Towards Real-Time Natural Language Corrections for Assistive Robots

Alexander Broad*[§], Jacob Arkin[†], Nathan Ratliff[‡], Thomas Howard[†] and Brenna Argall*[§]
*Northwestern University, Evanston, IL, 60208
Email: alex.broad@u.northwestern.edu, brenna.argall@northwestern.edu
[†]University of Rochester, Rochester, NY, 14627
Email: jarkin@ur.rochester.edu, thomas.howard@rochester.edu
[‡]Lula Robotics Inc., Seattle, WA, 98102
Email: nathan.ratliff@lularobotics.com
[§]Rehabilitation Institute of Chicago, Chicago, IL, 60611

***Abstract*—We propose a generalizable natural language interface that allows users to provide corrective instructions to an assistive robotic manipulator in real-time. Allowing human operators to modify properties of how their robotic counterpart achieves a goal on-the-fly increases the utility of the system by incorporating the strengths of the human partner (e.g., visual acuity and environmental knowledge). Our natural language interface is based on a probabilistic graphical model, specifically a Distributed Correspondence Graph (DCG), that is employed to assign semantic meaning to user utterances in the context of the robots environment. We then use the desired corrections to alter the behavior of the robotic manipulator by treating the modifications as constraints on the motion generation (planning) paradigm. In this paper, we highlight four dimensions along which a user may wish to correct the behavior of his or her assistive manipulator. We develop our language model using data collected from Amazon Mechanical Turk in hopes of capturing a comprehensive selection of terminology that real people use to describe desired corrections. To demonstrate the efficacy of our approach, we run a pilot study on hardware with users unfamiliar with robotic systems and analyze points of failure and future directions.**

I. Introduction

As the desire for, and prevalence of, household robots continues to grow, the ability of humans to communicate particular goals and actions to their robot counterparts will be an integral facet in their successful adoption. Similarly, it will be important for users to be able to *customize* properties of how a robot achieves a desired goal and quickly interfere amid safety concerns. In particular, we consider that the automation may not have all pertinent environmental information (e.g., whether it is safe to move near another object or whether the object currently in the arm's grasp is stable) when planning and therefore may be insufficient for robotic solutions that promise long-term utility. To address these problems, we propose a natural language interface that allows users to interject *corrections* to the trajectory of a robotic manipulator in real-time. This approach balances the strengths of a human user (i.e., perception and general situational awareness) with the strengths of the robot (i.e., control) to create a highly adaptive and collaborative system. It also allows for online customization to human preferences.

Natural language is an exciting modality for human-robot communication for a number of reasons. One such reason is that it provides an intuitive bridge between non-expert users and their robot partners, allowing users to modify the behavior of a robot without previous knowledge of the system or programming experience. Additionally, as natural language is a very expressive form of communication, users can provide corrections with any level of detail or fidelity that they choose. Natural language is an especially important modality in assistive and rehabilitation robotics where a user's ability to physically interface with a robot is frequently limited by injury or disability.

Within the domain of assistive and rehabilitation robotics, there are a wide range of patients who retain control of their vocal muscles but have difficulty using standard human-robot interfaces that require a high level of fine motor control (e.g., joysticks and/or buttons). Therefore, providing an alternative, yet natural, means by which a user can interact with their assistive robot is of great utility.

A primary challenge in building a language-based communication system for human-robot partners is the translation from unstructured natural language to an actionable representation in the robot's world model. To be flexible and robust, the robotic system must be able to comprehend both qualitative and quantitative aspects of a user's utterance. However, achieving this goal from first principals is a significant challenge due to the complexity of natural language. The problem of how to best map linguistic elements to their corresponding symbolic representations in the real world is known as the *symbol grounding problem* ([7]). Solving this problem is important in practice because it allows a robot to interpret how a spoken phrase relates to the current environment and therefore improve its internal representation of the world.

Our solution is to limit the size and scope of the language that the robot must understand to the space of *corrections*. We begin with a robotic system able to generate a preliminary trajectory to achieve a desired goal. The initial assumption is that the user does not have a strong preference in how the goal is achieved. We then allow the user to provide natural language commands to correct the behavior as they see fit.

Therefore in our approach, the robotic system is not required to comprehend the full granularity of an initial utterance. Instead, we focus on allowing the user to provide corrections that *modify essential attributes* of the robot's behavior such as the speed, orientation of the end effector, or spatial constraints throughout the trajectory. This model allows us to interpret natural language instructions as *constraints* on the initial trajectory that can be used to update the trajectory on-line.

To date, natural-language *corrections* have seen limited use within robotics. Moreover, our motivating and target application domain is assistive and rehabilitation robotics, which at least clinically *does not yet leverage* natural-language interfaces for operating assistive machines that physically move in the world—in spite of the fact that such interfaces could be incredibly useful for particular patient populations.

This work aims to aid users of assistive robots by allowing end-users *without robotics expertise* and *with severe motor impairments* to use free-form natural language to adapt, on the fly, the autonomous behavior of their robotic aids—in response to environment dynamics or novel scenarios, and also to best fit their personal preferences. The approach moreover is general to a broad range of collaborative human-robot domains, in addition to rehabilitation and assistance.

We begin with related work in Section II followed by a detailed presentation of the scope of our problem in Section III. We then describe our approach in Section IV and our implementation in Section V. Finally, we evaluate our work in Section VI and conclude in Section VII.

II. Related Work

This section overviews related literature in the areas of speech-based interfaces for assistive robots, corrections provided to robot systems and natural language understanding.

A. Assistive Robots

The value of speech as an input modality for persons with severe motor impairments to operate assistive machines has been leveraged for use with robotic arms [5], robotic wheelchairs [13] and mobile manipulators [20]. The human typically uses speech to provide high-level goals or teleoperation commands for the autonomy. For example, Volosyak et al. [20] present an autonomous system called FRIEND-II, an assistive mobile manipulator that actively queries the human for task goals or execution assistance, and through speech the user provides high-level (e.g., "pour a drink") and low-level (e.g., "gripper up") instruction.

B. Corrections from Humans

Pose corrections from a human also are provided to assistive robot manipulators, though typically through interfaces other than speech. For example, in the KARES-II system, the user indicates the target object with their gaze and visual-servoing automation brings the object near user's head, who then fine-tunes position through a shoulder motion interface [4]. Pose corrections—translational or rotational—are provided as static

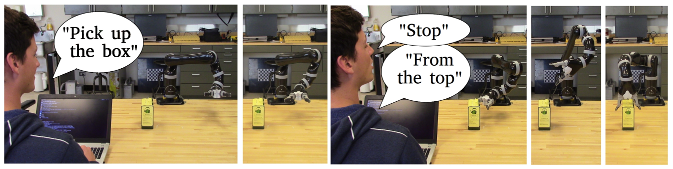

Fig. 1. Example workflow. From left to right : (1) User issues initial command. (2) Arm begins to move and user recognizes undesirable action. (3) User stops arm and provides desired correction to motion. (4) Arm begins moving again, this time with new planning parameters. (5) Arm completes the task with user correction applied.

step amounts to the MANUS manipulator on the FRIEND-II system, via EEG selections from a graphical display [19]. Our approach tackles a broader scope of corrective instruction than only direct pose corrections, for example spatial relations (e.g "go over the box") between the robot and the environment.

Outside of the field of assistive robotics, previous methods that provide explicit corrections largely rely on tactile sensors that measure physical interactions. Schmid et al. [16] use small nudges on a robot's end-effector to define high-level goals and trajectory modifications. Similarly, Argall et al. [2] record finger swipes that translates into iterative changes in the orientation or position of their iCub robot. (See Argall and Billard [1] for a survey of touch-based corrections.)

Implicit corrections to robotic manipulators also are provided as rewards within Reinforcement Learning paradigms—where the robot essentially is told the (positive or negative) value of a state-action pair, without being instructed explicitly on what to do instead. For example, within the field of Learning from Demonstration there are works that first seed an initial policy with human demonstrations, and then update that policy based on rewards (implicit corrections) accumulated with experience [11].

C. Natural Language Understanding

There is a wealth of related work in developing algorithms that improve a robot's ability to understand natural language. Specifically, there is much interest in using natural language as a method of specifying task goals for mobile robots and robotic manipulators. For example, the system presented by MacMahon et al. [12] parses natural language into a set of high-level route instructions that represent knowledge about spatial actions and layouts. Similarly, Matuszek et al. [14] learn a natural language-to-action model based on example pairs of commands and their corresponding control expressions. These works focus on developing high level plans that can be fed to a local planning method, rather than on instantaneous low-level corrections to motion trajectories.

In closely related work, Tellex et al. [18] describe a probabilistic graphical model (the Generalized Grounding Graph, or G^3) that defines a factor graph connecting natural language constituents to groundings in the environment. Howard et al. [10] define an alternative graphical model formulation (the Distributed Correspondence Graph, or DCG) that assumes conditional independence of all linguistic phrases and conditional independence of grounding elements within a grounding, which reduces the search space and increases the efficiency of the learning and inference computations. Both of these approaches demonstrate their effectiveness in using language as a method of communicating high level goals.

Our work builds on previous language-based interfaces for robotic manipulators by allowing users to impart preferences on *how* the robot achieves the desired goal by modifying the robot's trajectory during run-time. One major difference is that previous work places the burden of dealing with a dynamic environment on the robot. Our approach instead leverages the human's perceptual and situational awareness abilities to provide a robust solution in unpredictable environments, in addition to being responsive to the user's personal preferences.

III. Problem Definition

Our principal aim is to develop robots that can assist people with everyday tasks. These machines are particularly important for users with limited motor control as they can restore lost ability and a sense of personal freedom. We know that end-users within the rehabilitation domain overwhelmingly prefer to retain as much control as possible, however we also know that many assistive machines are inaccessible to those with severe motor impairments—precisely because of the control complexity in operating these machines, and limitations in the control interfaces available to these patients. There is an opportunity for robotics autonomy to offload a portion of the control burden, and to introduce natural language as an interface for the collaborative operation of assistive robots.

In order for collaborative assistive robots to be a reality, we need to develop natural interfaces and simple methods for users to control not only *what* their robot does, but *how* it does it. Namely, as human-robot teams work together in shared spaces, users need to be able to augment pertinent characteristics of the behavior of their robotic counterpart.

However, there is the supplementary problem of how to map a spoken command to the robot's world model when considering language as a method of human-robot communication. Robotic manipulators in particular require significant computation to appropriately control, due to the number of variables that must be specified. Moreover, unless the user is willing to define the full and exact trajectory in joint space, it is likely that there are many, if not an infinite, number of ways to ground the language in the environment.

To solve this problem, our approach builds a model that is both effective at grounding utterances in the environment and robot control, and also is efficient even for non-experts. While previous research has focused on using natural language as a method for providing high-level planning goals to robotic partners, our approach uses language as a means for providing instantaneous corrections to a robot's trajectory.

Through corrections, a human partner can impart relevant contextual information as well as personal preferences to their robotic counterpart. This may improve the robustness and acceptance of the joint human-robot system. By empowering the human to provide corrections, we aim to addresses the questions of how users can customize their robot's behavior to their specific needs, and how the robot system can provide robust operation for long-time assistance. An example of the work flow can be seen in Figure 1.

IV. Approach

Our natural language model is based on a grammar and set of features that specifically relate language constituents to corrective actions along the robot's trajectory. By using an initial trajectory with a known grounding as a starting point and focusing on the space of *corrections*, both the language understanding and motion generation problems become significantly more tractable from a computation standpoint.

A. Distributed Correspondence Graph

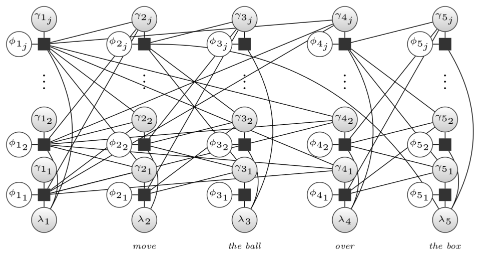

Fig. 2. A DCG used to infer natural language corrections from the utterance "move the ball over the box" The factors f_{i_j}, groundings γ_{i_j}, and correspondence variables ϕ_{i_j} are functions of the symbols used to represent natural language corrections.

In order to understand natural language instructions, it is necessary to assign language to a meaningful set of concepts ("groundings") in the robot's world context, a task known as *symbol grounding*. These groundings can be perceived objects, regions in the world, paths for navigation, actions the robot can take, *et cetera*. For example, consider the instruction shown in Figure 2, "move the ball over the box". Both "the ball" and "the box" ground to objects in the world model, "over" grounds to a region above the box object, and "move" grounds to an intended manipulation action as constrained by the grounded region and objects. Presented more formally, the challenge of providing robots with the capacity to understand natural language can be described as finding the most likely ($\mathbf{x}^*(t)$) robotic behavior or action ($\mathbf{x}(t)$) given the context of an environment model (Υ) and a provided natural language utterance (Λ):

$$\mathbf{x}^*(t) = \arg\max_{\mathbf{x}(t)} p\left(\mathbf{x}\left(t\right)|\Lambda, \Upsilon\right) \tag{1}$$

In practice, this is a difficult maximization problem because of the unrestricted nature of both the language and the environment inputs. Some contemporary work has approached modeling this problem via factor graphs that dynamically predict the expressed groundings for linguistic constituents. In particular, the G^3 model [17] uses the compositional and hierarchical parse structure of language to generate a factor graph that connects language constituents (phrases: $\lambda_i \in \Lambda$)

to groundings (objects, locations, paths: $\gamma_i \in \Gamma$) and binary correspondence variables (true/false: $\phi_i \in \Phi$) that indicate a true or false association between a particular grounding and language phrase.

This approach provides an effective framework for understanding robotic instructions, but becomes increasingly computationally demanding as the complexity of the environment and/or the robotic platform increase(s) (e.g., an environment composed of sets of objects or a robotic manipulator with 6 degrees of freedom). In order to provide real–time functionality for a more diverse set of practical scenarios, it is necessary to improve the run–time efficiency of the model.

The model used in our work is an alternative graphical model formulation of the grounding problem, the Distributed Correspondence Graph [10]. Specifically addressing the issue of runtime performance, researchers proposed the Distributed Correspondence Graph (DCG), replacing search in the space of planning trajectories with search in the space of boundaries and preferences of behavior (e.g., regions instead of locations; constraints instead of sampled paths). This effectively changes the problem of inferring a solution from the continuum of robot actions to one of formulating a robot planning problem that can then be handed off to a dedicated planner. This change provides two specific advantages when dealing with the size of the search space. First, the space of planning constraints that can be understood is finite, bounded by the cartesian product of the objects in the environment and the considerable relationships between them. Second, the expression of groundings can be evaluated independently from one another. By assuming conditional independence of grounding constituents, the model can be factorized such that only the correspondence variables are unknown. We thus compute the most likely correspondence variables:

$$\mathbf{\Phi}^* = \underset{\phi_{ij} \in \Phi}{\arg\max} \prod_i \prod_j p\left(\phi_{ij} | \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon\right) \tag{2}$$

The DCG model has recently been adapted and applied to several collaborative human-robot scenarios [3, 6, 8, 9]. The model presented in this paper differs from these previous adaptations in several distinct ways. First, the language used to describe corrective updates differs from that what is used to describe initial commands, therefore subset of language that must be understood changes. Often, the instructions are ambiguous comparison utilizing implicit information. For example, given an initial instruction to, "Pick up the cup," the robot may then receive an updating instruction, "by the handle". In this case, the update is modifying the preferences for grasp locations, but the grasp object is implicit. To address this, we introduce the notion of prior context, consisting of the initially expressed instruction and its grounded root phrase. This prior context is used as additional information during the update of the planning constraints.

Secondly, while computational efficiency and real-time performance was a consideration in each of the other DCG adaptations, it is a necessity in this work due to the nature of the problem we are addressing. For example, again consider an initial instruction, "Pick up the cup"; the robot may initiate a trajectory to pick the cup up from the top. When provided with the update statement, "by the handle," the system must understand and incorporate the new information in real-time. Certainly, any execution of the update instruction much be initiated prior to the completion of the initial task.

Similar to other applications, our approach requires the incorporation of a grounding space that is sufficient for representing all relevant semantic information. In our particular implementation this is the space of natural language corrections. Given the real–time aspect of our system, it is also important that this grounding space be efficient in its representation; the run–time performance of the model is dependent on the size of the search space, which in turn depends on the space of groundings. We discuss the space of groundings in IV-A2.

1) Features and Training : The factor nodes in the graphical model are represented by log-linear models with binary features. Optimized feature weights ($w_k \in W$) are learned from an annotated corpus of examples via the Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) algorithm, as shown in Equations 3 and 4.

$$\mathbf{\Phi}^* = \underset{\phi_{ij} \in \Phi}{\arg\max} \prod_i \prod_j f\left(\phi_{ij}, \gamma_{ij}, \lambda_i, \Gamma_{c_{ij}}, \Upsilon\right) \tag{3}$$

$$f\left(\phi, \gamma, \lambda, \Gamma_c, \Upsilon\right) = \frac{e^{\sum_k w_k f_k(\phi,\gamma,\lambda,\Gamma_c,\Upsilon)}}{\sum_{\phi_m \in \Phi} e^{\sum_k w_k f_k(\phi_m,\gamma,\lambda,\Gamma_c,\Upsilon)}} \tag{4}$$

While the DCG model is capable of expressing features as defined by the environmental context (i.e., features for capturing spatial characteristics of the environment, such as the distance to nearest object), our implementation only considers features in the context of language and the presence of objects. Given the graphical model's hierarchical nature, and the dependency of a given factor on child node groundings, we also incorporate features for grounding inheritance.

2) Space of Groundings : The space of groundings is composed of six different symbols and can be broken into two categories as defined by the type of semantic information they express.

3) Environmental Groundings : Object, spatial relationship and region grounding types are all used in our model to refer to the environment.

Object groundings represent a physical object and have specific properties capturing model-relevant information about the object. Specifically, each object has an associated type (e.g., "ball" or "box"), pose in the world, list of type-specific landmarks, and a unique identifier to disambiguate between objects of the same type. The type-specific landmark information defines grasp locations in grasping tasks and path preferences for pick-and-place tasks.

Spatial relation and region groundings represent relative spatial information. *Spatial relations* groundings consist solely of a type; the noun phrase "the left" would ground to a spatial relation grounding of type "left". *Region* groundings combine spatial relation information with object groundings; so, the noun phrase "the ball on the left" would ground to a region composed of a spatial relation grounding of type "left" and an object grounding of type "ball".

Environmental groundings however are insufficient for representing the kind of higher-level robotic planning problem information expected from natural language instructions, so we also incorporate symbols meant to express constraining information about the planning problem.

4) Planner Groundings : Our model implementation uses constraint, cost, and goal type groundings to express relevant information about the behavior expected by the robot that will constrain the planning problem.

Constraint groundings consist of a type, value and an associated object grounding. The constraint type defines the kind of constraint that is being applied to the action (e.g., velocity, orientation), where the value is defined by the value property.[1] The associated object defines to which object the constraint should be applied. So, the verb phrase "slowly move the ball" would be expressed by "velocity," associated object "ball" and a low value that satisfies the notion of "slowly".

Cost groundings represent preferences for different paths. Each cost is composed of a type, an object and a landmark. In this case, the type can either be a "reward" or a "penalty". The object property defines the object associated with the cost, and the landmark indicates a preferred path relative to the object. For example, the instruction "move the ball over the box" would have an expressed cost grounding of type "reward," associated object "box," and a landmark to indicate the top of the box, thereby representing a preference for the path of motion to pass over the top of the box.

Goal groundings represent desired goal information. A goal is composed of a type (e.g., "grasp" or "place"), an associated object, and a landmark. The goal type defines the type of action, the object defines the object associated with the action, and the landmark defines a preferred location for the action. For example, the instruction "pick up the box by the top" would have an expressed goal grounding of type "grasp," associated object "box," and a landmark to indicate the top of the box, thereby representing a desired goal to pick up the box by the top.

5) Corrective Language: Our approach focuses specifically on grounding online corrections to robot trajectories. In this work, we examine the space of corrections for a robotic manipulator along four dimensions. They are:

- The *speed* with which the manipulator executes a trajectory (ℓ_1).
- The *orientation* of the manipulator's end effector (ℓ_2).
- The *position* of the end effector relative to other objects in the environment (ℓ_3).
- The gross contact point during *grasping* motions (ℓ_4).

[1]Currently these values are hand defined continuous values that can be interpreted relative to each other (e.g., faster or slower), but can not be set to any value the user would like.

Note that the first correction concerns the *rate* of the trajectory execution (ℓ_1), the second concerns *orientation* within the world frame (ℓ_2), the third concerns the location of *intermediary points* along the trajectory (ℓ_3), and the fourth concerns the location of the *final goal* of trajectory (ℓ_4).

B. Motion Planner: Riemannain Motion Optimization

The resulting corrections can be fed directly into many standard path planning approaches, especially optimization-based motion generators that provide convenient interfaces to shaping the robot's behavior through costs and constraints.

Our implementation makes use of Riemannian Motion Optimization (RieMO) [15], which provides both a flexible interface to arbitrary smooth constraints on the motion and an interface to shaping the geometry of the workspace around obstacles to bias the way in which the end-effector moves around them. We encode the grounded constituents discussed above into the motion using a combination of these two tools along with trajectory re-timing to modulate execution speeds.

V. Implementation

Here we provide details our end-to-end implementation on an assistive manipulator. We employ a MICO robotic arm, a 6 Degree of Freedom (DoF) robotic arm from Kinova Robotics with a 2-finger gripper. Our language model is based on an adaptation of the open-source Human to Structured Language (H2SL) software package[2] that allows us to convert free-form text input to its corresponding structured language grounding in the robot's environment.

Training data for the language model was gathered using Amazon Mechanical Turk (AMT), an online crowdsourcing platform. Using this platform, we present participants with pairs of videos that show the MICO robotic arm performing the same task with one significant modification to some characteristic of its trajectory. We then ask the subjects to describe the main difference between the two videos, using language that describes the variation as a *correction*.

In total 77 AMT participants contributed 280 labeled examples. We added to this dataset 24 examples gathered during an initial study with three members from our lab. We analyze the resulting responses to remove data that either wasn't appropriately worded as a correction or was repetitive of previous responses. This resulted in a total of 28 training examples. Our corpus was comprised of 100 phrases and an average of 3.57 phrases per example.

VI. Evaluation

We first analyze the language model in Section VI-A and then assess the full system with a pilot study in Section VI-B.

A. Language Model

We evaluate our language model along similar metrics to those proposed by Howard et al. [10]. First, we perform all possible combinations of leave-one-out cross-validation on our model. Using this approach, we correctly infer the constraint

[2]https://github.com/tmhoward/h2sl

in 16 of the 28 variations. In each case where the held-out example was incorrect, we find the error was due to a word existing in the example but not in our training set for that test. The only exception was one case where the correct grounding was expressed in addition to an erroneous margin constraint. Additionally, in cases where the model suggested the incorrect inference, it was never completely incorrect—that is, the correct semantic information was expressed for all leaf phrases (no children) with known words.

We also report the *average runtime* of correction inference using our DCG model. On our test data the system runs at an average of 0.0236 seconds per inference and on our training data the system runs at an average of 0.0253 seconds per inference. As we increase the scope of our corrections, we expect increased complexity of our model to increase as well.

B. Pilot Study

Here we report on a pilot study with novice users. The study consisted of three able-bodied participants recruited from the Rehabilitation Institute of Chicago. Each experiment was executed with a distinct set of initial conditions (starting location and environment). For each type of correction, we developed a predefined environment with a specific task that was described to the user in simple language (e.g., "move to the other side of the block"). Users triggered the correction by verbalizing "stop," and then described the intended correction using free-form language. If the correction provided by the user was successfully parsed and grounded in our model, it was then passed to the motion planning algorithm as a set of constraints, which were used to update the trajectory. If our model was unsuccessful in parsing or grounding the user's language, the planning parameters were not updated and the MICO would continue along its initial trajectory. For each type of correction, this procedure was repeated until either (1) the user provided language that was successfully parsed and grounded or (2) a maximum of 3 attempts. In between unsuccessful attempts we asked users to modify their phrasing to try and help the robot understand their correction. An example of this workflow can be seen in Figure 1.

We analyze the results by noting how far along the processing pipeline users were able to get in each trial. A successful trial involves (1) parsing the user input, (2) grounding the user input, (3) ensuring the grounding matches the desired correction, and (4) updating the motion planner with a new set of constraints based on the grounding.

The pilot study included a total of 21 full trials (3 users, 7 full trials each). Of the 21 full trials, 8 resulted in a failure to parse the user input, 3 were grounded incorrectly by our language model and in the remaining 10 trials, the user input was successfully parsed, grounded and applied to the executed trajectory. In all, there were 46 total attempts, of these 28 attempts resulted in a failure to parse the user input, 3 resulted in a failure to ground to a correction, 5 were grounded incorrectly and 10 were full successes. We analyze these results by separating the failure cases from the successes. In particular, we examine each point of failure and reason about areas of improvement.

When an attempt resulted in a *failure to parse*, we can immediately say that this was due to a lack of data in the grammar we developed from the Amazon Mechanical Turk responses. This issue is easily improved through an expanded grammar, which can come from further data collection. The fact that a large number of failed attempts occurred at this level, and that not many occurred after this point, is encouraging as this is the simplest part of the system to improve. It also illustrates that there is likely a difference in the language that users choose when using the real hardware versus watching videos.

When a user input was successfully parsed but our language model *failed to ground* the utterance in the symbol space, it suggests that our model does not cover the full space of corrections and that we either need a more complete set of features or we need to include more training data. In fact, during analysis, we realized that all 3 attempts that ended in this scenario did so due to a single missing feature. Specifically, our model did not appropriately associate certain object landmarks with spatial relations. After the completion of the pilot study, we added this feature to our model and it now successfully grounds all 3 utterances.

When a user input was successfully parsed and grounded, but the there was a *mismatch* between the grounding and the desired correction, it could potentially point to the fact that there is a conflict between the user input and training data. However, our analysis shows that in all 5 attempts that ended in this scenario, the user provided ambiguous input that could easily be interpreted in the manor presented by our language model. For example, in one experiment the user was trying to correct the method by which the MICO picked up a teabox, the user provided the correction "move over box." Our model grounded this statement to a cost symbol representing the desire for the arm to move above the box, as opposed to the desired grounding of a goal symbol suggesting that the arm pick up the box from its top.

Lastly, in cases when the full system succeeded we asked the users to give us their impressions of how well the modified trajectory matched their desired correction and their general thoughts on the timing of the system. In all 10 trials that ended in success, users reported "high satisfaction" with both the modifications to the trajectory and the responsiveness.

VII. Conclusion

In this paper, we present a natural language interface that allows a human user to *correct* the behavior of a collaborative robot system at run-time. We discuss a pilot study used to gain insights into the viability of such an approach, leading to a future full evaluation with end-users suffering from a range of motor impairments. Our method improves the user experience while simultaneously improving the communication and execution of a user's desired trajectory, when compared to high-level goal driven approaches. We believe our collaborative approach will improve human-robot communication and aid in the adoption of assistive robotic technologies.

References

[1] Brenna D. Argall and Aude G. Billard. A Survey of Tactile Human-Robot Interactions. *Robotics and Autonomous Systems*, 58(10):1159 – 1176, 2010.

[2] Brenna D. Argall, Eric L. Sauser, and Aude G. Billard. Tactile Guidance for Policy Refinement and Reuse. In *IEEE 9th International Conference on Development and Learning (ICDL)*, pages 7–12. IEEE, 2010.

[3] Jacob Arkin and Thomas M Howard. Towards Learning Efficient Models for Natural Language Understanding of Quantifiable Spatial Relations. In *Proceedings of Robotics: Science and Systems (RSS)*, 2015.

[4] Zeungnam Bien, Myung-Jin Chung, Pyung-Hun Chang, Dong-Soo Kwon, Dae-Jin Kim, Jeong-Su Han, Jae-Hean Kim, Do-Hyung Kim, Hyung-Soon Park, Sang-Hoon Kang, Kyoobin Lee, and Soo-Chul Lim. Integration of a Rehabilitation Robotic System (KARES II) with Human-Friendly Man-Machine Interaction Units. *Autonomous Robots*, 16(2):165–191, 2004.

[5] Michel Busnel, Riadh Cammoun, Franoise Coulon-Lauture, Jean-Marie Détriché, Gérard Le Claire, and Bernard Lesigne. The robotized workstation "MASTER" for users with tetraplegia: Description and evaluation. *Journal of Rehabilitation Research and Development*, 36 (3):217–229, 1999.

[6] Felix Duvallet, Matthew R Walter, Thomas Howard, Sachithra Hemachandra, Jean Oh, Seth Teller, Nicholas Roy, and Anthony Stentz. Inferring maps and behaviors from natural language instructions. In *Experimental Robotics*, pages 373–388. Springer, 2016.

[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. *arXiv preprint arXiv:1503.05079*, 2015.

[9] Thomas M Howard, Istvan Chung, Oron Propp, Matthew R Walter, and Nicholas Roy. Efficient natural language interfaces for assistive robots. In *IEEE/RSJ International Conference on Intelligent Robots and System (IROS)*. IEEE, 2014.

[10] Thomas M Howard, Stefanie Tellex, and Nicholas Roy. A Natural Language Planner Interface for Mobile Manipulators. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6652–6659. IEEE, 2014.

[11] Petar Kormushev, Sylvain Calinon, and Darwin G Caldwell. Reinforcement Learning in Robotics: Applications and Real-World Challenges. *Robotics*, 2(3):122–148, 2013.

[12] Matt MacMahon, Brian Stankiewicz, and Benjamin Kuipers. Walk the Talk: Connecting Language, Knowledge, and Action in Route Instructions. *Def*, 2(6):4, 2006.

[13] Christian Mandel and Udo Frese. Comparison of wheelchair user interfaces for the paralysed: Head-joystick vs. verbal path selection from an iffered route-set. In *Proceedings of the European Conference on Mobile Robots*, 2007.

[14] Cynthia Matuszek, Evan Herbst, Luke Zettlemoyer, and Dieter Fox. Learning to Parse Natural Language Commands to a Robot Control System. In *Experimental Robotics*, pages 403–415. Springer, 2013.

[15] Nathan Ratliff, Marc Toussaint, and Stefan Schaal. Understanding the Geometry of Workspace Obstacles in Motion Optimization. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4202–4209. IEEE, 2015.

[16] Andreas J Schmid, Martin Hoffmann, and Heinz Wörn. A Tactile Language for Intuitive Human-Robot Communication. In *Proceedings of the International Conference on Multimodal Interfaces*, pages 58–65. ACM, 2007.

[17] Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R Walter, Ashis Gopal Banerjee, Seth Teller, and Nicholas Roy. Approaching the symbol grounding problem with probabilistic graphical models. *AI magazine*, 32(4):64–76, 2011.

[18] 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 *AAAI Conference on Artificial Intelligence, North America*. AAAI Publications, 2011.

[19] Diana Valbuena, Marco Cyriacks, Ola Friman, Ivan Volosyak, and Axel Graser. Brain-Computer Interface for High-Level Control of Rehabilitation Robotic Systems. In *Proceedings of the International Conference on Rehabilitation Robotics (ICORR)*, 2007.

[20] Ivan Volosyak, Oleg Ivlev, and Axel Gräser. Rehabilitation Robot FRIEND II - The General Concept and Current Implementation. In *Proceedings of the International Conference on Rehabilitation Robotics (ICORR)*, 2005.