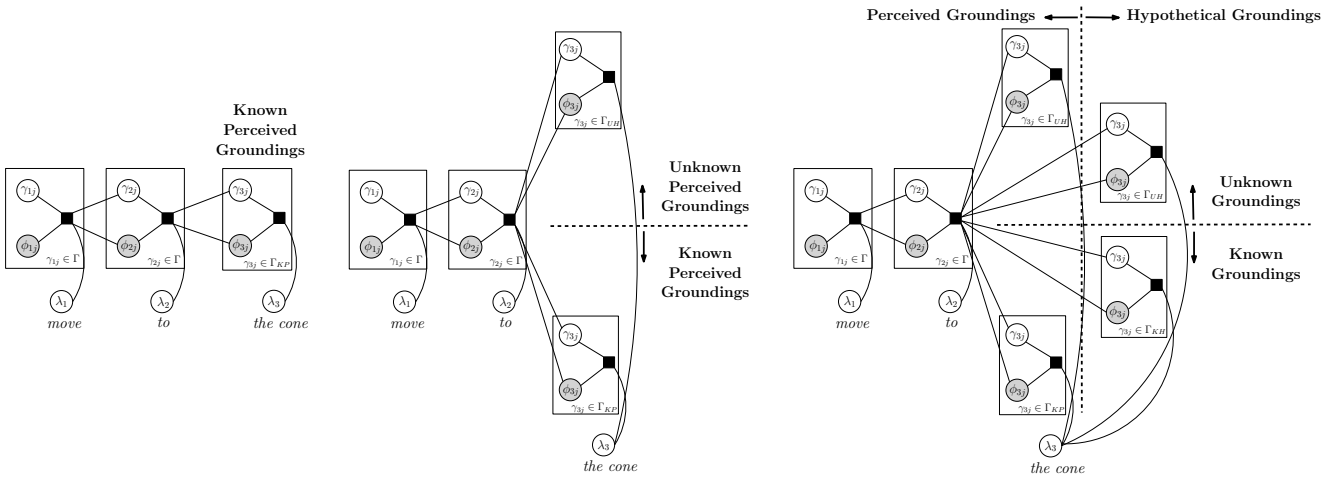
where

$$\begin{aligned} A_{UHC} = \exp\Big(& \sum_{f \in F_{\text{DCG}}} \mu_f f(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}) \\ & + \sum_{f' \in F_U} \mu_{f'} f'(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}) \\ & + \sum_{f'' \in F_H} \mu_{f''} f''(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}) \\ & + \sum_{f''' \in F_C} \mu_{f'''} f'''(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}) \Big), \end{aligned}$$

$$\begin{aligned} B_{UHC} = \sum_{\phi_{ij} \in \{0,1\}} \exp\Big(& \sum_{f \in F_{\text{DCG}}} \mu_f f(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}) \\ & + \sum_{f' \in F_U} \mu_{f'} f'(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}) \\ & + \sum_{f'' \in F_H} \mu_{f''} f''(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}) \\ & + \sum_{f''' \in F_C} \mu_{f'''} f'''(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon_{KP} \cup \Upsilon_{UP} \cup \Upsilon_{KH} \cup \Upsilon_{UH}) \Big), \end{aligned}$$

and F_C is the set of hand-coded binary features for detecting color phrases or properties (i.e., $F_C = f_{color} \cup f_{word}$).

Finally, the pseudo-code for grounding and learning new symbols over the DCG-UPUP-Away model is presented in Alg. 1. First, the graphical model M is initialized and trained with the initial set of training data (lines 1-6). Grounding to unknown symbols and/or hypothetical objects are tackled between the lines 7-24. In particular, the world model is updated with the perceived objects (line 8) and then hypothetical objects are added (line 9). The natural language command is entered as an input (line 10). Using the model M with the language command $\boldsymbol{\lambda}$ and the most recent world model Υ, the grounding problem is solved (line 11). If the obtained grounding is hypothetical, then the robot initiates the exploration (line 13); otherwise the robot moves towards the grounded perceived object (line 15). Note that



(a) The DCG model (b) The DCG-UPUP model (c) The DCG-UPUP-Away model

Fig. 4: The graphical models constructed for the command "*move to the cone*".

the exploration in this research is considered as the robot rotating in its current location. At the end of solving the grounding problem, a new training file is generated based on the given command $\boldsymbol{\lambda}$, the resulting grounding $\boldsymbol{\gamma}^*$, and the current world model Υ (line 16). If there exists an unknown phrase in the given command and the resulting grounding is not hypothetical, then a new object type (or grounding variable) is generated (line 18) and the set of grounding variables as well as the feature sets are updated (lines 19-22). Finally, the newly generated object type γ' replaces the unknown object type in training file T_u (line 23), and the model M is retrained with the new set of training data. Consequently, the initially unknown object becomes a known object in the new model M in the further iterations.

Algorithm 1 Grounding/Learning over DCG-UPUP-Away

1: **procedure** DCG-UPUP-AWAY
2: $M \leftarrow$ new DCG-UPUP-Away
3: $M.\Gamma \leftarrow$ init_groundings
4: $M.F \leftarrow$ init_features
5: $T \leftarrow$ init_training
6: M.train$(M.\Gamma, M.F, T)$
7: **while** true **do**
8: $\Upsilon \leftarrow$ perceive_objects(camera, $M.\Gamma$)
9: $\Upsilon \leftarrow \Upsilon +$ hypothesize_objects$(M.\Gamma)$
10: $\boldsymbol{\lambda} \leftarrow$ get_nl_command()
11: $[\boldsymbol{\phi}^*, \boldsymbol{\gamma}^*] \leftarrow M$.ground$(\boldsymbol{\lambda}, \Upsilon)$
12: **if** $\boldsymbol{\gamma}^*$.is_hypothesized() **then**
13: spin_in_place()
14: **else**
15: drive_to$(\boldsymbol{\gamma}^*)$
16: $T_u \leftarrow$ gen_unsupervised_training$(\boldsymbol{\lambda}, \boldsymbol{\gamma}^*, \Upsilon)$
17: **if** is_unknown$(\boldsymbol{\lambda})$&!$\boldsymbol{\gamma}^*$.is_hypothesized() **then**
18: $\gamma' \leftarrow$ new_grounding$(\boldsymbol{\lambda}, \boldsymbol{\gamma}^*, \Upsilon)$
19: $M.\Gamma \leftarrow M.\Gamma + \gamma'$
20: $M.F \leftarrow M.F + f_{\text{word}}(\boldsymbol{\lambda}[\text{noun}])$
21: $M.F \leftarrow M.F + f_{\text{color}}(\boldsymbol{\gamma}^*[\text{color}])$
22: $M.F \leftarrow M.F + f_{\text{obj}}(\boldsymbol{\gamma}^*[\text{obj}])$
23: $T_u \leftarrow$ replace_unknown(T_u, γ')
24: $T \leftarrow T + T_u$
25: M.train$(M.\Gamma, M.F, T)$

IV. EVALUATION

The performance of the DCG-UPUP-Away model is demonstrated in two experiments. First, a simulated turtlebot within randomly generated simulated environments is given a series of user-generated natural language commands. Second, an actual turtlebot is given specific commands in a laboratory environment in order to demonstrate novel behaviors enabled by the DCG-UPUP-Away model. Both experiments assume a perfect object recognizer that translates the raw sensor data into a world model Υ that can be used by the DCG-UPUP-Away model, as well as an initial set of hand-labeled training examples for training the LLM to ground cubes, spheres, and cylinders. In all trials, training the model with 53 positive examples took less than 1 minute on a Lenovo Thinkpad X1 Carbon, and grounding a command took under 40 seconds.

A. Experimental Setup

The simulated testing environments are randomly generated in Gazebo. Ten worlds are created, and each is populated with a random collection of objects in randomized locations. There are 8 possible object types (including cubes, spheres, and cylinders) in 3 possible colors, for a total of 24 objects. Each object has a 15% chance of being added to a given map. Using such a procedure to generate environments coupled with the limited field of view of the turtlebot has caused 87% of the objects to be placed outside the initial field of view of the robot, which demonstrates the need for the ability to ground commands to hypothesized objects.

After generating the 10 worlds, the screenshots of a world with a highlighted single object are uploaded to Amazon Mechanical Turk. For each image, the users were instructed to write a command "for approaching the highlighted object." These image-command pairs were saved for evaluating whether a robot, when placed in the corresponding simulated world and given the natural language command, successfully approaches the correct object. An example screenshot, with an annotation supplied by a user, is shown in Fig. 5.

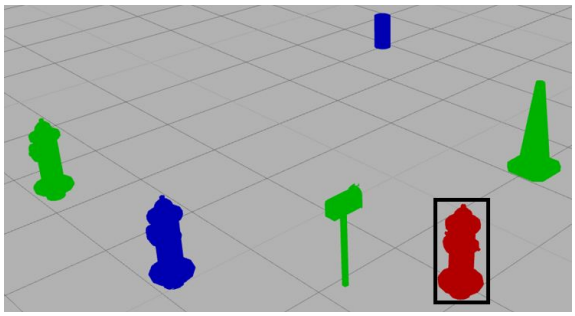
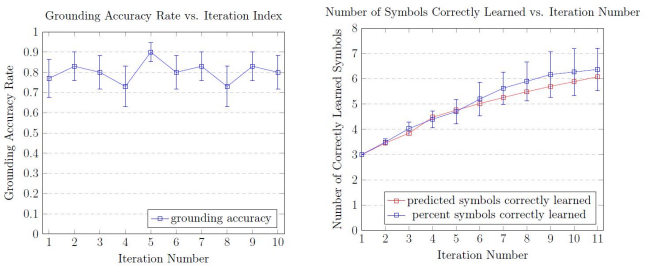


Fig. 5: A simulated world with a highlighted object presented on Amazon Mechanical Turk, labeled by a user as "Move to the red fire hydrant."



(a) Overall grounding accuracy. (b) Number of learned symbols.

Fig. 6: The performance results of the simulation study.

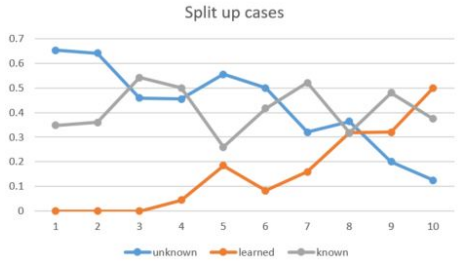


Fig. 7: The percentage of symbols during the simulations.

Ten image-command pairs are randomly selected without any replacement from the pool of all pairs. Note that a trial along the paper refers to 10 ordered pairs, and each specific pair is called one iteration. Accordingly, 30 trials are generated, each consisting of 10 iterations, for a total of 300 evaluations. When executing a trial, the turtlebot is first trained on the initial, hand-curated training set. The turtlebot is then given the natural language command from the first iteration, and then retrained using the initial data supplemented by unsupervised training examples generated by the first iteration. The retrained turtlebot is given the command from the next iteration, and appropriately retrained after each execution until all 10 iterations have been executed.

The metrics we consider for the performance of the model are the grounding accuracy (how likely the DCG-UPUP-Away model correctly grounds a phrase) and the number of known symbols. We further divide the grounding accuracy results to examine when phrases are grounded to known, unknown, or learned objects.

B. Grounding Accuracy

As discussed previosuly, the turtlebot is retrained between iterations, thus the grounding accuracy may change as a function of iteration number. In fact, the mean grounding accuracy remains between 70% and 90% across all iterations, as shown in Figure 9d. Although the overall grounding accuracy remains relatively constant, the underlying behavior within the DCG-UPUP-Away model changes over the course of a trial. For example, Fig. 9f illustrates 3 curves showing what fraction of correctly grounded phrases refer to known objects, unknown objects, or learned objects as a function of iteration number. In the first iteration, nearly 70% of correctly grounded commands refer to known objects, but by the 10^{th} iteration that number has fallen to nearly 10%, replaced almost entirely by correctly grounding to learned objects.

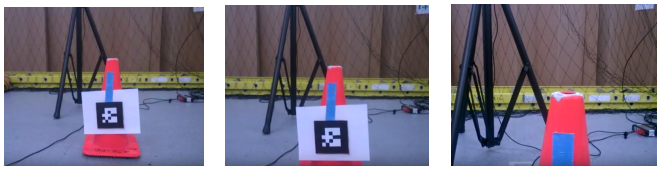
C. Learned Symbols

In order to better examine the learning behavior exhibited by the DCG-UPUP-Away model, the other performance metric considered is the number of correctly known symbols. Note that the symbols may be incorrectly learned by associating a phrase with the wrong sort of object due to the nature of unsupervised learning. Initially, the turtlebot is trained with cubes, spheres, and cylinders, but the genrated environments may contain up to 5 additional object types (i.e., fire hydrants, drills, mailboxes, door handles, and traffic cones). Whenever an unknown phrase is grounded to such an unknown object, the turtlebot learns the new symbol. Thus, one may calculate the expected number of known symbols as a function of the iteration number using combinatorics to count how many unknown objects are present. The recorded number of correctly learned symbols are plotted in Fig. 9e in blue, as well as the expected number in red.

As expected, the blue curve starts at 3 (for the cube, sphere, and cylinder), and stochastically monotonically increases. In 10% of trials, all 8 symbols were correctly learned. In other trials the DCG-UPUP-Away model incorrectly grounded unknown phrases (and therefore learned an incorrect symbol) or the 10 iterations collectively never referred to the five initially unknown objects, preventing the DCG-UPUP-Away model from ever learning the new symbol. Furthermore, learning symbols correctly improves the grounding accuracy: for each additional correctly learned symbol, the turtlebot is over 4% more likely to correctly ground a command.

D. Hardware Demonstration

In addition to the simulation studies, the DCG-UPUP-Away model was tested on an actual turtlebot in a laboratory setting. The turtlebot was placed facing a cone (unknown). In addition, a cube (known) and a crate (unknown) were located behind the turtlebot. All objects were labeled with the AR-track tags [3], which were used to generate the world model Υ from a kinect camera mounted on the turtlebot.



(a) t= 0 sec. (b) t= 45 sec. (c) t= 50 sec.

Fig. 8: An illustration of learning new symbol. The turtlebot initially does not know what a cone is, a command is given as "move towards the cone". (a) Since there is an unknown object in its perceived world, it grounds the unknown phrase "cone" to the unknown object, (b,c) it drives to the cone.

Three natural language commands were used to demonstrate all capabilities of the DCG-UPUP-Away model. First, the turtlebot was given the command "move towards the cone." The turtlebot drove to the cone, demonstrating that it perceived the cone as unknown, recognized the phrase "cone" as unknown, and grounded the unknown phrase to the unknown object. Thus, a command was correctly grounded to an unknown perceived object as illustrated in Fig. 8. Second, the turtlebot was given the command "move towards the cube." The turtlebot rotated in place until the cube came in perception, and then approached the cube. In other words, the command was first grounded to a known hypothesized object, and then it was grounded to a known perceived object once the cube was seen. Finally, the turtlebot was given the command "move towards the crate." Once again, the turtlebot explored its surrounding by rotating at its current location and drove to the crate once it perceived it (as illustrated in Fig. 9). The experimental results demonstrate two important behaviors: 1) the turtlebot must have learned what a cone was, otherwise the unknown phrase ("crate") would have been grounded to the cone, and 2) the turtlebot grounded the command to an unknown hypothesized object until the crate was perceived. The interested reader is referred to the following link [1] for the videos corresponding to these experiments.

E. Limitations

The previous sections demonstrated that the proposed model DCG-UPUP-Away results in the successful execution of various natural language commands. This section discusses the main limitations of the model. In particular, the most obvious limitation of the DCG-UPUP-Away model is the assumption of referring an unknown phrase to the first perceived unknown object. One strategy to relax this assumption has been explored in Section III-D by associating language adjectives with object properties. However, a more sophisticated strategy is required for generalizable solutions. Moreover, the DCG-UPUP-Away model assumes a one-to-one correspondence between unknown phrases and unknown objects; thus it cannot, for example, learn synonyms by grounding unknown phrases to the known object types.

[1] https://www.youtube.com/playlist?list=PL8sYMUToK9s6dAu3qMHHOef8FyhOnDK4E

V. Related Works

This work is closely related to solving the grounding problem over probabilistic graphical models that contain three main variables: grounding variables, language command, and the world model. In the previous works, the domains of the grounding variables, the language command, and the world model have been restricted to the known phrases and the perecived groundings [1], [2]. Furthermore, although some works reason about unknown environments, many probabilistic grounding techniques assume fully observable worlds [4], [5].

In this paper, we propose an unsupervised learning process to learn new symbols (i.e., new phrases or objects). An alternative way of grounding unknown or ambiguous symbols can be done via human-robot dialogue (e.g., [9]). For example, Ros et al. broadly approach resolving language ambiguity using two techniques. First, a robot attempts to model the human's perspective on the scene to determine which objects may be visible to the human. This technique relies on insights from child development studies that show how children employ such reasoning on their own and has been successfully used in other robotics literature [10], [11], [12], [13]. The second strategy relies on the robot asking a human for more information. For example, the robot may ask for spatial relations or object features in distinguishing between objects. Choosing exactly which question to ask, of course, requires reasoning about what information best discriminates among potential groundings (e.g., [14]). For example, the entropy of the probability distribution over groundings is used to estimate the grounding uncertainty in [14]. Accordingly, higher entropy leads to more questions which improves the grounding accuracy rate. Note that a critical issue in robotic question-asking is the proper balance between too many questions and not enough questions while simultaneously determining what sorts of question to ask [15], [16].

One common approach for autonomous language learning provides a robot with semantic representations of the world that must be associated with language. Such associations may be formally expressed using predicate logic, but ultimately the problem of language acquisition is reframed as a mapping problem from words to pre-defined semantics (e.g., [17], [18], [19], [20]). Unfortunately, hand-labeled representations necessarily require intensive human involvement in generating training data [21]. As a result, some studies consider the opposite approach and try to associate words directly to objects or actions without creating formal symbolic representations. For example, using online raw video data and sentences, a system is able to learn shape categories without being told ahead of time that four right angles define rectangular objects [22], [23].

Finally, there exist some studies in the literature considering the idea of hypothesizing objects out of perception. For example, Duvallet et al. uses a framework to propose a latent map that is partially observed by the language command [4]. Accordingly, in the example of a ball outside the door, the

(a) t=0 sec. (b) t=25 sec. (c) t=50 sec. (d) t=70 sec. (e) t=95 sec. (f) t=120 sec.

Fig. 9: An illustration of grounding to a hypothetical object. The robot initially knows all objects in the world other than a crate. The turtlebot is given a command as "move towards the crate". (a) First, it does not see an unknown object in its perceived world so it creates a hypothetical unknown object, (b,c,d) it explores the world by rotating at its current location until it perceives an unknown object, (e) It perceives an unknown object and grounds to it, (f) it drives to the crate.

phrase "pick up the ball outside the door" generates a region of high probability near the door and low probability further away. Sampling from this distribution, as well as updating the distribution as more observations are made, yields a useful map to plan in. Similarly, some other works generate distributions over maps or exactly place objects in unknown environments if their locations are uniquely described [24], [5].

VI. Conclusion

This paper addressed the problem of understanding natural language commands within a robot's symbolic world model. The main contribution of the paper was to propose a new probabilistic graphical model called DCG-UPUP-Away, which allows the explicit representation of 1) unknown phrases or objects, and 2) hypothetical objects that can be out of field. Moreover, the proposed model has the capability to learn new symbols in an online fashion, so the learned phrases or objects become known when they are encountered again. The performance of the proposed model was evaluated via simulations and real experiments, where a turtlebot was used and various natural language commands were given. The results indicated that the DCG-UPUP-Away model can ground correct objects approximately 80% of the time. Some potential future directions can be extending the model to reason about multiple unknown (hypothetical) objects or understanding the synonyms of known objects.

References

[1] S. Tellex, T. Kollar, S. Dickerson, M. Walter, A. Banerjee, S. Teller, and N. Roy, "Understanding natural language commands for robotic navigation and mobile manipulation," in *National Conference on Artificial Intelligence*, 2011.

[2] T. Howard, S. Tellex, and N. Roy, "A natural language planner interface for mobile manipulators," In *International Conference on Robotics and Automation*, June 2014.

[3] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2011, pp. 3400–3407.

[4] F. Duvallet, M. Walter, T. Howard, S. Hemachandra, J. H. Oh , S. Teller, N. Roy, and A. T. Stentz , "Inferring maps and behaviors from natural language instructions," in *International Symposium on Experimental Robotics*, June 2014.

[5] T. Williams, R. Cantrell, G. Briggs, P. Schermerhorn, and M. Scheutz, "Grounding natural language references to unvisited and hypothetical locations," 2013.

[6] J. MacGlashan, M. Babes-Vroman, M. desJardins, M. Littman, S. Muresan, S. Squire, S. Tellex, D. Arumugam, and L. Yang, "Grounding english commands to reward functions," in *Proceedings of Robotics: Science and Systems*, Rome, Italy, July 2015.

[7] M. Beetz, U. Klank, I. Kresse, A. Maldonado, L. Mosenlechner, D. Pangercic, T. Ruhr, and M. Tenorth, "Robotic roommates making pancakes," in *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on*, Oct 2011, pp. 529–536.

[8] M. Bollini, S. Tellex, T. Thompson, N. Roy, and D. Rus, *Interpreting and Executing Recipes with a Cooking Robot*. Heidelberg: Springer International Publishing, 2013, pp. 481–495.

[9] R. Ros, S. Lemaignan, E. A. Sisbot, R. Alami, J. Steinwender, K. Hamann, and F. Warneken, "Which one? grounding the referent based on efficient human-robot interaction," in *19th International Symposium in Robot and Human Interactive Communication*, Sept 2010, pp. 570–575.

[10] H. Moll and M. Tomasello, "Twelve- and 18-month-old infants follow gaze to spaces behind barriers," In *British Journal of Developmental Psychology*, 2004.

[11] ——, "Level 1 perspective-taking at 24 months of age," In *Developmental Science*, 2006.

[12] J. G. Trafton, N. L. Cassimatis, M. D. Bugajska, D. P. Brock, F. E. Mintz, and A. C. Schultz, "Enabling effective human-robot interaction using perspective-taking in robots," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 35, pp. 460–470, 2005.

[13] J. G. Trafton, A. C. Schultz, M. Bugajska, and F. Mintz, "Perspective-taking with robots: experiments and models," in *ROMAN 2005. IEEE International Workshop on Robot and Human Interactive Communication, 2005.*, Aug 2005, pp. 580–584.

[14] R. Deits, S. Tellex, P. Thaker, D. Simeonov, T. Kollar, and N. Roy, "Clarifying Commands with Information-Theoretic Human-Robot Dialog," *Journal of Human-Robot Interaction*, vol. 2, no. 2, pp. 58–79, 2013.

[15] T. W. Fong, C. Thorpe, and C. Baur, "Robot, asker of questions," *Robotics and Autonomous Systems*, 2003.

[16] N. Roy, J. Pineau, and S. Thrun, "Spoken dialogue management using probabilistic reasoning," in *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL-2000)*, Hong Kong, 2000.

[17] A. S. Clark, "Unsupervised language acquisition: Theory and practice," 2001.

[18] J. M. Siskind, "Lexical acquisition in the presence of noise and homonymy," in *Proceedings of the 12th National Conference on Artificial Intelligence, Seattle, WA, USA, July 31 - August 4, 1994, Volume 1.*, 1994, pp. 760–766.

[19] J. M. Zelle and R. J. Mooney, "Learning to parse database queries using inductive logic programming," in *AAAI/IAAI*. Portland, OR: AAAI Press/MIT Press, August 1996, pp. 1050–1055.

[20] R. Ge and R. J. Mooney, "A statistical semantic parser that integrates syntax and semantics," in *Proceedings of the Ninth Conference on Computational Natural Language Learning*, ser. CONLL '05, 2005, pp. 9–16.

[21] R. J. Mooney, "Learning to connect language and perception," in *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, ser. AAAI'08, 2008, pp. 1598–1601.

[22] C. Yu and D. H. Ballard, "A multimodal learning interface for grounding spoken language in sensory perceptions," *ACM Trans. Appl. Percept.*, vol. 1, no. 1, pp. 57–80, July 2004.

[23] D. K. Roy and A. P. Pentland, "Learning words from sights and sounds: a computational model," *Cognitive Science*, vol. 26, no. 1, pp. 113–146, 2002.

[24] M. R. Walter, S. Hemachandra, B. Homberg, S. Tellex, and S. Teller, "Learning semantic maps from natural language descriptions," in *Proceedings of Robotics: Science and Systems (RSS)*, Berlin, Germany, June 2013.