Asking for Help Using Inverse Semantics

Stefanie Tellex[1]
Brown University
Box 1910
Providence, RI 02912, USA
stefie10@cs.brown.edu

Ross A. Knepper[1]
MIT CSAIL
32 Vassar St.
Cambridge, MA 02139, USA
rak@csail.mit.edu

Adrian Li
University of Cambridge
Department of Engineering
Cambridge CB2 1PZ, UK
alhl2@cam.ac.uk

Thomas M. Howard
MIT CSAIL
32 Vassar St.
Cambridge, MA 02139, USA
tmhoward@csail.mit.edu

Daniela Rus
MIT CSAIL
32 Vassar St.
Cambridge, MA 02139, USA
rus@csail.mit.edu

Nicholas Roy
MIT CSAIL
32 Vassar St.
Cambridge, MA 02139, USA
nickroy@csail.mit.edu

ABSTRACT

Robots inevitably fail, often without the ability to recover autonomously. We demonstrate an approach for enabling a robot to recover from failures by communicating its need for specific help to a human partner using natural language. Our approach automatically detects failures, then generates targeted spoken-language requests for help such as "Please give me the white table leg that is on the black table." Once the human partner has repaired the failure condition, the system resumes full autonomy. We present a novel *inverse semantics* algorithm for generating effective help requests. In contrast to forward semantic models that interpret natural language in terms of robot actions and perception, our inverse semantics algorithm generates requests by emulating the human's ability to interpret a request using the Generalized Grounding Graph (G^3) framework. To assess the effectiveness of our approach, we present a corpus-based online evaluation, as well as an end-to-end user study, demonstrating that our approach increases the effectiveness of human interventions compared to static requests for help.

1. INTRODUCTION

Robotic capabilities such as robust manipulation, accurate perception, and fast planning algorithms have led to recent successes such as robots that can fold laundry [Maitin-Shepard et al., 2010], cook dinner [Bollini et al., 2012], and assemble furniture [Knepper et al., 2013]. However, when robots execute these tasks autonomously, failures often occur due to perceptual errors, manipulation failures, and other issues. A key aim of current research is reducing the incidence of these types of failures but eliminating them completely remains an elusive goal.

When failures occur, a human can often intervene to help a robot recover. When the human is familiar with the robot and its task as well as its common failure modes, they can often provide this help without an explicit request. However, if a person is unfamiliar with the robotic system and not knowledgeable about its capabilities and limitations, they might not know how to help the robot recover from a failure. This situation will occur frequently when robots interact with untrained users in the home. Moreover, even trained users who are deeply familiar with the robot's capabilities may experience problems during times of high cognitive load, such as a human supervising a large team of robots on a factory floor.

[1]The first two authors contributed equally to this paper.

HRI '14 Germany

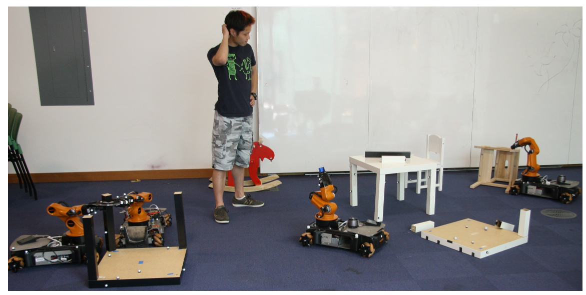

Figure 1: A robot engaged in assembling an IKEA LACK table requests help using natural language. A vague request such as "Help me" is challenging for a person to understand. Instead, this paper presents an approach for generating targeted requests such as "Please hand me the black table leg."

To address these problems, we propose an alternative approach to recovering from the inevitable failures which occur when robots execute complex tasks in real-world environments: when the robot encounters failure, it verbally requests help from a human partner. After receiving help, it continues executing the task autonomously. The contribution of our paper is a family of algorithms for formulating the pithiest unambiguous natural language request so that a human not otherwise cognitively engaged can render appropriate aid.

Our algorithm generates natural language requests for help by searching for an utterance that maximizes the probability of a correspondence between the words in the language and the action the robot desires the human to perform, mak-

(a) (b) (c)

Figure 2: During autonomous assembly, circumstances occasionally arise that the robot cannot correct. When the arrangement of parts does not permit the robot to reach its target, it may request human assistance (a). After this brief human intervention (b), autonomous operation resumes (c).

ing use of the G^3 (Generalized Grounding Graph) model of a person's language understanding faculty [Tellex et al., 2011]. When understanding language, the G^3 framework maps from linguistic symbols to low-level motor actions and perceptual features that the robot encounters in the environment. In this paper, we invert the model, mapping from a desired low-level motor action that the robot would like the human to execute to a symbolic linguistic description. By modeling the probability of a human misinterpreting the request, the robot is able to generate targeted requests that humans follow more quickly and accurately compared to baselines involving either generic requests (e.g., "Help me") or template-based non-context-specific requests.

As a test domain, we focus on a human-robot team assembling pieces of IKEA furniture, shown in Figure 1. We evaluate our approach using a corpus-based evaluation with Amazon Mechanical Turk as well as a user study. The corpus-based approach allows us to efficiently test the performance of different algorithms and baselines. The user study assesses whether we have met our engineering goals in the context of an end-to-end system. Our evaluation demonstrates that the inverse semantics language generation system improves the speed and accuracy of a human's intervention when a human-robot team is engaged in a furniture assembly task and also improves the human's subjective perception of their robotic teammates.

2. RELATED WORK

Traditional methods for generating language rely on a dedicated language-generation system that is not integrated with a language-understanding framework [Jurafsky and Martin, 2008, Reiter and Dale, 2000]. These approaches typically consist of a sentence planner combined with a surface realizer to guide decision making of what to say, but contain no principled model of how an instruction-follower would comprehend the instruction [Striegnitz et al., 2011, Garoufi and Koller, 2011, Chen and Mooney, 2011]. Our approach differs in that it generates by inverting a module for language understanding.

Some previous work has approached the generation problem by inverting a semantics model. Golland et al. [2010] use a game-theoretic approach combined with a semantics model to generate referring expressions. Our approach, in contrast, uses probabilistic grounded semantics yielding emergent biases towards shorter sentences unless a longer, more descriptive utterance is unambiguous. Goodman and Stuhlmüller [2013] describes a rational speech-act theory of language understanding, where the speaker chooses actions that maximize expected global utility. Similarly, recent work has used Dec-POMDPs to model implicatures and pragmatics in language-using agents [Vogel et al., 2013a,b] but without focusing on grounded, situated language as in this paper. There is a deep connection between our models and the notion of legibility and predictability for grasping, as defined by Dragan and Srinivasa [2013]. Roy [2002] presents an algorithm for generating referring expressions in a two-dimensional geometric scene which uses an ambiguity score to assess the quality of candidate descriptions. Our algorithm, in contrast, generates complete requests rather than noun phrases and asks the listener to follow a complex request rather than simply selecting an object.

Our approach views the language generation problem as inverse language understanding, building on the G^3 approach described by Tellex et al. [2011]. A large body of work focuses on language understanding for robots [MacMahon et al., 2006, Dzifcak et al., 2009, Kollar et al., 2010, Matuszek et al., 2012]. The G^3 framework particularly lends itself to inversion because it is a probabilistic framework which explicitly models the mapping between words in language and aspects of the external world, so metrics based on entropy may be used to assess the quality of generated utterances.

Cooperative human-robot activities, including assembly, have been broadly studied [Wilson, 1995, Simmons et al., 2007, Dorais et al., 1998, Fong et al., 2003]. These architectures permit various granularities of human intervention through a sliding autonomy framework. A failure triggers the replay of video of the events preceding failure, from which the human must obtain situational awareness. In contrast, our approach leverages natural language to convey to the user exactly how the problem should be resolved.

3. ASSEMBLING FURNITURE

Our assembly system comprises a team of KUKA youBots, which collaborate to assemble pieces of IKEA furniture [Knepper et al., 2013]. A team of robots receives assembly instructions encoded in a STRIPS-style planning language. A centralized executive takes as input the symbolic plan and executes each plan step in sequence. Each symbolic action corresponds to a manipulation or perception action to be performed by one or two robots. To assemble the simple LACK table, execution of the 48-step plan takes approximately ten minutes when no failures occur. In our experiments, failures occurred at a rate of roughly one every two minutes. Since perception is not a focus of this paper, we employ a VICON motion capture system to track the loca-

```
function conditions_satisfied(ℓ – list of conditions)
 1: q ← World state
 2: for all c ∈ ℓ do
 3:     if c not satisfied in q then
 4:         a ← generate_remedy_action(c)          ▷ See Section 3.2
 5:         generate_help_request(a)               ▷ See Section 4
 6:         while c not satisfied do
 7:             if time > 60 then                  ▷ wait up to 60 seconds
 8:                 return false
 9: return true
function executive(g – goal state)
 1: repeat
 2:     p ← symbolic_plan(g)                       ▷ p – list of actions
 3:     f ← true                                   ▷ f – are we finished?
 4:     while p ≠ ∅ do
 5:         s ← p[0]                               ▷ first plan step
 6:         if conditions_satisfied(s.preconditions) then
 7:             s.execute()
 8:             if not conditions_satisfied(s.postconditions) then
 9:                 f ← false
10:         else
11:             f ← false
12:         p.retire(s)                            ▷ s succeeded; remove it from p
13: until f                                        ▷ no actions failed
```

Figure 3: A simple executive algorithm generates robot actions and help requests.

tion of each participating robot, human and furniture part during the assembly process. Thus the team is aware of the steps to assemble the furniture. When the team detects a failure, they request help using one of the approaches described in Section 4. Figure 3 shows the algorithm used to control the robots and request help.

3.1 Detecting Failures

To detect failures, the system compares the expected state of the world to the actual state, as sensed by the perceptual system (line 6 of the `executive` function). We represent the state, q, as a vector of values for logical predicates. For example, elements of the state for the IKEA LACK table include whether the robot is holding each table leg, whether the table is face-up or face-down, and whether each leg is attached to the table. In the furniture assembly domain, we compute the state using the tracked pose of every rigid body known to a VICON system, including each furniture part, each robot chassis and hand, and each human. The VICON state, $x \in \mathbb{R}^n$, is continuous and high-dimensional. We implemented a function f that maps x onto the lower-dimensional state vector q. The system recomputes q frequently, since it may change independently of any deliberate robot action, such as by human intervention or from an unintended side-effect.

Prior to executing each action, the assembly executive verifies the action's preconditions against q. Likewise, following each action, the postconditions are verified. Any unsatisfied condition indicates a failure and triggers the assembly executive to pause the assembly process and initiate error recovery. For example, the robot must be grasping a table leg before screwing it into the hole. If it tries and fails to pick up a leg, then the post-condition for the "pick up" action will not be satisfied in q and detects a failure.

3.2 Recovery Strategy

When a failure occurs, its description takes the form of an unsatisfied condition. The system then asks the human for help to address the problem. The robot first computes actions that, if taken, would resolve the failure and enable it to continue assembling the piece autonomously. The system computes these actions using a pre-specified model of physical actions a person could take to rectify failed preconditions. Remedy requests are expressed in a simple symbolic language. This symbolic request, a, specifies the action that the robot would like the person to take to help it recover from failures. However these symbolic forms are not appropriate for speaking to an untrained user. In the following section, we explore a series of approaches that take as input the symbolic request for help and generate a language expression asking a human for assistance.

4. ASKING FOR HELP FROM A HUMAN PARTNER

Once the system computes a symbolic representation of the desired action, a, it searches for words, Λ, which effectively communicate this action to a person in the particular environmental context, M, following line 5 of the `conditions_satisfied` function. This section describes various approaches to the `generate_help_request` function which carries out this inference. Formally, we define a function h to score possible sentences:

$$\underset{\Lambda}{\operatorname{argmax}}\ h(\Lambda, a, M) \tag{1}$$

The specific function h used in Equation 1 will greatly affect the results. We define three increasingly complex approaches for h which lead to more targeted natural language requests for help by modeling the ability of the listener to understand it. The contribution of this paper is a definition for h using *inverse semantics*. Forward semantics is the problem of mapping between words in language and aspects of the external world; the canonical problem is enabling a robot to follow a person's natural language commands [MacMahon et al., 2006, Kollar et al., 2010, Tellex et al., 2011, Matuszek et al., 2012]. Inverse semantics is the reverse: mapping between specific aspects of the external world (in this case, an action that the robot would like the human to take) and words in language. To apply this approach we use the G^3 model of natural language semantics. Previously, we used the G^3 framework to endow the robot with the ability to follow natural language commands given by people. In this paper, instead, we use G^3 as a model for a person's ability to follow natural language requests.

The inference process in Equation 1 is a search over possible sentences Λ. We define a space of sentences using a context-free grammar (CFG). The inference procedure creates a grounding graph for each candidate sentence using the parse structure derived from the CFG and then scores it according to the function h.

4.1 Understanding Language

This section briefly describes the model for understanding language; then the following sections describe how to invert it. When understanding language, the G^3 framework imposes a distribution over *groundings* in the external world, $\gamma_1 \ldots \gamma_N$, given a natural language sentence Λ. Groundings are the specific physical concepts that are referred to by the language and can be objects (e.g., a table leg or a robot), places (e.g., a particular location in the world), paths (e.g., a trajectory through the environment), or events (e.g., a sequence of actions taken by the robot). Each grounding corresponds to a particular constituent $\lambda_i \in \Lambda$. For example, for a sentence such as "Pick up the table leg," the

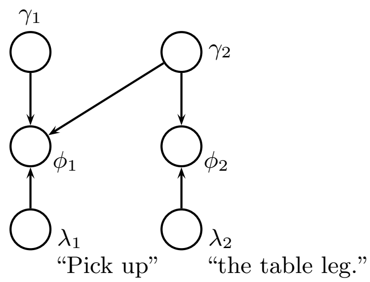

Figure 4: Grounding graph for the request, "Pick up the table leg." Random variables and edges are created in the graphical model for each constituent in the parse tree. The λ variables correspond to language; the γ variables correspond to groundings in the external world. Edges in the graph are created according to the parse structure of the sentence.

grounding for the phrase "the table leg" corresponds to an actual table leg in the external world, and the grounding for the entire sentence corresponds to the actions of a person as they follow the request. Understanding a sentence in the G^3 framework amounts to the following inference problem:

$$\underset{\gamma_1 \cdots \gamma_N}{\operatorname{argmax}}\, p(\gamma_1 \ldots \gamma_N | \Lambda, M) \tag{2}$$

The environment model M consists of the robot's location along with the locations and geometries of objects in the external world. The computed environment model defines a space of possible values for the grounding variables, $\gamma_1 \ldots \gamma_N$. A robot computes the environment model using sensor input; in the domain of furniture assembly, the system creates the environment model using input from VICON.

To factor the model, we introduce a correspondence vector, Φ, as in Tellex et al. [2011]. Each entry $\phi_i \in \Phi$ corresponds to whether linguistic constituent $\lambda_i \in \Lambda$ corresponds to the groundings associated with that constituent. For example, the correspondence variable would be $True$ for the phrase "the white table leg" and a grounding of a white leg, and $False$ if the grounding was a different object, such as a black table top. We assume that $\gamma_1 \ldots \gamma_N$ are independent of Λ unless Φ is known. Introducing Φ enables factorization according to the structure of language with local normalization at each factor over a space of just the two possible values for ϕ_i.

The optimization then becomes:

$$\underset{\gamma_1 \cdots \gamma_N}{\operatorname{argmax}}\, p(\Phi | \Lambda, \gamma_1 \ldots \gamma_N, M) \tag{3}$$

We factor the expression according to the compositional syntactic structure of the language Λ.

$$\underset{\gamma_1 \cdots \gamma_N}{\operatorname{argmax}} \prod_i p(\phi_i | \lambda_i, \gamma_{i_1} \ldots \gamma_{i_k}, M) \tag{4}$$

This factorization can be represented as a directed graphical model where random variables and edges in the model are created according to the structure of the language. We refer to one of these graphical models as a *grounding graph*. Figure 4 shows an example graphical model; the details of the factorization are described by Tellex et al. [2011].

4.2 Speaking by Reflex

The simplest approach from the assembly executive's perspective is to delegate diagnosis and solution of the problem to the human with the simple fixed request, Λ = "Help me." This algorithm takes into account neither the environment or the listener when choosing what to say. We refer to this algorithm as S_0.

4.3 Speaking by Modeling the Environment

Next, we describe a more complex model for speaking, that takes into an account a model of the environment, but not a model of the listener. We compute this model using the G^3 framework. The system converts the symbolic action request a to a value for the grounding variable, $\gamma_a \in \Gamma$. This variable, γ_a, corresponds to the entire sentence; we refer to the value of γ_a as γ_a^*. It then searches for the most likely sentence Λ according to the semantics model. Equation 1 becomes:

$$\underset{\Lambda}{\operatorname{argmax}}\, h(\Lambda, \gamma_a^*, M) \tag{5}$$

To speak using a model of the environment, the robot searches for language that best matches the action that the robot would like the human to take. It does not consider other actions or groundings in any way when making this determination. Formally:

$$h(\Lambda, \gamma_a^*, M) = \max_{\Gamma | \gamma_a = \gamma_a^*} p(\Lambda | \Gamma, M) \tag{6}$$

With the correspondence variable, this function is equivalent to:

$$h(\Lambda, \gamma_a^*, M) = \max_{\Gamma | \gamma_a = \gamma_a^*} p(\Phi | \Lambda, \Gamma, M) \tag{7}$$

We refer to this metric as S_1 because the speaker does not model the behavior of the listener at all, but simply tries to say something that matches the desired action γ_a^* in the environment with high confidence.

4.4 Speaking by Modeling the Listener and the Environment

The previous S_1 metric scores shorter, ambiguous phrases more highly than longer, more descriptive phrases. For example, "the white leg" will always have a higher score than "the white leg on the black table" because the corresponding grounding graph for the longer sentence is identical to the shorter one except for an additional factor, which causes the overall probability for the more complex graph to be lower (or at most equal). However, suppose the robot team needs a specific leg; for example, in Figure 5, the robots might need specifically the leg that is on the black table. In this case, if the robot says "Hand me the white leg," the person will not know which leg to give to the robot because there are several legs in the environment. If the robot instead said, "Please hand me the white leg that is on the black table," then the person will know which leg to give to the robot.

To address this problem, we augment our robot with a model of the listener's ability to understand a request in the particular environment. More specifically, rather than simply maximizing the probability of the action given the request, we minimize the uncertainty a listener would experience when using the G3 model to interpret the request.We refer to this metric as S_2 because it includes a model of the listener's uncertainty in its computation. The S_2 metric measures the probability that the listener will correctly understand the requested action γ_a^*:

$$h(\Lambda, \gamma_a^*, M) = p(\gamma_a = \gamma_a^* | \Phi, \Lambda, M) \tag{8}$$

To compute this metric, we marginalize over values of Γ, where $\gamma_a = \gamma_a^*$:

$$h(\Lambda, \gamma_a^*, M) = \sum_{\Gamma | \gamma_a = \gamma_a^*} p(\Gamma | \Phi, \Lambda, M) \tag{9}$$

We factor the model with Bayes' rule:

$$h(\Lambda, \gamma_a^*, M) = \sum_{\Gamma | \gamma_a = \gamma_a^*} \frac{p(\Phi | \Gamma, \Lambda, M) p(\Gamma | \Lambda, M)}{p(\Phi | \Lambda, M)} \tag{10}$$

We rewrite the denominator as a marginalization and conditional distribution on Γ':

$$h(\Lambda, \gamma_a^*, M) = \sum_{\Gamma | \gamma_a = \gamma_a^*} \frac{p(\Phi | \Gamma, \Lambda, M) p(\Gamma | \Lambda, M)}{\sum_{\Gamma'} p(\Phi | \Gamma', \Lambda, M) p(\Gamma' | \Lambda, M)} \tag{11}$$

The denominator is constant so we can move the summation to the numerator:

$$h(\Lambda, \gamma_a^*, M) = \frac{\sum_{\Gamma | \gamma_a = \gamma_a^*} p(\Phi | \Gamma, \Lambda, M) p(\Gamma | \Lambda, M)}{\sum_{\Gamma'} p(\Phi | \Gamma', \Lambda, M) p(\Gamma' | \Lambda, M)} \tag{12}$$

Next we assume that $p(\Gamma | \Lambda, M)$ is constant, K, for all Γ, so it can move outside the summation. This term is constant because Γ and Λ are independent when we do not know Φ:

$$h(\Lambda, \gamma_a^*, M) = \frac{K \times \sum_{\Gamma | \gamma_a = \gamma_a^*} p(\Phi | \Gamma, \Lambda, M)}{K \times \sum_{\Gamma'} p(\Phi | \Gamma', \Lambda, M)} \tag{13}$$

The constant K cancels, yielding:

$$h(\Lambda, \gamma_a^*, M) = \frac{\sum_{\Gamma | \gamma_a = \gamma_a^*} p(\Phi | \Gamma, \Lambda, M)}{\sum_{\Gamma'} p(\Phi | \Gamma', \Lambda, M)} \tag{14}$$

This equation expresses the S_2 metric. It finds a sentence, Λ, that minimizes the entropy of the distribution over γ_a given Λ by modeling the ability of the listener to understand the language. Specifically, note that computing the denominator of Equation 14 is equivalent to the problem of understanding the language in the particular environment because the system must assess the mapping between the language Λ and the groundings Γ' for all possible values for the groundings. In our implementation we use the G^3 framework to compute an approximation for this term. In practice, the inference step is expensive, so we limit the overall number of language candidates to the top 10 most confident, as in our previous work of following natural language commands [Tellex et al., 2011].

4.5 Training

To train the model, we collected a new dataset of natural language requests given by a human to another human in the furniture assembly domain. We created twenty-one videos of a person executing a task involved in assembling a piece of furniture. For example, one video shows a person screwing a table leg into a table, and another shows a person handing a table leg to a second person. The people and objects in the video are tracked with VICON, so each video has an associated context consisting of the locations, geometries, and trajectories of the people and objects. We asked annotators on Amazon Mechanical Turk to view the videos and write a natural language request they would give to ask one of the people to carry out the action depicted in the video. Then we annotated requests in the video with associated groundings in the VICON data. The corpus contains 326 requests with a total of 3279 words. In addition we generated additional positive and negative examples for the specific words in our context-free grammar.

"Help me" (S_0)	"Help me."
Templates	"Please hand me part 2."
G^3 S_1	"Give me the white leg."
G^3 S_2	"Give me the white leg that is on the black table."
Hand-written Request	"Take the table leg that is on the table and place it in the robot's hand."

Figure 5: Scene from our dataset and the requests generated by each approach.

4.6 Template Baseline

As a baseline, we implemented a template-based algorithm with a lookup table of requests given to a human helper, similar to the approach of Fong et al. [2003] among others. These generic requests take the following form:

- "Place part 2 where I can see it,"
- "Hand me part 2," and
- "Attach part 2 at location 1 on part 5." (i.e. screw in a table leg)

Note that the use of first person in these expressions refers to the robot. Since VICON does not possess any semantic qualities of the parts, they are referred to generically by part identifier numbers. Such templates can be effective in simple situations, where the human can infer the part from the context. However, the ambiguity can become hard to track. At best, the programmer could encode a look-up table of semantic descriptors, such as "white leg" instead of "part 2," but even in this case, the template baseline can be expected to falter in complex situations with multiple white legs.

5. EVALUATION

The goal of our evaluation was to assess whether our algorithms increase the effectiveness of a person's help, or in other words, to enable them to more quickly and accurately provide help to the robot. To evaluate whether our algorithms enable a human to accurately provide help compared to baselines, we use an online corpus-based evaluation. In addition we conducted a user study to assess whether our

Table 1: Fraction of Correctly Followed Requests

Metric	% Success	95% Confidence
Chance	20.0	
"Help me" Baseline (S_0)	21.0	± 8.0
Template Baseline	47.0	± 5.7
G^3 Inverse Semantics with S_1	52.3	± 5.7
G^3 Inverse Semantics with S_2	64.3	± 5.4
Hand-Written Requests	94.0	± 4.7

leading algorithm improves the speed and accuracy of a person's help to a team of robots engaged in a real-world assembly task.

5.1 Corpus-Based Evaluation

Our online evaluation used Amazon Mechanical Turk (AMT) to measure whether people could use generated help requests to infer the action that the robot was asking them to perform. We presented a worker on AMT with a picture of a scene, showing a robot, a person, and various pieces of furniture, together with the text of the robot's request for help. Figure 5 shows an example initial scene, with several different requests for help generated by different algorithms, all asking the human to carry out the same action. Next, we asked the worker to choose an action that they would take that best corresponds to the natural language request. Since the worker was not physically in the scene and could not directly help the robot, we instead showed them videos of a human taking various actions in the scene and asked them to choose the video that best matched the request for help. We chose actions to film based on actions that would recover from typical failures that the robots might encounter. A trial consists of a worker viewing an initial scene paired with a request for help and choosing a corresponding video.

We created a dataset consisting of twenty trials by constructing four different initial scenes and filming an actor taking five different actions in each scene. For each trial we generated requests for help using five different methods. We present results for the four automatic methods described in Section 4, as well as a baseline consisting of hand-written requests which we created by experimentation on AMT to be clear and unambiguous. For the "help me" and hand-written baselines, we issued each of the twenty generated requests to five subjects, for a total of 100 trials. We issued each request in the template and G^3 approaches to fifteen users for a total of 300 trials. Results appear in Table 1.

Our results show that the "Help me" baseline performs at chance, whereas the template baseline and the G^3 inverse semantics model both improved performance significantly. The S_1 model may have improved performance over the template baseline, but these results do not rise to the level of statistical significance. The S_2 model, however, realizes a significant improvement, $p = 0.002$ by Student's t-test, due to its more specific requests, which model the uncertainty of the listener. These results demonstrate that our model successfully generates help requests for many conditions.

5.2 User Study

In our experiment, humans and robots collaborated to assemble pieces of IKEA furniture. The study split 16 participants into two conditions, using a between-subjects design, with 8 subjects in each condition. In the baseline condition, robots requested help with the S_0 approach, using only the words "Please help me." In the test condition, robots requested help using the S_2 inverse semantics metric. Our goal was to assess whether our system meets our engineering goals: for a user with limited situational awareness, the end-to-end human/robot furniture assembly system would show increased effectiveness when generating requests using the inverse semantics metric (S_2) compared to the "help me" metric (S_0). The accompanying video is online at http://youtu.be/2Ts0W4SiOfs.

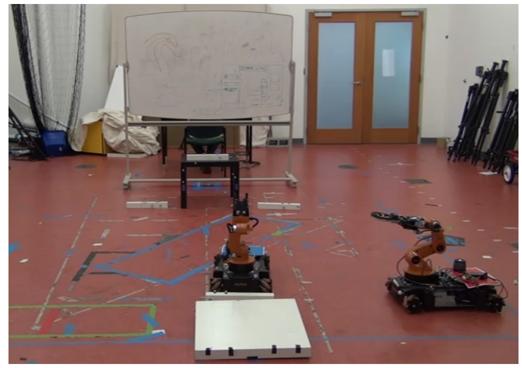

Figure 7: Initial configuration for the user study. The user is seated behind the whiteboard in the background.

We measure effectiveness by a combination of objective and subjective measures. The objective measures are most important, as they directly indicate the ability of our approach to improve effectiveness of the complete human-robot team. We report two objective measures: *efficiency* – the elapsed time per help request, and *accuracy* – the number of error-free user interventions. Taken together, these measures show how effectively the human's time is being used by the robots. We also report two subjective measures derived from a post-trial survey, as well as our observations of the subjects and their own written feedback about the system, to gain an understanding of their view of the strengths and weaknesses of our approach.

5.2.1 Procedure

Subjects in each condition were gender-balanced and had no significant difference in experience with robots or furniture assembly. To familiarize users with the robot's capabilities, we gave them a list of actions that might help the robots. During preliminary trials, subjects had problems when handing parts to the robot (called a hand-off), so we demonstrated this task and then gave each user the opportunity to practice. The entire instruction period lasted less than five minutes, including the demonstration. During the experiment, we instructed users to focus on a different assembly task and only help the robots when requested.

For each subject, the robot team started from the same initial conditions, shown in Figure 7. Some failures were inevitable given the initial conditions (e.g., a table top turned upside down; a part on a table out of the robots' reach.) Other failures happened naturally (e.g., a table leg that slipped out of a robot's gripper.) When a failure occurred during assembly, the robot team first addressed the person by saying, "Excuse me." Next, the system generated and spoke a request for help through an on-board speaker and also projected it on a large screen to remove dependence

Take the table leg that is on the table and place it in the robot's hand.
Take the table leg that is under the table and place it in the robot's hand.
Take the table leg that is next to the table and place it in the robot's hand.
Pick up the table leg that is on the table and hold it.
Take the table leg that is on the table and place it on the floor in front of the robot.

(a)

Screw the white table leg into the hole in the table top.
Screw the black table leg into the hole in the table top.
Take the white table leg and insert it in the hole, but do not screw it in.
Move the white table leg over near the table top.
Take the table top and place it near the white table leg on the floor.

(b)

Take the white table leg that is next to the table and put it in front of the robot.
Take the black table leg that is next to the table and put it in front of the robot.
Take the black table leg that is far away from the table and put it in front of the robot.
Take the white table leg that is on top of the table and place it in the robot's hand.
Pick up the white table leg next to the table and hold it.

(c)

Take the white table, flip it over, and set it down in place.
Take the black table, flip it over, and set it down in place.
Take the white table and move it near the robot, keeping it upside-down.
Pick up the white table and hold it.
Take the white table, flip it over, and put it in the robot's hand.

(d)

Figure 6: The four initial scenes from the evaluation dataset, together with the hand-written help requests used in our evaluation.

on understanding synthesized speech. Finally, the human intervened in whatever way they felt was most appropriate.

After communicating a help request, the robots waited up to 60 seconds for the user to provide help, while monitoring whether the precondition that triggered the failure had been satisfied. If the the environment changed in a way that satisfied the request, the robot said "Thank you, I'll take it from here," and we counted the person's intervention as successful. In cases where the allotted time elapsed, the robot instead said "Never mind, I'll take it from here," and moved on to a different part of the assembly process. These instances were recorded as failed interventions. For each intervention, we recorded the time elapsed and number of actions the human took in attempting to solve the problem.

Each trial ran for fifteen minutes. Although we tried to limit experimenter intervention, there were several problems with the robotic assembly system that required expert assistance. Experimenters intervened when either of two situations arose: potential damage to the hardware (19 times), or an infinite loop in legacy software (15 times). In addition, software running on the robots crashed and needed to be restarted 5 times. In the future, we plan to address these issues using methods for directing requests to the person most likely to satisfy them, rather than only targeting requests at untrained users.

5.2.2 Results and Discussion

Over the course of the study, the robots made 102 help requests, of which 76 were satisfied successfully within the 60-second time limit. The most common request type was the hand-off, comprising 50 requests. For the non-hand-off requests, we observed a significant improvement in intervention time for the test condition (25.1 sec) compared to baseline (33.3 sec) with $p = 0.092$ by t-test. For hand-off requests, differences in elapsed time between the two conditions did not rise above the level of statistical significance. After observing the trials, we noticed that subjects found it difficult to successfully hand a part to the robot, despite our initial training.

To assess accuracy of interventions, we observed the initial action attempted for each intervention and whether it was the action desired by the robot. In ambiguous situations, the user often tried one or more methods of helping the robot before arriving at the correct solution or running out of time. For the baseline "Help me" case, the user led with the correct action in 57.1% of interventions, compared to 77.6% for the test method, $p = 0.039$ by t-test. This difference indicates that the inverse semantics method enabled the robot to more successfully communicate its needs to the human compared to the baseline, thus enabling the person to efficiently and effectively aid the robot.

We observed a difference in a third objective measure, the overall success rate, although the difference failed to reach statistical significance. Baseline condition users satisfied 70.3% of the robot's requests within 60 seconds, whereas 80% of inverse semantics requests did so, $p = 0.174$ by t-test. Many users failed to successfully help the robot due to the difficulty of handoffs or due to other issues in the robotic system, pointing to the many non-linguistic factors affecting the success of our system.

We also report two subjective measures derived from a post-trial survey. We asked users to score the robot's effectiveness at communicating its needs on a 5-point Likert scale. Users found the natural language condition much more effective than the baseline condition with a significance of $p = 0.001$ by Kruskal-Wallis test. Second, we asked whether users felt more effective working with the robots on two assembly tasks at once, or working alone on one kit at a time. Users preferred parallelism significantly more in the natural language condition than in the baseline condition, with a significance of $p = 0.056$ by Kruskal-Wallis test.

Together, these results show that the inverse semantics method often improved the speed and accuracy of human subjects as they provided help to the robot team. Moreover, our subjective results show that humans felt that the robots were more effective at communicating when they used the inverse semantics system and that they were more effective when working with the robotic team. Qualitatively, subjects preferred the language generation system; Figure 8 shows comments from participants in the study in each condition: even when users successfully helped the robots in the baseline condition, they frequently complained that they did not know what the robot was asking for.

Despite these promising successes, important limitations remain. A significant problem arose from the ability of the robots' to accept handoffs from minimally trained human

"Help me" Baseline

"I think if the robot was clearer or I saw it assemble the desk before, I would know more about what it was asking me."

"Did not really feel like 'working together as a team' – For more complex furniture it would be more efficient for robot to say what action the human should do?"

"The difficulty is not so much working together but the robots not being able to communicate the actual problem they have. Also it is unclear which ones of the robots has the problem."

G^3 Inverse Semantics with S_2

"More fun than working alone."

"I was focused on my own task but could hear the robot when it needed help.... However, I like the idea of having a robot help you multitask."

"There was a sense of being much more productive than I would have been on my own."

Figure 8: Comments from participants in our study.

users. Our results suggest that improving the nonverbal communication that happens during handoff would significantly improve the overall effectiveness of our system. Second, a significant limitation of the overall system was the frequent intervention by the experimenters to deal with unexpected failures. Both of these conditions might be modified by a more nuanced model of the help that a human teammate could provide. For example, if the robots could predict that handoffs are challenging for people to successfully complete, they might ask for a different action, such as to place the part on the ground near the robot. Similarly, if the robots were able to model the ability of different people to provide targeted help, they might direct some requests to untrained users, and other requests to "level 2" tech support. The different types of interventions provided by the experimenters compared to the subjects points to a need for the robots to model specific types of help that different people can provide, as in Rosenthal et al. [2011].

5.3 Conclusion

The goal of our evaluation was to assess the effectiveness of various approaches for generating requests for help. The corpus-based evaluation compares the inverse semantics method to several baselines in an online evaluation, demonstrating that the inverse semantics algorithm significantly improves the accuracy of a human's response to a natural language request for help compared to baselines. Our end-to-end evaluation demonstrates that this improvement can be realized in the context of a real-world robotic team interacting with minimally trained human users. This work represents a step toward the goal of mixed-initiative human-robot cooperative assembly.

Our end-to-end evaluation highlights the strength of the system, but also its weakness. Robots used a single model for a person's ability to act in the environment; in reality, different people have different abilities and willingness to help the robot. Second, because the robots spoke to people, requesting help, some subjects responded by asking clarifying questions. Developing a dialog system capable of answering questions from people in real time could provide disambiguation when people fail to understand the robot's request. As we move from robot-initiated to mixed-initiative communication, the reliance on common ground and context increases significantly. Since our models can be expected to remain imperfect, the demand for unambiguous sentences becomes less satisfiable. In the long term, we aim to develop robots with increased task-robustness in a variety of domains by leveraging the ability and willingness of human partners to assist robots in recovering from a wide variety of failures.

6. ACKNOWLEDGMENTS

This work was supported in part by the Boeing Company, and in part by the U.S Army Research Laboratory under the Robotics Collaborative Technology Alliance.

The authors thank Dishaan Ahuja and Andrew Spielberg for their assistance in conducting the experiments.

References

M. Bollini, S. Tellex, T. Thompson, N. Roy, and D. Rus. Interpreting and executing recipes with a cooking robot. In *13th International Symposium on Experimental Robotics*, 2012.

D. L. Chen and R. J. Mooney. Learning to interpret natural language navigation instructions from observations. In *Proc. AAAI*, 2011.

G. Dorais, R. Banasso, D. Kortenkamp, P. Pell, and D. Schreckenghost. Adjustable autonomy for human-centered autonomous systems on mars, 1998.

A. Dragan and S. Srinivasa. Generating legible motion. In *Robotics: Science and Systems*, June 2013.

J. Dzifcak, M. Scheutz, C. Baral, and P. Schermerhorn. What to do and how to do it: Translating natural language directives into temporal and dynamic logic representation for goal management and action execution. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 4163–4168, 2009.

T. Fong, C. Thorpe, and C. Baur. Robot, asker of questions. *Journal of Robotics and Autonomous Systems*, 42:235–243, 2003.

K. Garoufi and A. Koller. Combining symbolic and corpus-based approaches for the generation of successful referring expressions. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 121–131. Association for Computational Linguistics, 2011.

D. Golland, P. Liang, and D. Klein. A game-theoretic approach to generating spatial descriptions. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 410–419. Association for Computational Linguistics, 2010.

N. D. Goodman and A. Stuhlmüller. Knowledge and implicature: Modeling language understanding as social cognition. *Topics in cognitive science*, 5(1):173–184, 2013.

D. Jurafsky and J. H. Martin. *Speech and Language Processing*. Pearson Prentice Hall, 2 edition, May 2008. ISBN 0131873210.

R. A. Knepper, T. Layton, J. Romanishin, and D. Rus. IkeaBot: An autonomous multi-robot coordinated furniture assembly system. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, Karlsruhe, Germany, May 2013.

T. Kollar, S. Tellex, D. Roy, and N. Roy. Toward understanding natural language directions. In *Proc. ACM/IEEE Int'l Conf. on Human-Robot Interaction*, pages 259–266, 2010.

M. MacMahon, B. Stankiewicz, and B. Kuipers. Walk the talk: Connecting language, knowledge, and action in route instructions. In *Proc. Nat'l Conf. on Artificial Intelligence (AAAI)*, pages 1475–1482, 2006.

J. Maitin-Shepard, J. Lei, M. Cusumano-Towner, and P. Abbeel. Cloth grasp point detection based on multiple-view geometric cues with application to robotic towel folding. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, Anchorage, Alaska, USA, May 2010.

C. Matuszek, N. FitzGerald, L. Zettlemoyer, L. Bo, and D. Fox. A joint model of language and perception for grounded attribute learning. *Arxiv preprint arXiv:1206.6423*, 2012.

E. Reiter and R. Dale. *Building Natural Language Generation Systems*. Cambridge University Press, Jan. 2000. ISBN 9780521620369.

S. Rosenthal, M. Veloso, and A. K. Dey. Learning accuracy and availability of humans who help mobile robots. In *Proc. AAAI*, 2011.

D. Roy. A trainable visually-grounded spoken language generation system. In *Proceedings of the International Conference of Spoken Language Processing*, 2002.

R. Simmons, S. Singh, F. Heger, L. M. Hiatt, S. C. Koterba, N. Melchior, and B. P. Sellner. Human-robot teams for large-scale assembly. In *Proceedings of the NASA Science Technology Conference*, May 2007.

K. Striegnitz, A. Denis, A. Gargett, K. Garoufi, A. Koller, and M. Theune. Report on the second second challenge on generating instructions in virtual environments (give-2.5). In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 270–279. Association for Computational Linguistics, 2011.

S. Tellex, T. Kollar, S. Dickerson, M. Walter, A. Banerjee, S. Teller, and N. Roy. Understanding natural language commands for robotic navigation and mobile manipulation. In *Proc. AAAI*, 2011.

A. Vogel, M. Bodoia, C. Potts, and D. Jurafsky. Emergence of gricean maxims from multi-agent decision theory. In *Proceedings of NAACL 2013*, 2013a.

A. Vogel, C. Potts, and D. Jurafsky. Implicatures and nested beliefs in approximate Decentralized-POMDPs. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, Sofia, Bulgaria, August 2013b. Association for Computational Linguistics.

R. Wilson. Minimizing user queries in interactive assembly planning. *IEEE Transactions on Robotics and Automation*, 11(2), April 1995.