Contextual Awareness: Understanding Monologic Natural Language Instructions for Autonomous Robots

Jacob Arkin,[1] Matthew R. Walter,[2] Adrian Boteanu,[3]
Michael E. Napoli,[1] Harel Biggie,[1] Hadas Kress-Gazit,[3] and Thomas M. Howard[1]

***Abstract*— Today, there are many examples of humans and robots regularly interacting in a variety of domains, such as manufacturing, coordinated assembly, and rehabilitation. A resulting demand for more generally accessible communication interfaces has motivated several recent independent research efforts focused on providing robotic systems with a robust natural language interface. Natural language interfaces enable intuitive interaction for untrained and non-expert users. However, achieving real-time performance is particularly challenging, yet essential, to enable flexible, efficient communication. The length of the language input directly impacts the run-time performance and quickly becomes a practical issue when the input is a sequence of multiple sentences, or a monologue. In this work, we propose a variant of a contemporary probabilistic graphical model for language understanding that introduces novel segmentation of the input into a sequence of sentences to be labeled in order. We introduce the notion of a continuously updated prior context that retains the meaning of previous sentences as the inference process proceeds. This prior context serves as evidence during future sentence evaluations. We evaluate our model on two natural language corpora, and demonstrate its utility on a Clearpath Husky A200 mobile manipulator and a simulated Rethink Robotics Baxter Robot.**

I. Introduction

Interaction between humans and robots has become an integrated aspect of many daily tasks in manufacturing, assistive rehabilitation, and agriculture, among others. Traditionally, this human-robot interaction has mostly occurred through a joystick or complex graphical interface. Often, learning how to safely and effectively use these systems requires training and experience, which is both costly and difficult to scale. As humans and robots continue to interact more frequently, it is important to develop interfaces that are efficient and intuitive.

Natural language interfaces for human-robot interaction are a more generally accessible way for people to communicate with and work alongside robots. In addition to being intuitive, natural language also provides a rich and highly flexible means of communication, allowing for both low-level (e.g. joint-level control) and high-level (e.g. implicit task sequence) instructions.

Natural language understanding for robot instructions can be formulated as a problem of associating a free-form utterance with a set of semantic symbols, an instance of the "symbol grounding" [7] problem. Early approaches to symbol grounding for human-robot language understanding typically treated language as a flat structure with deterministic semantic associations. While effective for simple instructions, this representation is difficult to generalize and does not capture the meaning of more complex language structures, such as nested clauses (e.g., "Pick up the blue block near the bin on the right side of the table"). More recently, notable progress has been made by posing the problem as inference on a graphical model, demonstrating successful grounding of instructions for robotic platforms such as autonomous forklifts, UAVs, and autonomous wheelchairs [8, 12, 24].

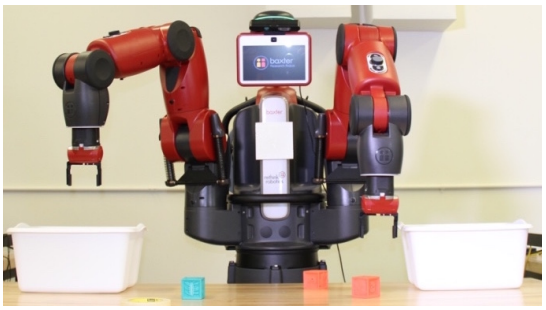

Fig. 1. An illustration of the Rethink Robotics Baxter Research Robot in an environment where it might receive the monologue command "Pick up the blue cube. Drop it in the right box. Pick up the red cube. Drop it in the left box."

A limitation of current approaches is the run-time cost of inference, an important performance metric for realizing fluid communication. Multiple factors can impact how quickly models will interpret an instruction. Recent work has focused on reducing the size of the search space. For example, the Hierarchical Distributed Correspondence Graph (HDCG) [9] learns a set of rules from language to exclude highly unlikely symbols from the search space. An instruction such as "pick up the block" would inform a set of rules excluding all object symbols that are not associated with the word 'block'. Another example, the Adaptive Distributed Correspondence Graph model (ADCG) [20] efficiently learns about groups of environment constituents (e.g. "the row of blocks") and their associated properties (e.g. "the middle block of the row of blocks") by dynamically instantiating the space of groups as conditioned on the expression of environment constituents.

Much less effort has focused on addressing the challenges of interpreting long (e.g. multi-sentence) instructions, where the length significantly contributes to the computational cost of inference. Models like the Distributed Correspondence Graph (DCG) [11] factorize the input across phrases (see

[1]Electrical and Computer Engineering, University of Rochester, Rochester, NY 14623, USA
[2]Toyota Technological Institute at Chicago, Chicago, IL 60637, USA
[3]Sibley School of Mechanical and Aerospace Engineering, Cornell University, Ithaca, NY 14853, USA

Figure II-B), representing each phrase as a set of factors in the graph. Consequently, the number of factor evaluations in the graph depends directly on the number of phrases. This becomes particularly relevant when the input is a monologue. The DCG represents monologues as a single parse tree with an implicit root phrase to connect the individual sentences. Therefore, the full monologue must be evaluated before producing a solution. Additionally, some features of language, such as anaphora, are difficult to label without context from other phrases in the instruction. While there exist linguistic approaches for resolving anaphora [15, 19], our approach maintains the environmental context when resolving meaning.

In this work, we present the Monologic Distributed Correspondence Graph (MDCG), which segments monologues into sequences of sentences to more efficiently assign meaning to individual instructions. The model treats sentences as individual parse trees labeled in order of occurrence. We maintain the context of previously inferred groundings from prior sentences for use when evaluating subsequent factors. In this way, the semantic labels for anaphora and other context-dependent linguistic features can be resolved directly. The model also reduces the total number of factor evaluations by eliminating the implicit phrases that the DCG would need to represent the monologue as a single parse tree. Perhaps most importantly, it allows the robot to begin to execute the command once the first sentence is labeled, rather than wait for a fully labeled tree. A faster response to the instruction should help achieve a more fluid interaction.

We evaluate our model using two corpora consisting of various simulation-based robot behaviors paired with natural language commands collected using the Amazon Mechanical Turk crowd-source platform. We also demonstrate the method on a Clearpath Husky A200 mobile manipulator and a simulated Rethink Robotics Baxter Research Robot.

II. Grounding Natural Language Instructions

A. Problem Description

The problem of interpreting language-based instructions can be formulated as the task of assigning semantic meaning to language input according to the context of the robot's world representation and the space of possible robot behaviors. Probabilistic approaches treat this as an inference problem for which the goal is to find the most likely robot behavior ($x^*(t)$) as conditioned on both the natural language instruction ($\mathbf{\Lambda}$) and the robot's representation of the relevant physical environment ($\mathbf{\Upsilon}$), shown in Equation 1.

$$x^*(t) = \underset{x(t) \in \mathbf{x}(t)}{\arg\max} \; p(x(t)|\mathbf{\Lambda}, \mathbf{\Upsilon}) \tag{1}$$

With this representation, the environment is typically composed of relevant information regarding environmental elements, such as the pose of objects and their associated semantics (e.g., color and type). The input instruction is a set of linguistic constituents ($\mathbf{\Lambda} = \{\lambda_1, \lambda_2, ..., \lambda_n\}$) obtained from a parse of the text, where each constituent is an individual language phrase. $\mathbf{x}(t)$ represents the set of possible robot behaviors, $\mathbf{x}(t) = \{x_1(t), x_2(t), ..., x_m(t)\}$, where a behavior is a trajectory. Therefore, $x^*(t)$ is the optimal robot behavior given the stated conditions. This problem is challenging for a couple of reasons. First, it is not obvious how to model any of the individual variables. For example, the required richness of the environment model likely depends on nature of the instructions. Second, the space of each of the variables is potentially massive (e.g. unstructured language inputs, continuous robot action space, cluttered environments), leading to high computational costs.

B. Technical Approach

Several modern approaches pose this problem as search over a factor graph. Tellex et al. [24] proposed Generalized Grounding Graphs (G^3) that represent language understanding with a factor graph and treat the labeling of each linguistic constituent as conditionally independent given labels of any child phrase. More recently, Howard et al. proposed the Distributed Correspondence Graph (DCG) [11], which introduces the assumption that the expressions of grounding constituents for individual phrases are conditionally independent. This allows the model to factorize the distribution over these groundings constituents. Correspondence variables (ϕ_{ij}) represent the association between the i^{th} phrase (λ_i) and the j^{th} grounding constituent of the i^{th} phrase (γ_{ij}). The DCG structures inference as search over unknown correspondences for factors with known linguistic and grounding constituent random variables. Individual phrases must also consider the set of correspondences produced by any child phrases ($\mathbf{\Phi}_{c_i}$). Additionally, the model represents the space of robot actions as robot motion constraints for a planner, as opposed to sampled state-action space trajectories, in order to significantly reduce the size of the grounding space. Because the number of expressible constraints is finite, the space of groundings is therefore bounded, which makes search over unknown correspondence variables possible without sampling. The full DCG is shown in Equation 2.

$$\Phi^* = \underset{\phi_{ij} \in \mathbf{\Phi}}{\operatorname{argmax}} \prod_{i=1}^{|\mathcal{N}|} \prod_{j=1}^{|\Gamma|} p(\phi_{ij}|\gamma_{ij}, \lambda_i, \mathbf{\Phi}_{c_i}, \mathbf{\Upsilon}) \tag{2}$$

C. Monologues of Robot Instructions

When using language to communicate with robots, it is desirable to allow people to use multiple uninterrupted sentences, or a monologue. However, we observed that models like DCG do not produce efficient structures when representing monologues. Consider the example monologue, "Pick the blue cube up. Drop it in the right box". By reasoning jointly over the two sentences at their root phrases, it is possible to infer the meaning of this instruction using the constituency parse structure. But, there are two interesting implications of this structure that have exploitable characteristics.

First, the second sentence in the monologue contains an anaphora (i.e., "it"), and interpreting its meaning requires the previous sentence as context. If the factors for the second sentence were aware of the grounding context of the first sentence, the semantic label for the anaphora "it" could

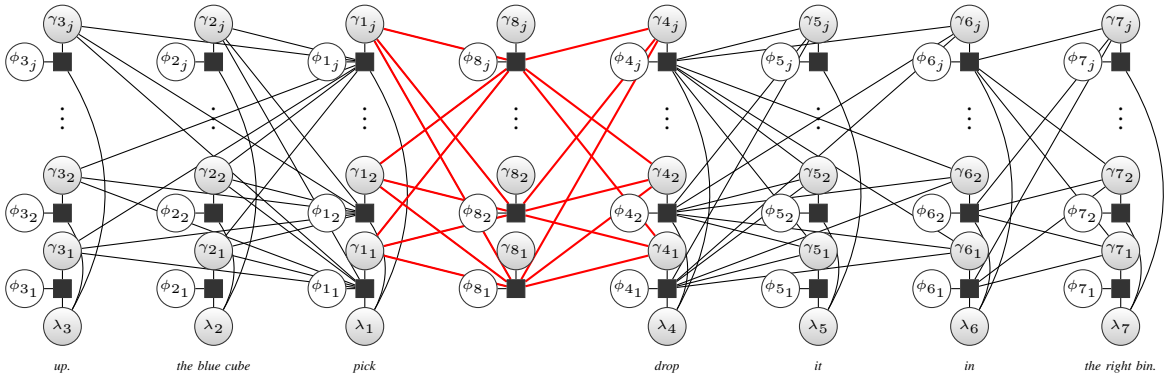

Fig. 2. An illustration of the Distributed Correspondence Graph model for natural language understanding of the expression "Pick the blue cube up. Drop it in the right bin." Note that inference goes from left to right for the first sentence and right to left for the second sentence, before meeting up at the center to interpret the meaning of the full utterance.

be resolved immediately. It is important to note that there exist linguistic solutions for resolving anaphora [1, 19], but our approach allows for direct environmental consideration during this resolution.

Second, it is not necessary to infer the meaning of both sentences in order to begin following the command. Instead, it is possible to interpret and execute the first sentence while concurrently grounding the meaning of the second sentence. However, this isn't possible with the DCG due to its inherent structure and the need to reason over the full monologue. More generally, while it is possible to treat sentences individually with the DCG model, it is not obvious how to determine that a user has finished expressing the current instruction. In a case where the model has grounded a sentence, but the user then provides an appending sentence to the monologue, the DCG would need to regenerate the full monologic parse tree and perform inference in its entirety.

These observations are jointly exploitable. By introducing a mechanism that provides factor evaluations for the second sentence with contextual awareness of the first sentence, we can eliminate the implicit root phrase that connects the two sentences. Additionally, an improved model of the meaning of the monologue allows incremental grounding. In the following section, we propose the Monologic Distributed Correspondence Graph (MDCG) that introduces novel extensions to the DCG in order to represent the accumulated context of the monologue input. In doing so, we eliminate the necessity of implicit language phrases and therefore introduce the potential for a more efficient and fluid interaction. We also introduce a hierarchical version of MDCG, called HMDCG, as a monologue extension to the HDCG.

D. Monologic Distributed Correspondence Graphs

We begin by describing the modification to the graph structure that is viable as a consequence of providing appropriate contextual awareness.

The DCG treats monologues similarly to single sentence instructions in that $\mathbf{\Lambda}$ does not change to reflect the notion of sentences. In order to represent sentences in our graph structure, we instead define the language input as a sequence of sentences, where each sentence is composed of phrases ($\mathbf{\Lambda} = [\lambda_{11}, , ..., \lambda_{1|N_1|}, \lambda_{21}, ..., \lambda_{n|N_n|}]$). In this new notation, λ_{ij} is the j^{th} phrase of the i^{th} sentence, and the root phrase of the i^{th} sentence can be represented as $\lambda_{i|N_i|}$, where $|N_i|$ is the number of phrases in the i^{th} sentence. We update the notation for the other variables as well. γ_{ijk} refers to the k^{th} grounding constituent for the j^{th} phrase of the i^{th} sentence. Similarly, ϕ_{ijk} refers to the mapping between the j^{th} phrase of the i^{th} sentence and the k^{th} grounding constituent for the j^{th} phrase of the i^{th} sentence. Finally, $\mathbf{\Phi}_{c_{ij}}$ refers to the set of child correspondences of the j^{th} phrase of the i^{th} sentence.

At this point, the graphical structure no longer has the required connections to resolve context-dependent phrases. It is necessary to provide the factor evaluations for current and future sentences with the appropriate contextual evidence from sentences that have already been grounded. We introduce a random variable that represents the contextually relevant set of correspondences from prior sentences, shown in equation 3 and 4 where $G(\cdot)$ is the accumulation function.

$$\mathbf{\Phi}_i^C = G(\mathbf{\Phi}_{(i-1)}^C, \mathbf{\Phi}_{(i-1)|N_{i-1}|}) \tag{3}$$

$$\mathbf{\Phi}_i^C = G(G(\cdots(G(\mathbf{\Phi}_1^C, \mathbf{\Phi}_{1|N_1|}), \cdots), \mathbf{\Phi}_{(i-1)|N_{i-1}|}) \tag{4}$$

Consider the context for each sentence of the monologic instruction, "Pick up the red block. Nevermind. Pick up the blue block." The first sentence has an empty set for the context variable. The context for the second sentence is the inferred action from the previous sentence, namely to pick up the red block. Given this evidence, the inferred groundings for the second sentence would be to not perform the first action. If the accumulation of prior context ($G(\cdot)$) was simpy treated as a union operation, the context for the final sentence would therefore be composed of the union of labels that mean to both pick up the red block and not pick up the red block. This is not likely to be useful because of the contradictory nature of the evidence. Instead, it makes more sense to have empty context for the final sentence; this implies $G(\cdot)$ should merge the information in a way that is not commutative.

Given these variable definitions, the Monologic Distributed Correspondence Graph model is describable and can

be seen in Equation 5. Note that $|\Gamma_{ij}|$ is the total number of grounding constituents for the j^{th} phrase of the i^{th} sentence, and K is the likelihood evaluation for all factors.

$$K = \prod_{i=1}^{|S|} \prod_{j=1}^{|N_i|} \prod_{k=1}^{|\Gamma_{ij}|} p(\phi_{ijk}|\gamma_{ijk}, \lambda_{ij}, \mathbf{\Phi}_{c_{ij}}, \mathbf{\Phi}_i^C, \mathbf{\Upsilon}) \tag{5a}$$

$$\Phi^* = \underset{\phi_{ijk} \in \mathbf{\Phi}}{\operatorname{argmax}} K \tag{5b}$$

Each factor in this evaluation is represented by a log-linear model using binary features; the goal is to evaluate the likelihood of a correspondence variable given a known language input and grounding consitituent, shown in Equation 6. Feature weights are learned offline during training by maximizing the likelihood of correspondences via the Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm [26]. The model is trained on an aligned corpus of parse-trees of user example instructions collected from Amazon Mechanical Turk [4] and the associated per-phrase ground truth labels assigned by experts. These corpora are typically specific to and representative of the desired domain (e.g. navigation vs. manipulation commands).

$$H = \prod_{i=1}^{|S|} \prod_{j=1}^{|N_i|} \prod_{k=1}^{|\Gamma_{ij}|} \Psi(\phi_{ijk}, \gamma_{ijk}, \lambda_{ij}, \mathbf{\Phi}_{c_{ij}}, \mathbf{\Phi}_i^C, \mathbf{\Upsilon}) \tag{6a}$$

$$\Phi^* = \underset{\phi_{ijk} \in \mathbf{\Phi}}{\operatorname{argmax}} H \tag{6b}$$

$$\Psi(\phi_{ijk}, \gamma_{ijk}, \lambda_{ij}, \mathbf{\Phi}_{c_{ij}}, \mathbf{\Phi}_i^C, \mathbf{\Upsilon}) = \frac{\exp(\sum_l \mu_l f_l(\phi_{ijk}, \gamma_{ijk}, \lambda_{ij}, \mathbf{\Phi}_{c_{ij}}, \mathbf{\Phi}_i^C, \mathbf{\Upsilon}))}{\sum_q \exp(\sum_l \mu_l f_l(\phi_{ijk}|\gamma_{ijk}, \lambda_{ij}, \mathbf{\Phi}_{c_{ij}}, \mathbf{\Phi}_i^C, \mathbf{\Upsilon}))} \tag{7}$$

As mentioned in Section I, Hierarchical Distributed Correspondence Graphs (HDCG) are a variant of DCG that improves the scalability of search as the complexity of the environment increases. The HDCG treats the space of groundings for natural language as a function of the utterance. This model introduces a set of rules ($P \in \mathbf{P}$), where each rule in P ($\rho \in P$) allows the admittance of a certain subset of grounding constituents. In this way, the space of possible grounding symbols is a function of the rules ($\mathbf{\Gamma}(P)$) and is therefore dynamically defined. DCG is initially evaluated in the space of rules $\mathbf{P}$, resulting in an inferred distribution of rule correspondences (Φ_ρ) conditioned on the utterance and the robot's environment model. The DCG is then evaluated using the distribution of reduced grounding spaces ($\mathbf{\Gamma}(P)$) to infer the most likely set of correspondences (Φ^*) for the most likely set of groundings (Γ^*). The first pass of the DCG is represented as a latent random variable in Equation 2, which can be marginalized as shown in Equation 8.

$$\Phi^* = \underset{\phi_{ij} \in \mathbf{\Phi}}{\operatorname{argmax}} \sum_{\Phi_\rho \in \mathbf{\Phi}_\rho} \prod_{i=1}^{|N|} \prod_{j=1}^{|\Gamma_i|} p(\phi_{ij}|\gamma(P)_{ij}, \lambda_i, \mathbf{\Phi}_{c_i}, \mathbf{\Upsilon}) \times p(\Phi_\rho|P, \mathbf{\Lambda}, \Phi_{\rho_c}, P_c, \mathbf{\Upsilon}) \tag{8}$$

The monologic modifications made to DCG are easily incorporated in HDCG because it is essentially a two-pass DCG model. Since the space of rules dictates the possible expressed groundings, we need to ensure the rules are conditioned on the context variables. We call this model Hierarchical Monologic Distributed Correspondence Graphs (HMDCG).

$$\mathcal{P} = p(\phi_{ijk}|\gamma(P)_{ijk}, \lambda_{ij}, \mathbf{\Phi}_{c_{ij}}, \mathbf{\Phi}_i^C, \Upsilon) \tag{9}$$

$$\Phi^* = \underset{\phi_{ijk} \in \mathbf{\Phi}}{\operatorname{argmax}} \sum_{\Phi_\rho \in \mathbf{\Phi}_\rho} \prod_{i=1}^{|S|} \prod_{j=1}^{|N_i|} \prod_{k=1}^{|\Gamma_{ij}|} \mathcal{P} \times p(\Phi_\rho|P, \mathbf{\Lambda}, \Phi_{\rho_c}, P_c, \mathbf{\Phi}_i^C, \mathbf{\Upsilon}) \tag{10}$$

III. Results

In this section we describe two applications of the MDCG/HMDCG for understanding multi-sentence natural language instructions on a pair of different platforms and symbolic representations. In both, the most appropriate evaluation metrics are accuracy (the ability to accurately reproduce the symbols of an instruction) and runtime (the time required to run probabilistic inference). All tests were run on a 2.2GHz Intel Core i7 processor with 16 GB of DDR3 RAM running Mac OS X 10.11 (x86-64) architecture.

A. Monologue Comprehension of Tactical Behavior Specification

The first evaluation considers the problem of instructions for the intelligence architecture that Barber et al. [2] describe. The symbolic language for this architecture is called the Tactical Behavior Specification (TBS), which describes actions, spatial relations, and objects that a robot must reason over in order to successfully complete a given task. Each TBS is composed of fields that include actions, modes, goals, goal constraints, and motion constraints. Spatial relations, such as "front", "left", or "right" describe regions in reference to some object described by an object type (e.g., "cone", "building", "person") and object color (e.g., "red", "white", "blue"). In this experiment, we trained a model composed of 49 annotated natural language instructions that produces 13,332 rule grounding training examples and $1,523,172$ symbol grounding training examples for factors of HMDCG. Using a beam search width of 2, both the HDCG and HMDCG were able to accurately reproduce all 49 examples with average and maximum inference times of 0.087 seconds and 0.813 seconds respectively.

1) Model Evaluation: The run-time performance of the second sentence of six monologue instructions is seen below in Table I. Run-times are separated for parsing and inference for each experiment. For comparison, we also ran the parse and inference times for each sentence and each context individually of each sentence in the monologue. The results of these are illustrated in Table II.

The results provide two interesting observations. First, the run-times associated with several of the simpler instructions

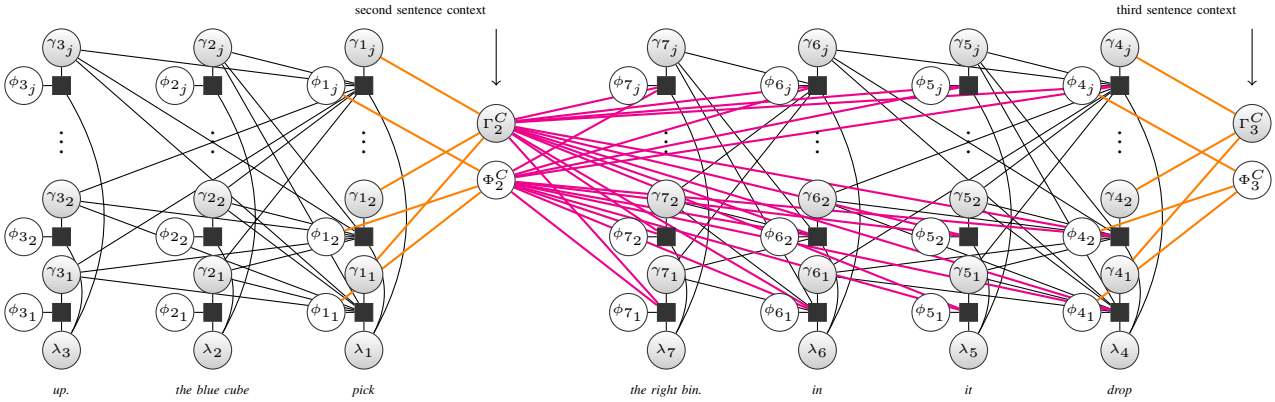

Fig. 3. The MDCG model.

sentence	context	run-time (sec) parse	inference
"walk to the white barrel."	"walk to the red cone."	0.001	0.019
"go near the blue building'.'	"walk to the red cone.", "walk to the white barrel."	0.001	0.043
"stay near the white barrel."	"walk to the cone."	0.001	0.093
"stay near the building."	"walk quickly to the barrel."	0.001	0.030
"keep behind the white person."	"walk to the door."	0.001	0.085
"navigate to the car that is near the building."	"keep behind the barrel."	0.003	0.635

TABLE I

PARSE AND MODEL INFERENCE PERFORMANCE OF THE MONOLOGUE INSTRUCTIONS.

sentence	run-time (sec) parse	inference
'walk to the red cone."	0.001	0.009
"walk to the white barrel."	0.001	0.008
"walk to the cone."	0.001	0.007
"walk quickly to the barrel."	0.001	0.014
"walk to the door."	0.001	0.007
"keep behind the barrel."	0.001	0.012
"go near the blue building."	0.001	0.011
"stay near the white barrel."	0.001	0.037
"stay near the building."	0.001	0.013
"keep behind the white person."	0.001	0.037
"navigate to the car that is near the building."	0.002	0.098

TABLE II

PARSE AND MODEL INFERENCE PERFORMANCE OF THE CONTEXT OF THE MONOLOGUE INSTRUCTIONS.

are approximately equal to the run-times for the two individual models without accounting for the additional layer that would be required to ground the meaning of the two sequential instructions. Second, the run-time of the last, more complex instruction is much less for the monologue model, which means that the interleaving of the two sentences is much more expensive in this example than the others. To further investigate this, we also ran the inference times for each multi-sentence parse tree for each monologue instruction. The results of these are illustrated in Table III.

In each case, the run-time of the monologue model is less than that of the single parse tree representation. In addition to having fewer phrases for evaluation, this is expected for two other reasons. First, the HDCG filters out symbols top-down, in that if a symbol might appear at the root of the sentence from the utterance or the context, it should be permitted for the entire graphical model. This slows down the search for both sentences (instead of just the second sentence), since each now has a larger symbolic representation. Second, we have already inferred a distribution (or maximum likelihood value) for the utterances that make up the context, so the size of the monologue HDCG is inherently smaller.

B. Monologue Comprehension of Linear Temporal Logic Groundings

Previous work extends the DCG model, the V-DCG model [3], by expanding the symbolic representation to include Linear Temporal Logic (LTL) symbols. LTL is a modal temporal logical formalism that is defined using Boolean prepositions and operators over an infinite discrete time series. In robotics, the GR(1) fragment of LTL has been used to define task specifications [21]. GR(1) LTL specifications describe the robot interacting in its environment as an assume-guarantee set of formulae, and can be synthesized into correct-by-construction controllers that guarantee the robot's behavior under the assumptions [13]. Conversely, if synthesis is unsuccessful, it can reveal problems in the task specification [23]. The V-DCG model produces groundings that contain both physical groundings and LTL formulae that are then synthesized into grounded controllers.

The LTL fomulae grounded from language are inferred using a symbol hierarchy refined from the original V-DCG symbol hierarchy. *Object* symbols are extended with two new symbols representing individual attributes (i.e. *type* and *color*), allowing for partially-defined sub-phrases to

sentence	run-time (sec)
“walk to the white barrel. walk to the red cone.”	0.029
“go near the blue building. walk to the red cone. walk to the white barrel.”	0.098
“stay near the white barrel. walk to the cone.”	0.170
“stay near the building. walk quickly to the barrel.”	0.077
“keep behind the white person. walk to the door.”	0.175
“navigate to the car that is near the building. keep behind the barrel.”	0.805

TABLE III

MODEL INFERENCE PERFORMANCE OF THE CONTEXT OF THE MONOLOGUE INSTRUCTIONS. SINCE OUR PARSER DOES NOT SUPPORT MULTI-SENTENCE PARSES AT THIS TIME, THE PARSE TREES FOR THIS COMPARISON WERE GIVEN TO THE MODEL.

contribute information that is disambiguated at a higher level in the tree. We also use *Spatial Relation* symbols such as *near*, *left* and *right*, as well as *Region* symbols associated with the locative prepositions “in” and “into,” which are designed to allow better spatial representations of relative object location. Lastly, the definition of an LTL scope symbol has been expanded to allow multiple sensor prepositions within a given scope.

We use two corpora of natural language instructions collected using crowdsourcing [3] for two different tasks using a simulated Baxter Research Robot. The first is a binary sorting task: cubes of two different colors are sorted into their respective bins (Figure 1). The second is a block stacking task: cubes of three different colors are rearranged to produce different stacks of blocks. The combined dataset consists of 22 monologue instructions composed of 60 separate sentences where each monologue was collected as a single response from a user. The combined corpus produces 1,132,710 symbol grounding training examples for MDCG. Using a beam width of 2, two MDCGs trained separately on one of the two corpora (sorting or stacking) were able to accurately and fully infer 19/21 individual stacking sentences and 28/39 individual sorting sentences; this exactly matches the accuracy results produced by the baseline V-DCG (no context). We also trained a single MDCG model for all 22 monologues that fully reproduced 17 of the stacking task sentences and 25 of the sorting task sentences, again matching the performance of the baseline V-DCG model. The results in Tables IV and V show that the run-time of the monologue-based model can be distributed across multiple queries.

Since our model interprets a monologue in a sequential manner, the inferred LTL formulae from prior sentences are carried over as new sentences are interpreted. Finally, a single specification is created and synthesized using the same method as shown by Boteanu et al. [3].

IV. PHYSICAL EXPERIMENT

The HMDCG was demonstrated using a Clearpath Robotics Husky A200 mobile manipulator (Figure 4). The groundings from the HMDCG produced goal states that were processed by the navigation system.

The software architecture used in the system demonstration included three processes: a speech-to-text process, a natural language understanding process using the HMDCG model, and a robot navigation process to control the robot’s behavior based on the inferred TBS sequence. In this experiment, natural language instructions that referenced one of four objects (represented by traffic cones) using assigned semantic object labels (“building”, “cone”, “traffic barrel”, and “car”) were executed by the mobile robot. The natural language process maintained the context of the previous utterances until navigation goals were reached, at which point the completed action symbol was removed from context. For example, if the sequence was ”go to the cone”, ”go to the building”, and ”go to the car” the system would eliminate the symbol for ”go to the cone” once it successfully navigated to that location. This enabled the robot to execute a variety of different behaviors using instructions that were provided by the human operator at different times. The traffic cones were positioned ahead, behind, left and right of the robot’s origin. Each label corresponded to a known goal configuration in the vicinity of its representative object that was received by the navigation system.

Fig. 4. An illustration of the experimental setup for the modified Clearpath Robotics Husky A200 mobile robot. Each of the four cones represented an object with the semantic tags “Building”, “Traffic Cone”, “Traffic Barrel”, and “Car”.

The navigation system consisted of both a motion planner and path following controller module. For path planning, the robot utilized a state lattice motion planner with a control set composed of three forward actions in a three-dimensional state space of position and heading with a known world model [22]. Each control set edge consisted of a parameterized action generated using a model predictive trajectory generator [10]. Completion of the goal state was sent to the HMDCG module.

V. RELATED WORK

Providing robots with the ability to understand spoken instructions has been a goal of general research interest for

sentence	monologue	order	run-time (sec)
“pick up the blue cube with your right hand and drop it in the right bin.”	1	1	0.696
“pick up the rightmost red cube and drop it in the left bin.”	1	2	0.483
“pick up the remaining red cube and drop it in the left bin.”	1	3	0.473
“pick up the blue cube with your right hand and pick up the red cube with your left hand.”	2	1	0.380
“put the blue cube in the right bin and put the red cube in the left bin.”	2	2	0.378
“pick the blue cube up.”	3	1	0.124
“drop it in the right box.”	3	2	0.144
“pick the red cube up.”	3	3	0.127
“drop it in the left box.”	3	4	0.150

TABLE IV

MODEL INFERENCE PERFORMANCE OF THE CONTEXT OF THE LTL GROUNDING MONOLOGUE INSTRUCTIONS.

sentence	monologue	run-time (sec)
“pick up the blue cube with your right hand and drop it in the right bin. pick up the rightmost red cube and drop it in the left bin. pick up the remaining red cube and drop it in the left bin.”	1	1.899
“pick up the blue cube with your right hand and pick up the red cube with your left hand. put the blue cube in the right bin and put the red cube in the left bin.”	2	0.825
“pick the blue cube up. drop it in the right box. pick the red cube up. drop it in the left box.”	3	0.765

TABLE V

MODEL INFERENCE PERFORMANCE OF THE CONTEXT OF THE LTL GROUNDING MULTI-SENTENCE MONOLOGUE INSTRUCTIONS.

some time. Earlier approaches tended to focus on parsing natural language into a formal structure [6, 14, 17]. Typically, a set of fixed control algorithms would handle the execution of actions from a limited, pre-specified set. The main limitation of these approaches is that, while they can handle limited variation in language input, they are not robust to the diverse possibility of input that non-expert users might provide.

More recent research efforts have focused on training probabilistic models to learn the associations between language and actions or paths. These models learn from examples collected from human users. One example, Vogel and Jurafsky [25], utilizes reinforcement learning theory to optimize a behavior policy that closely follows a referenced route. Addressing a similar domain, Matuscek et al [18] learned to translate from natural language to a map that is automatically labeled; this approach takes advantage of physical constraints imposed by the map, providing some robustness to uncertainty in the language and map. A main difference between these approaches and the work presented in this paper is the produced output; the related works produce actions, policies, maps, or controllers whereas our approach generates a description of the problem that can then be solved by planning algorithms or automatic controller synthesizers. While some of the discussed systems would be able to demonstrate accurate grounding of monologic route instructions, our approach is able to do so while maintaining the advantages of DCG.

Some promise can be found in closely related research that pose the problem of understanding robot instructions as inference over a probabilistic graphical model, similar to the work presented in this paper. Our approach builds on the ideas presented by Tellex et al [24] and Howard et al [11]. While these models utilize a similar algorithm, we modify the structure of the graph to incorporate the notion of sentences and prior context; this eliminates the need for factor evaluations connecting a given monologue's sentences' root-phrases and allows for chunked action execution, thereby improving the run-time performance while maintaining total training recall.

In this paper, we address ambiguities of natural language that naturally arise in monologues. Text-based methods have been extensively studied to solve problems such as anaphora and coreference resolution [15, 19]. Dependency parsing can further identify potential causal relations between sentences in monologues [1]. However, as early research has indicated, these problems require knowledge about the referenced entities in order to be solved [5]. Additionally, context has been shown to be critical in resolving anaphoras [16]. In the case of situated robot interaction, the physical environment provides the context of interpretation. Our monologue grounding method resolves anaphoras by leveraging groundings established by preceding instructions.

VI. CONCLUSION AND FUTURE WORK

We have presented a novel probabilistic model that addresses the problem of understanding natural language monologues of robot instructions. The model exploits correspondence and grounding variables that represent the meaning of previous instructions when interpreting the intent of the current utterance. This model permits algorithms that are inherently more efficient than is possible with existing models because it does not require that we repeatedly perform probabilistic inference on components of the model that we have previously explored, and the MDCG can be run in parallel with the speech recognition process for the next utterance without the need to backtrack. We compared our model against the state-of-the-art DCG and HDCG models

and showed that the MDCG and HMDCG enable better computational efficiency while preserving accuracy.

We also demonstrated the MDCG/HMDCG on a Clearpath Husky A200 mobile manipulator that was able to interpret and execute multiple instructions. As part of future work, we intend to further demonstrate the MDCG/HMDCG on a Rethink Robotics Baxter Research Robot for both the bin sorting and block stacking tasks. In addition to future user studies that explore the benefits of expressing instructions through short monologues, we are interested in incorporating dependency parsing into our framework to better model the meaning of complex instructions and bi-directional communication (dialogue). We also seek to develop a joint model of context and action, which will enable the model to infer the impact of a robot's behavior or an environment's dynamics on the content of the context over time.

VII. Acknowledgements

This work was supported in part by the Robotics Consortium of the U.S Army Research Laboratory under the Collaborative Technology Alliance Program and by the National Science Foundation under Grants IIS-1427547 and IIS-1427030

References

[1] Yoav Artzi, Nicholas FitzGerald, and Luke Zettlemoyer. Semantic parsing with combinatory categorial grammars. In *Proc. Assoc. for Computational Linguistics (ACL)*, 2013.

[2] Daniel J Barber, Thomas M Howard, and Matthew R Walter. A multimodal interface for real-time soldier-robot teaming. In *SPIE Defense+ Security*, pages 98370M–98370M. Int'l Society for Optics and Photonics, 2016.

[3] Adrian Boteanu, Thomas M. Howard, Jacob Arkin, and Hadas Kress-Gazit. A Model for Verifiable Grounding and Execution of Complex Natural Language Instructions. In *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2016.

[4] Michael Buhrmester, Tracy Kwang, and Samuel D Gosling. Amazon's mechanical turk a new source of inexpensive, yet high-quality, data? *Perspectives on psychological science*, 6(1):3–5, 2011.

[5] Jaime G Carbonell and Ralf D Brown. Anaphora resolution: a multi-strategy approach. In *Proc. Conf. on Computational linguistics*, pages 96–101. Association for Computational Linguistics, 1988.

[6] Juraj Dzifcak, Matthias Scheutz, Chitta Baral, and Paul 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 *Proc. IEEE Int'l Conf. on Robotics and Automation (ICRA)*, pages 4163–4168. IEEE, 2009.

[7] Stevan Harnad. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1):335–346, 1990.

[8] Sachithra Hemachandra, Felix Duvallet, Thomas M. Howard, Nicholas Roy, Anthony Stentz, and Matthew R. Walter. Learning Models for Following Natural Language Directions in Unknown Environments. In *Proc. IEEE Int'l Conf. on Robotics and Automation (ICRA)*, Seattle, WA, May 2015.

[9] Thomas M Howard, Istvan Chung, Oron Propp, Matthew R Walter, and Nicholas Roy. Efficient natural language interfaces for assistive robots. In *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS) Work. on Rehabilitation and Assistive Robotics*, 2014.

[10] Thomas M Howard and Alonzo Kelly. Optimal rough terrain trajectory generation for wheeled mobile robots. *Int'l J. of Robotics Research*, 26(2):141–166, 2007.

[11] Thomas M Howard, Stefanie Tellex, and Nicholas Roy. A natural language planner interface for mobile manipulators. In *Proc. IEEE Int'l Conf. on Robotics and Automation (ICRA)*, pages 6652–6659. IEEE, 2014.

[12] Albert S Huang, Stefanie Tellex, Abraham Bachrach, Thomas Kollar, Deb Roy, and Nicholas Roy. Natural language command of an autonomous micro-air vehicle. In *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems (IROS)*, pages 2663–2669. IEEE, 2010.

[13] Hadas Kress-Gazit, Georgios E Fainekos, and George J Pappas. Temporal-logic-based reactive mission and motion planning. *Trans. on Robotics*, 25(6):1370–1381, 2009.

[14] Geert-Jan M Kruijff, Hendrik Zender, Patric Jensfelt, and Henrik I Christensen. Situated dialogue and spatial organization: What, where... and why. *Int'l J. of Advanced Robotic Systems*, 4(2):125–138, 2007.

[15] Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. Stanford's multi-pass sieve coreference resolution system at the conll-2011 shared task. In *Proc. Conf. on Computational Natural Language Learning: Shared Task*, pages 28–34, 2011.

[16] Kenton Lee, Yoav Artzi, Jesse Dodge, and Luke Zettlemoyer. Context-dependent semantic parsing for time expressions. In *Proc. Assoc. for Computational Linguistics (ACL)*, 2014.

[17] M. MacMahon, B. Stankiewicz, and B. Kuipers. Walk the talk: Connecting language, knowledge, and action in route instructions. In *Proc. Nat'l Conf. on Artificial Intelligence (AAAI)*, 2006.

[18] Cynthia Matuszek, Dieter Fox, and Karl Koscher. Following directions using statistical machine translation. In *Proc. ACM/IEEE Int'l. Conf. on Human-Robot Interaction (HRI)*, pages 251–258. IEEE Press, 2010.

[19] Ruslan Mitkov. *Anaphora resolution*. Routledge, 2014.

[20] Rohan Paul, Jacob Arkin, Nicholas Roy, and Thomas M. Howard. Efficient Grounding of Abstract Spatial Concepts for Natural Language Interaction with Robot Manipulators. In *Proc. Robotics: Science and Systems (RSS)*, 2016.

[21] Nir Piterman, Amir Pnueli, and Yaniv Saar. Synthesis of reactive (1) designs. In *Verification, Model Checking, and Abstract Interpretation*, pages 364–380. Springer, 2006.

[22] Mikhail Pivtoraiko, Ross Alan Knepper, and Alonzo Kelly . Differentially constrained mobile robot motion planning in state lattices. *J. of Field Robotics*, 26(3):308–333, March 2009.

[23] Vasumathi Raman and Hadas Kress-Gazit. Explaining impossible high-level robot behaviors. *Trans. on Robotics*, 29(1):94–104, 2013.

[24] Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R Walter, Ashis Gopal Banerjee, Seth J Teller, and Nicholas Roy. Understanding Natural Language Commands for Robotic Navigation and Mobile Manipulation. In *Proc. Nat'l Conf. on Artificial Intelligence (AAAI)*, 2011.

[25] Adam Vogel and Dan Jurafsky. Learning to follow navigational directions. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 806–814. Association for Computational Linguistics, 2010.

[26] Ciyou Zhu, Richard H Byrd, Peihuang Lu, and Jorge Nocedal. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Trans. on Mathematical Software (TOMS)*, 23(4):550–560, 1997.