Journal Title
XX(X):1–14
©The Author(s) 2015
Reprints and permission:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Real-Time Natural Language Corrections for Assistive Robotic Manipulators

Alexander Broad[1,4], Jacob Arkin[2], Nathan Ratliff[3], Thomas Howard[2] and Brenna Argall[1,4]

Abstract

We propose a generalizable natural language interface that allows users to provide corrective instructions to an assistive robotic manipulator in real-time. This work is motivated by the desire to improve collaboration between humans and robots in a home environment. 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). This work is applicable to users with and without disability. Our natural language interface is based on the Distributed Correspondence Graph, a probabilistic graphical model that assigns semantic meaning to user utterances in the context of the robot's environment and current behavior. 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 to capture a comprehensive sample of terminology that people use to describe desired corrections. We then develop an end-to-end system using open-source speech-to-text software and a Kinova Robotics MICO robotic arm. To demonstrate the efficacy of our approach, we run a pilot study with users unfamiliar with robotic systems and analyze points of failure and future directions.

Keywords

Human-Robot Interaction, Assistive Robotics, Natural Language Processing, Constrained Motion Planning

1 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. While recent work has shown robotic manipulators capable of human-scale household tasks such as unloading a dishwasher (Huang et al. (2015)) and folding laundry (Maitin-Shepard et al. (2010)), we must consider that simply initializing a robot with a high-level goal will be insufficient for robotic solutions that promise long-term utility. 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. 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). This population includes patients suffering from movement disorders, such as Parkinson's Disease or Multiple Sclerosis, and those with spinal cord injuries that affect the low-cervical nerves (C5-C8) or below, to name a few. Therefore, providing an alternative, yet natural, means by which a user can interact with their assistive robot is of great utility.

[1]Northwestern University, Evanston, IL, 60208
[2]University of Rochester, Rochester, NY, 14627
[3]Lula Robotics Inc., Seattle, WA, 98102
[4]Rehabilitation Institute of Chicago, Chicago, IL, 60611

Corresponding author:
Alexander Broad, Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60208.
Email: alex.broad@u.northwestern.edu

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* (Harnad (1990)). 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 to 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 of the robot 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.

By limiting the scope of language and actions that the robot must understand, the challenge of grounding natural language in the robot's world view becomes significantly more tractable. We achieve this goal by building a language understanding model that efficiently characterizes the space of corrections that a user may wish to apply. This model allows us to then interpret natural language instructions as *constraints* on the initial trajectory, which can be used to modify and update the trajectory during run-time.

To date, natural-language *corrections* have seen limited use within the field of 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 aides—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 a discussion of related work in Section 2 followed by a detailed presentation of the scope of our problem in Section 3. We then describe our approach in Section 4 and our implementation of an end-to-end system in Section 5. Finally, we evaluate our work in Section 6, discuss the results and future work in Section 7 and conclude in Section 8.

2 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.

2.1 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 (Busnel et al. 1999; Kim et al. 2009), robotic wheelchairs (Mandel and Frese 2007) and mobile manipulators (Volosyak et al. 2005). The human typically uses speech to provide high-level goals or teleoperation commands for the autonomy. For example, in Volosyak et al. (2005) the autonomy on the FRIEND-II assistive mobile manipulator 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. Our work builds on these ideas by allowing a user to interrupt the motion of an assistive robot and provide natural language corrections to improve the utility of the robot.

2.2 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 (Bien et al. 2004). Pose corrections—translational or rotational—are provided as static step amounts to the MANUS manipulator on the FRIEND-II system, via EEG selections from a graphical display (Luith et al. 2007; Valbuena et al. 2007). 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 objects in the environment. Moreover, in our paradigm corrections are provided verbally.

Outside of the field of assistive robotics, there are approaches that provide explicit corrections, and rely on tactile sensors that measure physical interactions. Schmid et al. (2007) use small nudges on a robot's end-effector to define high-level goals and trajectory modifications. Similarly, Argall et al. (2010) record finger swipes on a touchpad that translates into iterative changes in the orientation or position of their iCub robot. For a survey of touch-based corrections, see Argall and Billard (2010). Corrections also are provided through kinesthetic demonstration as described in Niekum et al. (2015) and graphical interfaces Argall et al. (2012).

[*]The reason is largely historical, and a reaction to experiences 30 years ago with commercial natural-language interfaces used to drive powered wheelchairs, that created dangerous situations due to a lacking robustness to variations in tone. Natural language systems have progressed greatly since that time however. Commercial natural-language systems moreover are already used extensively to operate non-mobile devices such as computers and light switches.

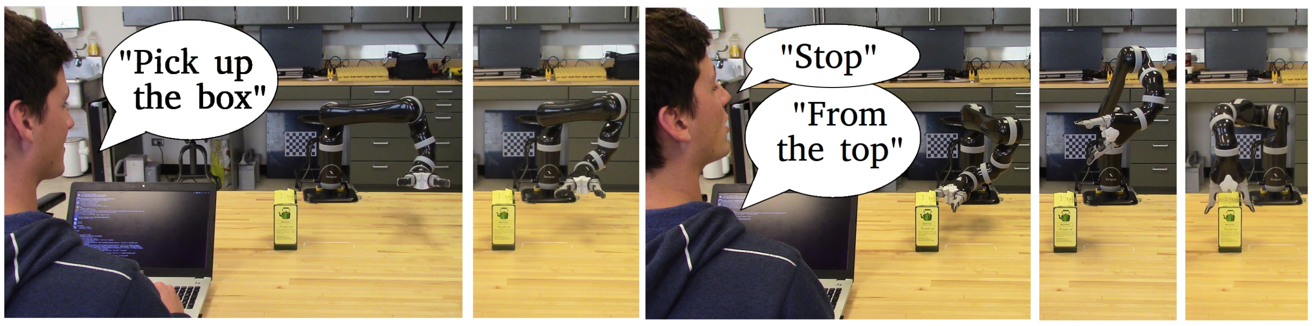

Figure 1. Example workflow. From left to right : (1) User issues initial command. (2) Arm begins to move and user recognizes undesirable action. (3) User stops arm and provides desired correction to motion. (4) Arm begins moving again, this time with new planning parameters. (5) Arm completes the task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

Implicit corrections to robotic manipulators 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 (Kormushev et al. (2013); Kober and Peters (2009)).

Corrections also can serve as a mechanism for the human to indicate their preference. A related topic is the area of learning user preferences over trajectories taken by robotic manipulators. For example, Abdo et al. (2015) use a collaborative filtering model to learn user preferences on how best to organize objects in their environment. Jain et al. (2015) learn user preferences by collecting a dataset of positive and negative examples which then defines a cost function used in a trajectory planning algorithm. Our work differs in its use of natural language as the means of communicating preference, and also in its focus on instantaneous corrections rather than long-term learning.

2.3 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 has been much recent work in using natural language to specify task goals for mobile robots and robotic manipulators. For example, the system presented in MacMahon et al. (2006) parses natural language into a set of high-level route instructions that represent knowledge about spatial actions and layouts. Similarly, Matuszek et al. (2013) learn a natural language-to-action model based on example pairs of commands and their corresponding control expressions. Hemachandra et al. (2015); Duvallet et al. (2016) describe approaches for following directions in partially observed environments, and Mei et al. (2016) develop a neural sequence-to-sequence model to translate natural language instructions to action sequences.

In related work, the Generalized Grounding Graph (G^3) Tellex et al. (2011b), a probabilsitic graphical model for natural language symbol grounding that assumes conditional independence of groundings across linguistic constituents, demonstrates effective inference of paths in a state-action space for controlling a robotic forklift. The Distributed Correspondence Graph (DCG) Howard et al. (2014b) extends this idea for efficient inference of motion planning constraints from natural language utterances by assuming conditional independence across both linguistic constituents and parts of the symbolic representation. This transforms the problem of finding the most likely grounding assuming a known (true) correspondence variable to inferring the most likely expression of correspondence variables in a known space of symbolic constituents. Applications of this model include inferring homotopic constraints for mobile robot motion planning(Yi et al. 2016) and Linear-Temporal Logic formulae for verifiable natural language interaction (Boteanu et al. 2016). Recent extensions of DCG include the Hierarchical Distributed Correspondence Graph (HDCG) (Howard et al. 2014a) and the Adaptive Distributed Correspondence Graph (ADCG) (Paul et al. 2016) which learn approximate representations of the full graphical model to achieve real-time performance. The HDCG is applied in Barber et al. (2016) for inferring commands to guide a mobile robot intelligence architecture.

Whereas these models focus on inferring trajecories or constraints that define a behavior, our approach focuses on modifying the robot's behavior at execution time in the context of robotic manipulators. Our approach leverages the human's perceptual and situation awareness abilities to complete partially specified behaviors and unforeseen environment dynamics.

3 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 independence. 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 and paralysis—precisely because of the control complexity in operating these machines, and limitations in the control interfaces available to these patients. There is an opportunity here for robotics autonomy to offload a portion of the control burden from the human, and to introduce natural language as an interface for collaborative operation of these 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 for a non-expert to use in real-time. 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 by which a user can provide instantaneous corrections over 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.

4 Approach

The cornerstone of our approach is a generalizable natural language understanding model that allows us to efficiently model the space of corrections, and importantly, perform inference in real-time. In operation, our approach first has the robot devise an initial trajectory for a given goal, and as that trajectory is executed, listen for corrections from the user. Inference is performed on any observed utterances, in order to interpret the phrase within the context of the robot's environment. This information is then used to apply the correction, instantaneously, within a motion planner.

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 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.

In this section we first explain our natural language model (Sec 4.1), then the space of corrections (Sec 4.1.1) and how these corrections influence the motion planning (Sec 4.2).

4.1 Grounding Natural Language Corrections

In order to understand natural language corrections, 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

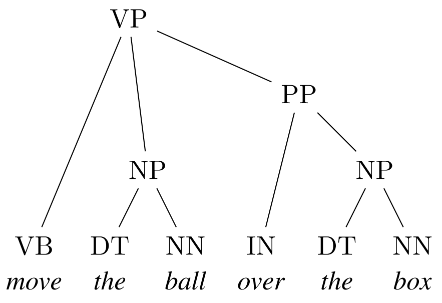

Figure 2. The parse tree for the robot instruction "move the ball over the box". Notice that in this parse, the prepositional phrase "over the box" attaches to the verb phrase at the root of the sentence instead of joining with the noun phrase "the ball." *Key:* verb phrase (VP), noun phrase (NP), prepositional phrase (PP), verb (VB), noun (NN), preposition (IN), determiner (DT).

instruction shown in both Figures 2 and 3, "move the ball over the box". Both "the ball" and "the box" ground to objects in the environment 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. We notice that the meaning of the utterance differs when interpreted as an instruction or a correction to the current behavior.

The environment model represents the physical workspace of the robot that defines geometric, appearance and pose information about constituent 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):

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

In practice, this is a difficult maximization problem because of the unrestricted nature of both the language and the environment inputs. The G^3 model (Tellex et al. (2011a)) uses the parse structure of the instruction to generate a factor graph that connects linguistic constituents (phrases: $\lambda_i \in \mathbf{\Lambda}$) to groundings (objects, locations, paths: $\gamma_i \in \mathbf{\Gamma}$) and binary correspondence variables (true/false: $\phi_i \in \mathbf{\Phi}$) which indicate a true or false association between a particular grounding and language phrase. The most likely grounding then is represented as:

$$\mathbf{\Gamma}^* = \underset{\gamma_i \in \mathbf{\Gamma}}{\arg\max}\; p\left(\mathbf{\Phi}|\mathbf{\Gamma}, \mathbf{\Lambda}, \Upsilon\right) \tag{2}$$

Both the language and correspondences are known ($\phi_i = TRUE$), and the model searches for the set of groundings that maximize the product of factors in the graph. By assuming conditional independence for phrases, given their child phrase groundings ($\mathbf{\Gamma_{c_i}}$), equation 2 can be factorized as shown in equation 3. Note that a child phrase refers to any phrase that hierarchically contributes to another given phrase; for example, the noun phrase "the box" in Figure 2 is considered a child of the prepositional phrase composed of the text "over". In this case, the factor evaluation for this prepositional phrase would incorporate the distribution of groundings expressed by its child phrases.

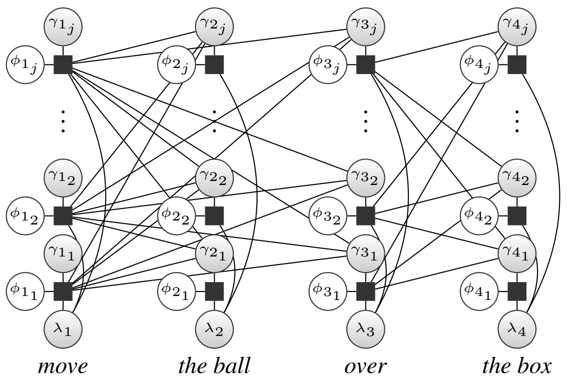

Figure 3. A DCG used to infer natural language corrections from the utterance “move the ball over the box”. The factors f_{i_j} (black squares) are functions of the groundings γ_{i_j}, language λ_i, correspondence variables ϕ_{i_j}, environment, and context used to represent natural language corrections. Known random variables are represented as gray circles, unknown random variables are reperesented as white circles, and the random variable for the environment, which connects to all factors, is not illustrated.

$$\mathbf{\Gamma}^* = \underset{\gamma_i \in \mathbf{\Gamma}}{\arg\max} \prod_{i=1}^{|\mathcal{N}|} p\left(\phi_i = TRUE | \gamma_i, \lambda_i, \mathbf{\Gamma_{c_i}}, \Upsilon\right) \quad (3)$$

This approach provides an effective framework for understanding robot instructions, but becomes increasingly computationally demanding as the symbolic representation grows. When the space of symbols is inclusive of sets in the space of sets (e.g. constraints, objects), search for the most likely set for each phrase is exponential with respect to the number of constituents.

The DCG (Howard et al. (2014b)) alternatively searches in the space of robot motion planning constraints. Probabilsitic inference is made linear in respect to the number of constituents by assuming conditional independence across constituents of each set for each phrase. This model assumes that the space of planning constraints that can be understood is known and finite, bounded by the cartesian product of the objects in the environment and the considerable relationships between them. 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. The model computes the most likely association between the ith phrase (λ_i) and the jth grounding constituent of the ith phrase (γ_{ij}), resulting in an assigned value for the correspondence between the ith phrase and the jth grounding constituent of the ith phrase (ϕ_{ij}) (Eqn. 4). Note that, similar to the G^3 model, each factor evaluation is conditioned on the inferred groundings of the child phrases ($\mathbf{\Gamma}_{c_i}$); because the DCG is searching for an unknown correspondence variable, it also necessary to incorporate the child phrase correspondences ($\mathbf{\Phi}_{c_i}$).

$$\mathbf{\Phi}^* = \underset{\phi_{ij} \in \Phi}{\arg\max} \prod_{i=1}^{|\mathcal{N}|} \prod_{j=1}^{|\mathcal{C}_i|} p\left(\phi_{ij} | \gamma_{ij}, \lambda_i, \mathbf{\Gamma}_{c_i}, \mathbf{\Phi}_{c_i}, \Upsilon\right) \quad (4)$$

The factor nodes in the graphical model are represented by log-linear models with weighted binary features, as shown in Equations 5 and 6. Feature weights ($w_k \in W$) are optimized from an annotated corpus of examples via the Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) algorithm.

$$\mathbf{\Phi}^* = \underset{\phi_{ij} \in \Phi}{\arg\max} \prod_{i=1}^{|\mathcal{N}|} \prod_{j=1}^{|\mathcal{C}_i|} f\left(\phi_{ij}, \gamma_{ij}, \lambda_i, \mathbf{\Gamma}_{c_i}, \mathbf{\Phi}_{c_i}, \Upsilon\right) \quad (5)$$

$$f\left(\phi, \gamma, \lambda, \mathbf{\Gamma_c}, \mathbf{\Phi_c}, \Upsilon\right) = \frac{e^{\sum_k w_k f_k(\phi, \gamma, \lambda, \mathbf{\Gamma_c}, \mathbf{\Phi_c}, \Upsilon)}}{\sum_{\phi_m \in \Phi} e^{\sum_k w_k f_k(\phi_m, \gamma, \lambda, \mathbf{\Gamma_c}, \mathbf{\Phi_c} \Upsilon)}} \quad (6)$$

We extend this model to consider corrections to actions by incorporating the notion of context that describes the current behavior or activity of the robot arm. It is often necessary to incorporate the prior context to resolve the meaning of a correction. For example, the correction “by the handle” may be incomprehensible without the symbols that describe the original instruction “pick up the cup”. This prior context is used as additional information during the update of the planning constraints. Our approach is based on the DCG because our symbolic representation for natural language corrections is composed of sets of constraints, regions, and objects that allow for conditional independence assumptions and we require real-time performance. The maximum likelihood estimate of the dsitribution of expressed groundings for the correction are merged with that of the prior context to determine the meaning of the utterance.

4.1.1 Space of Corrective Language In this work, we examine the space of corrections for a robotic manipulator along four dimensions. They include:

- 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).

We anchor these corrections to characteristics of the robot’s motion. 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 *subgoals* along the trajectory (ℓ_3). The final dimension concerns the location of the *final goal* of trajectory (ℓ_4). An example of each type of correction can be seen in Figure 4.

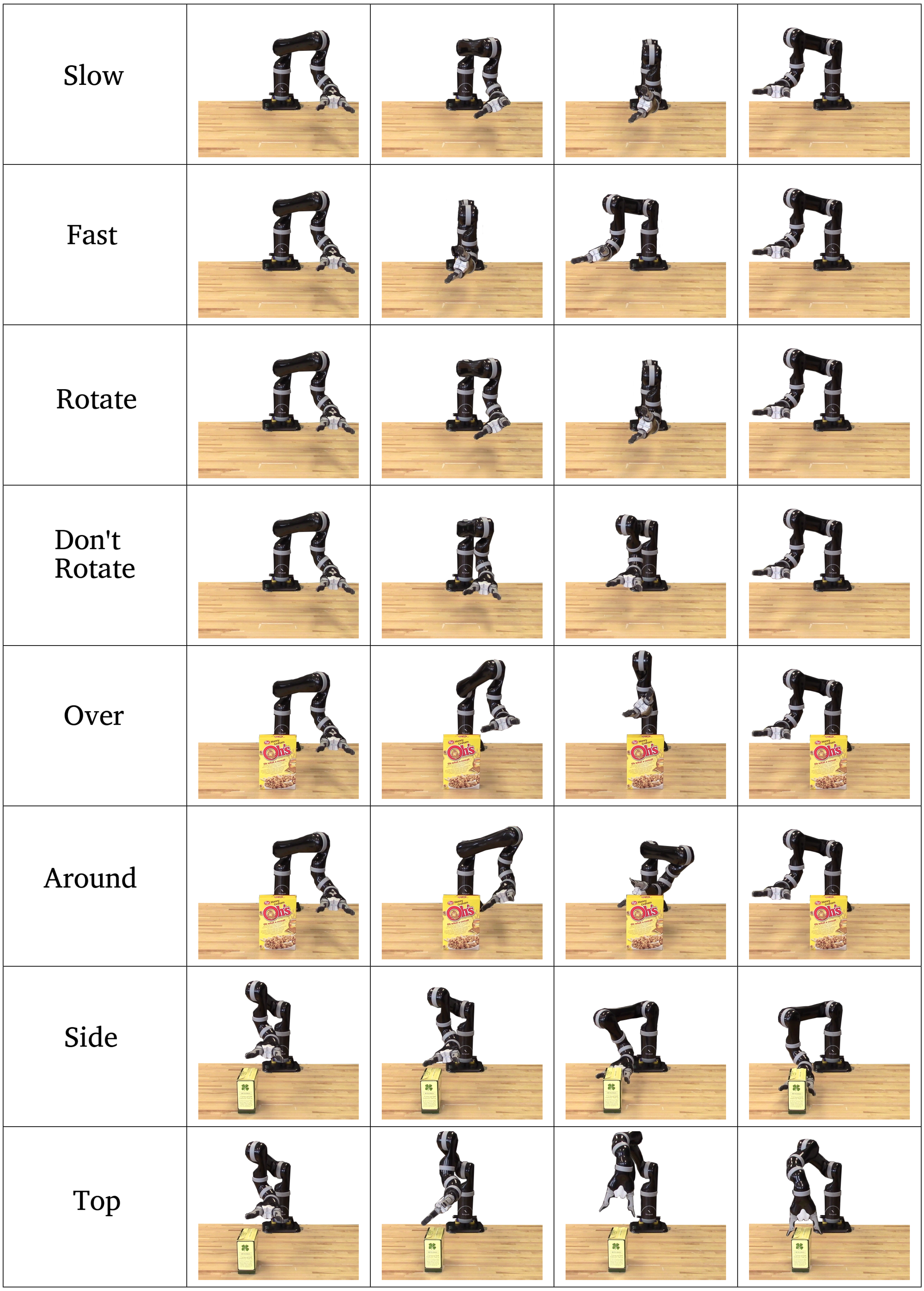

Figure 4. Example trajectory corrections. Rows 1 and 2: Correction type ℓ_1 (speed). Rows 3 and 4: Correction type ℓ_2 (orientation). Rows 5 and 6: Correction type ℓ_3 (position). Rows 7 and 8: Correction type ℓ_4 (grasp).

4.1.2 Symbolic Representation of Corrections We define a space of groundings that provides coverage for the space of natural language corrections. This symbolic representation is composed of six different symbols that can be decomposed into two categories as defined by the type of semantic information they express. The first category, environmental groundings, consists of three symbols: objects, spatial relations and regions. The second category, planner groundings, also consists of three symbols: constraints, costs and goals. Each is defined in more detail below.

Object, spatial relationship and region groundings are all symbols that 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 landmark information defines grasp locations in grasping tasks and path preferences for pick-and-place tasks. As an example, the noun phrase "the ball" would ground to an object grounding of type "ball" with known pose information (from the world model), a set of landmarks and an unknown unique identifier. In the current system, the specific landmarks are pre-defined based on the particular object. Spatial relation and region groundings represent relative spatial information. *Spatial relation* 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 information about the planning problem. Our model uses constraint, cost and goal type groundings to express relevant information which 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. The associated object defines to which object the constraint should be applied. So, the verb phrase "slowly move the ball" would be expressed by a constraint grounding of type "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.

4.2 Motion Planner: Riemannain Motion Optimization (RieMO)

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) (Ratliff et al. 2015), 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.

RieMO models motion optimization generically as a constrained optimization problem of the form

$$\min_{\xi} \sum_{t=1}^{T} \sum_{i} c_i^{(t)}(\boldsymbol{z}_i, \dot{\boldsymbol{z}}_i, \ddot{\boldsymbol{z}}_i) \quad \text{s.t.} \begin{cases} \boldsymbol{g}_i^{(t)}(\boldsymbol{z}_i, \dot{\boldsymbol{z}}_i, \ddot{\boldsymbol{z}}_i) \leq \boldsymbol{0} \\ \boldsymbol{h}_i^{(t)}(\boldsymbol{z}_i, \dot{\boldsymbol{z}}_i, \ddot{\boldsymbol{z}}_i) = \boldsymbol{0} \end{cases} \tag{7}$$

where $\xi = (\boldsymbol{q}^{(t)}, \dots, \boldsymbol{q}^{(t)})$ is a trajectory through the configuration space $\mathcal{C}$, and each $\boldsymbol{z}_i^{(t)} = \phi_i(\boldsymbol{q}^{(t)})$ denotes the configuration as represented in some task space $\mathcal{Z}_i$ defined by differentiable map $\phi_i : \mathcal{C} \to \mathcal{Z}_i$. $\boldsymbol{g}$ and $\boldsymbol{h}$ are differentiable functions defining inequality and equality modeling constraints on the system. Function g, for instance, includes joint limit constraints and obstacle collision constraints to ensure the motions are physically valid, while function h includes constraints for reaching the Cartesian target, achieving desired orientations at the end-effector, and shaping the approach behavior in the final half-second of the motion. The task spaces may be the space of the end-effector, various points on the body, or something more abstract like a higher-dimensional geometric space representing proximity to obstacles in the environment and the way those obstacles *warp* their surroundings (Ratliff et al. (2015)). This representation is general since any Riemannian geometry can be represented as a map of this form. The optimizer uses the Augmented Lagrangian algorithm to handle the more complicated constraints on the end-effector orientation and approach, along with barrier functions to handle the simpler inequality constraints that prevent obstacle and joint limit penetration.

Our work incorporates online corrections into the RieMO framework. Responding to grounded constituents commanding how to approach the object or how to move around the object amounts to modifying the constraints and workspace geometry of the problem and re-optimizing. Speeding up and slowing down the trajectory is implemented by scaling the time-constant of the trajectory. Each correction is described in detail next.

4.2.1 Position Correction We can control the behavior of the end-effector as it passes by the object (ℓ_3) by changing the way we represent the object's geometry. The obstacle itself is approximated as a spherical boundary region around the true object for simplicitly, and the workspace geometry around the sphere is represented using cylindrical coordinates. The orientation of the cylindrical coordinate system shapes how geodesics wrap around the object thereby biasing the motion in a semantically relevant way. Figure 5 shows examples of the wrapping behavior of the workspace geometry for various orientations of the cylindrical system. Shaping the workspace geometry in this way directly encodes our intuition for what we mean by passing "over" or "around" the object into the obstacle's surounding geodesic structure.

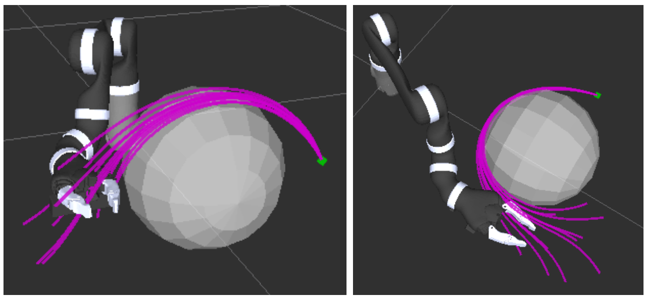
Figure 5. Sample trajectories (starting from a distribution of points around the initial end-effector location) following the natural gradient of an attractor to the target under a workspace geometry biasing the end-effector to go over (left) and around (right) the object.

Specifically, the cylindrical system is increasingly blended toward an ambient Euclidean geometry to transition smoothly to a Euclidean geometry away from the obstacle, following the Latent Euclideanization / Globalized Local Coordinates framework outlined in Ratliff et al. (2015). Let $\phi : \mathbb{R}^3 \to \mathbb{R}^3$ be a cylindrical coordinate map defined by $\phi_c(x, y, z) = (r, \theta, z)$ with $r = \sqrt{(x^2 + y^2)}$ and $\theta = \operatorname{atan2}(y, x)$. Assuming z points up, and defining $\alpha(\boldsymbol{x})$ to be a smooth proximity function that is zero away from the obstacle and approaches one close to the obstacle, then the geometric map we use to bias the system to go "around" the object is

$$\psi(\boldsymbol{x}) = \begin{bmatrix} \alpha(\boldsymbol{x})^{\frac{1}{2}} \mathbf{D}^{\frac{1}{2}} \phi(\boldsymbol{x}) \\ (1 - \alpha(\boldsymbol{x}))^{\frac{1}{2}} \boldsymbol{x} \end{bmatrix} \tag{8}$$

with $\boldsymbol{x} = (x, y, z) \in \mathbb{R}^3$ and $\mathbf{D} = \operatorname{diag}(10, 1, 1)$ (stretching the radial direction). This map becomes the identity map far from the obstacle (the lower block), but close to the obstacle it increasingly stretches the radial direction of the space orthogonal to the vertical axis so that distances toward the object measure much larger closer to the object (the upper block). Geodesics—shortest paths—through the resulting mapped space are naturally curves around the object, specifically around the vertical (z) axis of the workspace. Biasing the motion to go above the obstacle is implemented by *rotating the coordinate system* so that the cylindrical axis aligns with the forward pointing axis of the workspace.

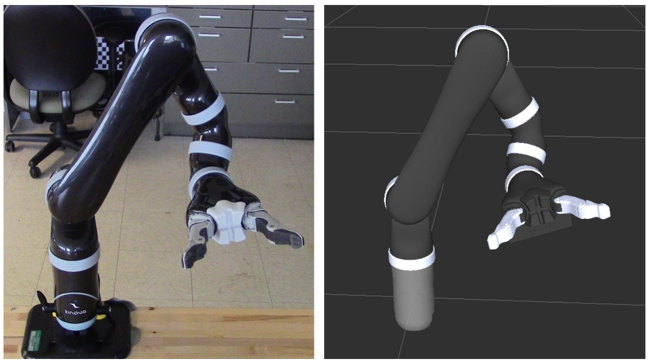
Figure 6. The MICO robotic arm. In real hardware (left) and simulation (right).

4.2.2 Orientation Correction To control the orientation of the end-effector during flight (ℓ_2), we add *equality constraints* to the end-effector's vertical axis (specifically, the x-axis) to remain parallel to the workspace's z-axis. During the approach to the obstacle we add additional shaping constraints as a function of the desired approach direction $\hat{\mathbf{a}}$ (a normalized vector):

1. Pass-through a point $\boldsymbol{p} = \boldsymbol{x}_g - \beta\hat{\mathbf{a}}$ for $\beta > 0$ in units of meters, where $\boldsymbol{x}_g$ is the end-effector goal.
2. Constrain the end-effector frame to align with the approach direction and vertical workspace orientation during the approach from $\boldsymbol{p}$ to $\boldsymbol{x}_g$.
3. Penalize the end-effector velocity at the passthrough point $\boldsymbol{p}$ as well as the acceleration to slow down smoothly in preparation for the final approach.

4.2.3 Speed Correction We represent the motion plan as a kinematic Linear Quadratic Regulator (LQR) around the local optimum encoding the problem's second-order information. Querying the LQR by rolling out a trajectory from a given start position and velocity approximates resolving the problem from that point since the LQR encodes the full solution to the problem's quadratic approximation. Therefore, to speed up and slow down the trajectory (ℓ_1), we rescale time variable of the LQR, itself, and requery the trajectory from the robot's current position and velocity state. That automatically implements smooth speedup and slowdown transitions to when moving to a new timescale.

4.2.4 Grasp Correction To account for modifications in the desired contact point during grasping (ℓ_4), we can simply modify the target position and orientation. Additionally, to account for the geometry of the object, we constrain the approach direction to be perpendicular to the grasp point.

5 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 (Figure 6). The implementation of our language model is based on an adaptation of the open-source Human to Structured

Correction	Description of First Video	Description of Second Video
Speed*	Move banana from right to left slowly	Move banana from right to left quickly
Speed*	Move baseball from right to left slowly	Move baseball from right to left quickly
Orientation	Move cup from right to left while spilling contents	Move cup from right to left without spilling contents
Orientation	Move wine glass from right to left while spilling contents	Move wine glass from right to left without spilling contents
Spatial constraint*	Move apple from right side of the box to left, go above box	Move apple from right side of the box to left, go behind box
Spatial constraint*	Move baseball from right side of the box to left, go above box	Move baseball from right side of the box to left, go behind box
Grasp point*	Pick up the cup from the handle	Pick up the cup from the side opposite the handle
Grasp point*	Pick up the teabox from the side	Pick up the teabox from the top

Table 1. Description of videos shown to participants on Amazon Mechanical Turk to gather training data for our language model. Video pairs marked with a * were shown to participants in both possible orders, as we expect users may want to express corrections in both directions along that dimension. The orientation examples were only shown in the direction indicated as we do not expect users to intentionally ask a manipulator to rotate while it moves.

Language (H2SL) software package,[†] which allows us to convert free-form text input to its corresponding structured language grounding in the robot's environment. The groundings in our model represent desired modifications to the trajectory planning as explained in Sections 4.1.2 and 4.1.1. In this application, our symbolic representation for natural language corrections is sufficiently small for real-time performance with the DCG; however, we could also employ the ADCG or HDCG for scenarios in which the symbolic representation becomes sufficiently complex.

Training data for the language model was gathered using Amazon Mechanical Turk (AMT),[‡] an online crowdsourcing platform used extensively by roboticists in the last few years for participant recruitment (Tellex et al. (2011b); Sorokin et al. (2010)).

Using AMT, 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*. That is, how would one best communicate to the assistive machine how to alter the path in the first video to achieve the path shown in the second video. Table 1 describes the videos that make up our training examples. In total, there were 14 pairs of videos.

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 a repeat of previous responses. This resulted in a total of 31 training examples. Our corpus was comprised of 100 phrases and an average of 3.23 phrases per example.

6 Evaluation

In this section we evaluate the full end-to-end system. The entire framework involves many complex hardware and software components, which include speech-to-text software,[§] a natural language understanding model (Sec 4.1) trained on Amazon Mechanical Turk data (Sec 5), the motion generation module that adapts online to corrections (Sec 4.2), and the MICO robotic arm. The system is implemented on real hardware, and operates in real-time. We first analyze the language model in Section 6.1 and then assess the current state of the full system with a pilot study in Section 6.2.

6.1 Language Model

We evaluate our model on the training dataset along similar metrics to those proposed in Howard et al. (2014b). First, we perform all possible combinations of leave-one-out cross-validation on our model. Using this approach, we correctly infer the constraint 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 not existing 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. We note that the average runtime is near one-fiftieth of a second as reported in Table 2.

	Run-time
Test Data	0.0236 +/- 0.01
Train Data	0.0253 +/- 0.0005

Table 2. Average runtime of constraint inference in seconds.

[†]https://github.com/tmhoward/h2sl

[‡]https://www.mturk.com/mturk/welcome

[§]Speech-to-text software was used during development of the system, but replaced with a command line interface during the pilot study to reduce external sources of error during evaluation. During development, speech recognition was incorporated through using a javascript library[¶] which was connected to the rest of the system through roslibjs, part of ROS Web Tools (Toris et al. (2015)).

6.2 Pilot Study

Here we report on a pilot study with novice users, which not only serves to demonstrate the real-time operation of the end-to-end system, but also allows us to analyze strengths and weaknesses of our approach. Additionally, we use this study to gather user feedback in preparation for a subject study by candidate end-users with motor impairments.

6.2.1 Study Protocol The study consisted of five participants without motor impairments recruited from the staff, students and visitors of the Rehabilitation Institute of Chicago. It included all seven types of corrections defined in Table 1. For each type of correction, we developed a specific task that was described to the user in simple language (e.g. "move to the other side of the block").

Users triggered a 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 (see Section 4.2). 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.

The following is an overview of the protocol used in each trial:

- Pre-Trial: Users were told the initial command for the given trial. For example, "pick up the box." They were then shown a pair of videos from the set used to produce the training data. The first video demonstrated how the MICO planned to solve the task, and the second video demonstrated a modification to that plan. Users were asked to come up with language that describes how to correct the behavior of the real arm to better match what was demonstrated in the second video. This approach was taken to reduce bias in the user's free-form language.
- Initialization: The MICO was initialized with the goal described to the user in the pre-trial phase.
- Recognition of undesirable behavior: Once the arm began to move, the user was free to choose when to stop the motion in order to apply a correction.
- Correction: The user described their desired correction with the language of their choosing.

Each trial was executed with a distinct set of initial conditions—in particular, the starting configuration and environment are specific to each task. Each scene is simple and intended to place specific focus on the type of correction desired by the user. The initial configuration of the arm is the same in the speed, orientation and position dimensions, and the arm starts on the right side of the workspace. In the grasping trials, the arm is initialized directly in front of the base to maximize the available configuration space for manipulation tasks. Correction dimensions that do not necessitate the presence of objects, such as the speed and rotation dimensions, are unobstructed. Those that do require an object, such as the position and grasping dimensions, include a single object so as not to distract from the participant's perception of the interaction. Example initializations can be seen in the far left column of Figure 4.

An example of this workflow can be seen in Figure 1. In each experiment, we record the following data—the initial planned trajectory, the user utterance, the updated planned trajectory accounting for the user input, whether our language model can parse the user input, whether our language model can ground the input, and whether the grounding matches the desired update. At the end of each trial, we also asked the user for feedback on the language understanding, the motion generation and the speed and responsiveness of the full system.

6.2.2 Results 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 35 full trials (5 users, 7 trials each). In this analysis, a full trial refers to the final attempt for a given correction (either the furthest a user got after 3 attempts, or the result of a successful attempt).

During our experiments, all 35 trials were successfully parsed, after which point and 14 failed to ground (remaining 21 trials were complete successes). If we include all failed attempts, we collected a total of 68 user inputs, zero of which failed to parse, 40 of which failed to ground, 7 of which were grounded incorrectly, and 21 of which were full successes.

6.2.3 Discussion of Results 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.

We first notice that zero of the attempts resulted in a *failure to parse*. We can attribute the generalizability of this part of the pipeline to our use of an open-source CKY statistical parser.[‖] The parser was trained on much larger datasets (the QuestionBank and Penn Treebank datasets) than our AMT dataset, which resulted in a robust system, particularly for the reduced set of language we observed during our experiments. In the future, the parser can be easily replaced and further improved through extended datasets and more modern approaches to parsing as described in Andor et al. (2016).

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 corrective language and that we either need a more complete set of features or we need to include more training data. Through further analysis, we recognized that the majority of failures were due to a dearth of training data related to two particular corrections

[‖] https://github.com/emilmont/pyStatParser dimensions (Rows 4 and 8 in Fig. 4). As a result, no participant was able to successfully apply the desired correction during the experimental trials that evaluated those correction dimensions. This result clearly indicates the need for a larger data collection when training the language model. If we ignore these two tasks, totaling 10 full trials (2 per user), we find that 21 of the 25 trials were successfully parsed, grounded and applied to the planner based on the user input. Additionally, of the remaining 4 trials that were not able to be grounded from this reduced set of 25 tasks, 2 more would have been successful if we had the word *above* in our training set. Unfortunately, all related data in our training set used the word *over* instead of *above*. This problem can be addressed in multiple ways, one of which is to recognize synonyms and other related phrases through word embeddings as described in Mikolov et al. (2013). However, by addressing the primary cause of many of the experimental failures-a lack of training data-we can also hope to improve the systems generalizability to a variety of different phrasings.

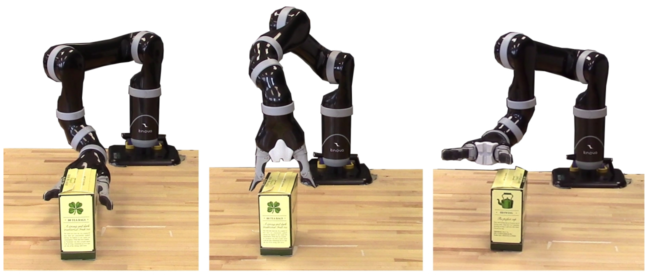
Figure 7. Example experimental result of ambiguous input from a user that resulted in a grounding different from the user's true desire. In the initial trajectory the arm tries to pick the box up on the side (left). In the desired correction the arm should pick the box up from the top (middle). Instead, the correction provided by the user ("over") resulted in the arm moving above the box without trying to grasp it (right).

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. Digging further into the experimental data, we see further evidence that this is the case in 5 of the 7 attempts. In particular, we find that 5 of the cases occurred in attempts related to the desired correction displayed in Row 8 of Figure 4. With limited training data, our language model was unable to appropriately distinguish between when a user wanted to pick up a box from the side or the top. In the other 2 attempts where there was a *mismatch* between the grounding and the desired correction, our analysis shows that the user provided ambiguous input that could easily be interpreted in the manner presented by our language model. For example, in one experiment in which 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 during its trajectory, as opposed to the desired grounding of a goal symbol suggesting that the arm pick up the box from its top. The resulting modification to the initial trajectory can be seen in Figure 7. The desire still remains, however, to account for ambiguous inputs, as the likelihood that a real user would provide such an input when using an assistive robot is quite high. We believe one interesting way to handle this is to incorporate the initial command into our language model. By building the task into the factor graph model, we can reason about the desired correction based on the high level goal and improve our ability to disambiguate user input.

6.2.4 User Survey Results Finally, after each full trial, we asked the users to give us their impressions on (1) how well they felt the system understood their commands, and (2) how consistent the corrected behavior was with their desired correction. To gather this data, the participants filled out their responses on a 7 point Likert scale where a response of 1 is low and a response of 7 is high. Unsurprisingly, we find a strong correlation between the success of the system and user satisfaction. When the system failed to ground and apply the user correction, the mean and standard deviation of responses to the three questions was as follows (1) $\mu = 1.15, \sigma = 0.36$ and (2) $\mu = 1.15, \sigma = 0.36$. When the system succeeded, the means rose to (1) $\mu = 6.86, \sigma = 0.45$ and (2) $\mu = 6.68, \sigma = 0.63$.

7 Discussion

Previous research has shown language to be a good method of communicating high-level goals to robot counterparts. We have extended this prior work to the space of real-time corrections, because we believe it will be equally important for users to be able to customize *how* a robot achieves the desired goal—since this will allow users to impart personal preferences on how a trajectory is planned and executed, in addition to adding robustness. This is particularly important when interacting with a robot that has complex control dynamics in unknown environments.

Another valid approach is to define a pre-specified language structure (Shimizu and Haas (2009)) that can be taught to the user in order to issue their desired correction. However, this approach greatly limits the generalizability of the system. Beyond the addition of a required learning phase, we believe that relying on a fixed-structure language interface will reduce the acceptance of these systems by users as minute mistakes in phrasing, or a poor understanding of the interface, can render the full system unusable.

In future work we hope to show that, in addition to instantaneous corrections, this method can also be used to learn user preferences and contextual knowledge over time. Furthermore, we expect to include additional correction dimensions, including differentiating between near-by objects, desired grasping strength and spatial constraints on all sub-links of the robotic arm in addition to the end-effector.

We also recognize a need to improve our strategy for coping with grounding failures. In particular, we would like to incorporate a measure of ambiguity in the inference process to automatically recognize points of uncertainty. If the model is then unsure of a particular inference it can then re-initiate communication with the user to resolve the ambiguity. An example of a dialog system like this is presented in Tellex et al. (2013).

We also do not address negative language. Negative language, such as the correction "do not move the ball over the box", produces a more challenging grounding problem and would increase the difficulty of natural language symbol grounding. In future work we hope to specifically address the problem of incorporating negative language into our model for understanding natural language corrections.

Another modification we plan to incorporate in future work is allowing users to decide whether or not they would like to stop the robot before providing their correction. In particular, we believe that there may really be two distinct methods by which people would like to interact with and provide corrections to their assistive device. In the first, the user will stop the device, and provide the correction. This type of interaction is likely preferential in scenarios where the original action is dangerous (e.g. interaction with another object) or not appropriate (e.g. moving a glass full of liquid while rotating the wrist). In the second type of interaction, we believe users would like to apply a correction without stopping the device. This type of interaction is likely preferential for quick modifications, like increasing or decreasing the speed. This again highlights one of the benefits of our planning approach, as the speed with which a planned trajectory is executed can be modified without re-planning by querying the trajectory with different dilation factors as described in Section 4.2.

8 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. 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.

This work aims to aid users of assistive robots by allowing non-experts to modify the behavior of their personal robots to better match their preferences through a natural language interface. Our target domain in particular is assistive robotic manipulators used by persons with severe motor impairments. In this paper 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.

By focusing on properties of previously planned trajectories this approach also generalizes to other assistive robotic devices such as mobile robotics. Additionally, this approach allows us to focus on robust, long-term behavior demonstrating the capacity for personal robots to take corrective actions in response to spoken stimuli. We believe our collaborative approach will improve human-robot communication and aid in the adoption of assistive robotic technologies.

Acknowledgments

We would like to thank Pascal Egli for his help during the pilot study and figure generation. We would also like to acknowledge the technical support of Jan Issac and Daniel Kappler in preparing the motion optimization software for the experiments. This material is based upon work supported by the National Science Foundation under Grant CNS 1329891. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the aforementioned institutions.

References

Nichola Abdo, Cyrill Stachniss, Luciano Spinello, and Wolfram Burgard. Robot, Organize My Shelves! Tidying up Objects by Predicting User Preferences. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1557–1564, 2015.

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. Globally Normalized Transition-Based Neural Networks. *arXiv:1603.06042*, 2016.

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

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

Brenna D. Argall, Brett Browning, and Manuela M. Veloso. Policy Feedback for the Refinement of Learned Motion Control on a Mobile Robot. *International Journal of Social Robotics*, 4(4): 383–395, 2012.

Daniel Barber, Thomas M. Howard, and Matthew R. Walter. A multimodal interface for real-time soldier-robot teaming. In *Proceedings of the SPIE 9837, Unmanned Systems Technology XVIII*, May 2016.

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.

Adrian Boteanu, Jacob Arkin, Thomas M. Howard, and Hadas Kress-Gazit. A model for verifiable grounding and execution of complex language instructions. In *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 2016.

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.

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.

Stevan Harnad. The Symbol Grounding Problem. *Physica D: Nonlinear Phenomena*, 42(1):335–346, 1990.

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 *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 5608–5615, 2015.

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) Workshop on Rehabilitation and Assistive Robotics*, 2014a.

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

Chien-Ming Huang, Maya Cakmak, and Bilge Mutlu. Adaptive Coordination Strategies for Human-Robot Handovers. In *Proceedings of Robotics: Science and Systems*, July 2015.

Ashesh Jain, Debarghya Das, Jayesh K Gupta, and Ashutosh Saxena. PlanIt: A Crowdsourced Approach for Learning to Plan Paths from Large Scale Preference Feedback. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 877–884, 2015.

Dae-Jin Kim, Ryan Lovelett, and Aman Behal. An Empirical Study with Simulated ADL Tasks using a Vision-Guided Assistive Robot Arm. In *Proceedings of the International Conference on Rehabilitation Robotics (ICORR)*, pages 504–509, 2009.

Jens Kober and Jan R Peters. Policy Search for Motor Primitives in Robotics. In *Proceedings of Advances in Neural Information Processing Systems*, pages 849–856, 2009.

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

Thorsten Luith, Darko Ojdanić, Ola Friman, Oliver Prenzel, and Axel Graser. Low Level Control in a Semi-Autonomous Rehabilitation Robotic System via a Brain-Computer Interface. In *Proceedings of the International Conference on Rehabilitation Robotics (ICORR)*, pages 721–728, 2007.

Matt MacMahon, Brian Stankiewicz, and Benjamin Kuipers. Walk the Talk: Connecting Language, Knowledge, and Action in Route Instructions. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 1475–1482, 2006.

Jeremy Maitin-Shepard, Marco Cusumano-Towner, Jinna Lei, and Pieter Abbeel. Cloth Grasp Point Detection Based on Multiple-View Geometric Cues with Application to Robotic Towel Folding. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 2308–2315, 2010.

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.

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.

Hongyuan Mei, Mohit Bansal, and Matthew R. Walter. Listen, Attend, and Walk: Neural Mapping of Navigational Instructions to Action Sequences. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, February 2016.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed Representations of Words and Phrases and their Compositionality. In *Proceedings of Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

Scott Niekum, Sarah Osentoski, George Konidaris, Sachin Chitta, Bhaskara Marthi, and Andrew G Barto. Learning grounded finite-state representations from unstructured demonstrations. *The International Journal of Robotics Research*, 34(2):131–157, 2015.

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

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

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, 2007.

Nobuyuki Shimizu and Andrew R Haas. Learning to Follow Navigational Route Instructions. In *Proceeings of International Joint Conference on Artificial Intelligence (IJCAI)*, volume 9, pages 1488–1493, 2009.

Alexander Sorokin, Dmitry Berenson, Siddhartha Srinivasa, and Martial Hebert. People Helping Robots Helping People: Crowdsourcing for Grasping Novel Objects. In *Proceeings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2117–2122, 2010.

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, 2011a.

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 *Proceedings of AAAI Conference on Artificial Intelligence, North America*, pages 1507–1514, 2011b.

Stefanie Tellex, Pratiksha Thaker, Robin Deits, Dimitar Simeonov, Thomas Kollar, and Nicholas Roy. Toward information Theoretic Human-Robot Dialog. In *Proceedings of Robotics: Science and Systems*, 2013.

Russell Toris, Julius Kammerl, David Lu, Jihoon Lee, Odest Chadwicke Jenkins, Sarah Osentoski, Mitchell Wills, and Sonia Chernova. Robot Web Tools: Efficient Messaging for Cloud Robotics. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 4530–4537, 2015.

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)*, pages 619–625, 2007.

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)*, pages 540–544, 2005.

Daqing Yi, Thomas M. Howard, Kevin Seppi, and Michael Goodrich. Expressing homotopic requirements for mobile robot navigation through natural language instructions. In *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 2016.

A Index to Multimedia Extensions

The multimedia extensions to this article are at: www.ijrr.org.

Extension	Media Type	Description
1	Video	Demonstrates two experimental trials of the described system which allows users to provide natural language corrections to alter the behavior of their assistive robot.