Make it so: Continuous, Flexible Natural Language Interaction with an Autonomous Robot

Daniel J. Brooks[1], Constantine Lignos[2], Cameron Finucane[3], Mikhail S. Medvedev[1], Ian Perera[4], Vasumathi Raman[3], Hadas Kress-Gazit[3], Mitch Marcus[2], Holly A. Yanco[1]

[1]University of Massachusetts Lowell, Department of Computer Science, 1 University Avenue, Lowell, MA 01854
{dbrooks,mmedvede,holly}@cs.uml.edu
[2]University of Pennsylvania, Department of Computer and Information Science, 3330 Walnut Street, Philadelphia, PA 19104
{lignos, mitch}@cis.upenn.edu
[3]Cornell University, Department of Computer Science, 4130 Upson Hall, Ithaca, NY 14853
cpf37@cornell.edu, vraman@cs.cornell.edu, hadaskg@cornell.edu
[4]University of Rochester, Department of Computer Science, 734 Computer Studies Bldg., Rochester, NY 14627, iperera@cs.rochester.edu

Abstract

While highly constrained language can be used for robot control, robots that can operate as fully autonomous subordinate agents communicating via rich language remain an open challenge. Toward this end, we developed an autonomous system that supports natural, continuous interaction with the operator through language before, during, and after mission execution. The operator communicates instructions to the system through natural language and is given feedback on how each instruction was understood as the system constructs a logical representation of its orders. While the plan is executed, the operator is updated on relevant progress via language and images and can change the robot's orders. Unlike many other integrated systems of this type, the language interface is built using robust, general purpose parsing and semantics systems that do not rely on domain-specific grammars. This system demonstrates a new level of continuous natural language interaction and a novel approach to using general-purpose language and planning components instead of hand-building for the domain. Language-enabled autonomous systems of this type represent important progress toward the goal of integrating robots as effective members of human teams.

1 Introduction

Robots have the ability to play a unique role in a team of humans in scenarios such as search and rescue where safety is a concern. Traditionally the overhead of interacting with such systems has been high, making integration of robots into human teams difficult. Improvements in natural language technology may, however, allow for this overhead to be reduced if a system can understand natural language input well enough to carry out the required scenario and maintain contact with human team members. In this paper we present a system that makes progress towards the use of robots as capable members of a human team by providing continuous, flexible, and grounded natural language communication with the robot throughout execution.

For an autonomous system to be fully integrated as a member of a human team, its communication abilities must exhibit several characteristics. A system must be able to communicate during the pre-execution phase, understanding natural language and its grounding in the physical world and reporting its understanding of the plan to the team. It must also be able to recieve its orders in a team discussion that avoids the hassle and inconvenience of requiring its operator to learn and translate instructions into a coded syntax and allows the other human team members to understand the role the autonomous system will be playing.

Interpreting natural language instructions and translating them into comprehensive plans for an autonomous system is a non-trivial task, and even with state-of-the-art language technology, errors are common. Due to the possibility of error, the operator must be aware of how the system understands its orders and what it intends to do. This can be accomplished if the system provides some way of reporting how the system has interpreted the operator's input.

Grounded natural language communication also requires the system to resolve references to objects in the world. Given that the robot and the operator can be in different locations, the robot must be able to consider the operator's location in resolving the intended referent. Furthermore, communication may refer to unseen or possible entities, yet the system must remember the appropriate action to take upon finding an instance of the entity.

The system must also exhibit the ability to continue interaction after completing execution of prior instructions. Often, interactions between an operator and robot occur in stages. That is, the operator may give the robot instructions and then wait to see what the outcome is before deciding on what the next course of action should be. The system must be able to remember information from previous instructions and be able to apply it to future instructions. For example, if the operator specifically tells the robot to watch out for an object because it is important, they should not have to repeat this information each time they ask the robot to perform a different task. Additionally, an operator may find that they need more information than what the system may have al-

ready provided them. In such cases, it is useful to be able to simply query the system about information it has already acquired, rather then repeating the previous task with slightly modified instructions.

2 Related Work

Approaches to language interfaces for interacting with robots span a wide range, including keywords (Perzanowski et al. 2001), simple grammars (Jones and Rock 2002), and grammars with well-articulated semantics (Dzifcak et al. 2009). A persistent issue has been the robustness of the language interface to the challenges posed by spontaneous speech (Cantrell et al. 2010). Regardless of whether the noise in natural language understanding comes from disfluency, acoustic noise, or simply errors in text processing systems, it is important that systems provide as much feedback as possible as to what was understood from the operator's commands. With little exception (Teixeira et al. 2003), regardless of the architecture of the system used the vocabulary and grammar are typically tuned to a small domain relevant to the scenario presented.

Allowing for richer modalities of communication between operator and robot may aid in understanding the robot's comprehension. The multimodal interface of Perzanowski et al. (2001) could interpret commands from spoken language, hand gestures for describing either distances or directions, and a dynamically generated map presented to the user on a tablet.

Natural interaction with autonomous systems is an active area of research. Chernova et al. (2011) conducted a study in which they collected data using crowd sourcing by creating an online game in which two people interacted with each other, one pretending to be a robot and the other pretending to be a human collaborator. They then used this information to generate natural behaviors for an actual robot during interactions with people in a real world mockup of the online game. Talamadupula et al. (2011) have examined planning for human robot teams in an open world using the `SapaReplan` planner. Their work focused on techniques for maintaing up-to-date world models and goals that are shared between humans and robots. Shah et al. (2011) use strategies based on human-human teaming to improve human-robot teaming, demonstrating the value of frequent updates in teaming tasks.

Tellex et al. (2011) have similar goals in interpreting natural language commands in that they find groundings in the environment to satisfy arguments to commands, but take a statistical approach that infers a plan through a probabilistic graphical model incorporating the language of the command and the available groundings.

Being able to adequately describe to an autonomous system is, however, often only the first half of the problem. The behavior of an autonomous system may still be puzzling to the operator, and thus an open avenue of research is developing autonomous systems that can sufficiently explain their decisions and actions (Brooks et al. 2010). This is particularly important for operators trying to understand unexpected behaviors so they can maintain better control in the future.

Commander: Tell me if you see any hostages.
Robot: I'll let you know if I see a hostage.
C: Defuse all the bombs you see.
R: Got it. I'll defuse all bombs.
C: Search the library, classroom, and lab.
R: Got it. I'll search the library, search the classroom, and search the lab.
C: Make it so.
R: Understood. I'm carrying out your orders now.
R: I see a hostage.
R: I'm now going to defuse in the library.
R: I'm done, and I'm in the classroom.

Figure 1: Sample interaction with the system.

3 System Goals

The application domain for this system is an urban search and rescue scenario where an autonomous mobile robot acts as part of a team of humans working to explore the area and react to the environment as required. The robot acts as the commander's subordinate, receiving orders and carrying them out. The robot's primary purpose is reconnaissance, entering areas that may be unsafe ahead of human team members. It is assumed that the commander and robot will rarely be colocated. Interaction with the system is implemented through a multimodal tablet interface that acts as a conduit for both sending instructions to the robot and displaying information about the remote situation and environment.

In the sample interaction shown in Figure 1, the system is asked to report when it encounters hostages, instructed to defuse any bombs it finds, and given a set of rooms to search. The system analyzes commands as they are given to it, and when it is told to begin carrying out orders it forms a plan from those commands and begins execution. The robot maintains contact with the operator during execution, informing the operator about anything it was explicitly asked to mention in addition to anything related to the goals it was given.

4 Architecture and Implementation

The operator interacts with the system using a tablet computer, currently an Apple iPad. The system is comprised of modular software subsystems which were assembled into a single system using ROS (Quigley et al. 2009). The robot used in our system is an iRobot ATRV Jr.

Our system's overall architecture, shown in Figure 2 is analogous in design to a three layer architecture (Gat 1998). The natural language components of the system make up the highest layer, which transforms natural language into logical statements, synthesizes a finite state automaton to carry out the requested plan, and communicates status back to the commander. The middle layer is formed by a hybrid controller, which maintains the current discrete state of the automaton given input from the environment, and the job scheduler and state manager on the robot. These modules work together to transform the logical propositions into a controller for the robot's actions. Finally, low-level continuous behaviors which primarily interact with the dynamics

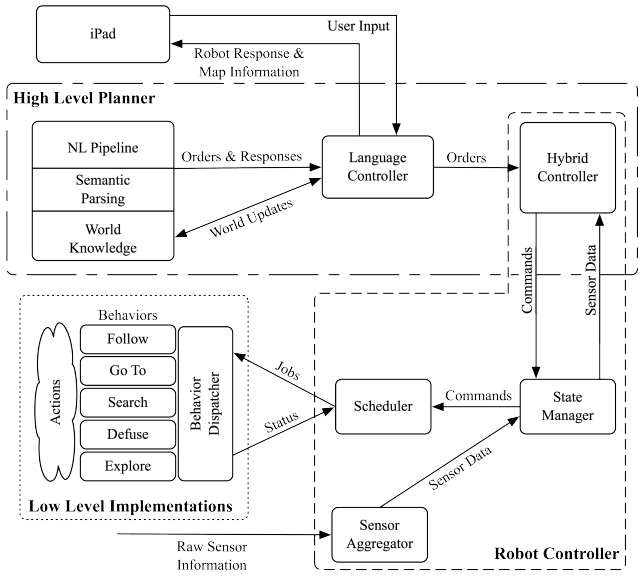

Figure 2: The end-to-end system architecture.

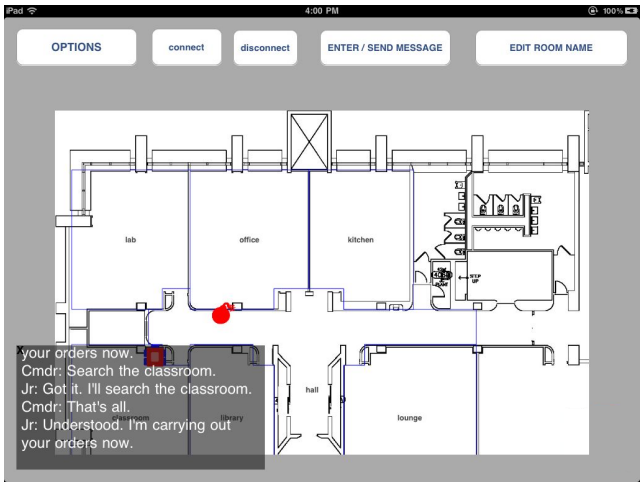

Figure 3: The operator interface shows the current dialog state in the lower left along with the robot position and an icon for a bomb the robot has identified.

of the changing world are implemented on the robot to carry out the requested actions.

4.1 Operator Interface

The operator commands the system through the use of a tablet computer. Natural language utterances are the primary form of communication between the operator and robot and are entered into the system using an interface similar in design to an instant messaging or text messaging program.

Information about the environment is reported by the robot using visual notifications on the display in addition to language notifications for important events, as shown in Figure 3. The interface also contains a map mode, which displays information about the layout of the world and the location of key objects within the world, including the robot's position. The map layout may be known in advance or produced by the robot as it explores. Map mode is considered to be a secondary form of communication which serves to augment the natural language interaction. It provides a source of common understanding to ground the conversation between the commander and robot by showing the objects or places relevant to the robot's operation. The map interface also displays camera imagery from the robot, allowing the commander to see various objects of interest such as bombs and hostages as they are identified by the robot.

4.2 Natural Language Processing

The operator enters natural language instructions into the user interface and each sentence is processed through a pipeline of natural language systems. These systems identify the syntactic structure of the sentences and extract semantic information from them.

While many previous systems have relied on per-scenario grammars that allow the unification of semantic information and natural language representations (Dzifcak et al. 2009), we adopted an approach of using a combination of robust, general purpose components. An advantage of such an approach compared to a per-scenario grammar is that the core language models need not be modified across scenarios; it is only the planning component that takes in the semantic structure that needs to be able to transform a general semantic representation into the relevant robot behaviors for the scenario.

Parsing. We use the Bikel parser (Bikel 2004) combined with the null element restoration of Gabbard, Marcus, and Kulick (2006) to parse sentences. Before being given to the parser, the input is tagged using MXPOST (Ratnaparkhi and others 1996). Null elements are the silent subjects and objects in sentences that appear in structures such as imperatives and relative clauses. For example, in the sentence *Go to the hallway and defuse the bomb* there is an understood subject *you* that is the subject of both the *go* and *defuse* verbs. To ease semantic interpretation, coordinate structures are split such that they are equivalent to two full clauses. For example, in *Go to the hallway and defuse the bomb* becomes *[You] go to the hallway* and *[You] defuse the bomb* through a combination of null element restoration and coordinate structure handling.

Semantic interpretation. The semantic interpretation module uses the parse tree to extract verbs and their arguments. For example, from the sentence *Go to the hallway and defuse the bomb*, the desired structure would be that the robot should *go(hallway)* and *defuse(bomb)*.

To extract verbs and their arguments from parse trees, we developed a module that used VerbNet (Schuler 2005), a large database of verbs and the types of arguments they take. The VerbNet database identifies verbs as members of senses, verbs which in similar contexts have similar meanings, such as *search*, *scout*, *scavenge* being roughly equivalent. VerbNet also provides the expected argument structure for each verb, for example the expected structure for *[You] put the*

ball on the desk would be NP V NP PP-THEME, where NP is a noun phrase, V is a verb, and a PP-THEME is a prepositional phrase that is the target of the verb.

To extract semantic information using VerbNet, verbs in each tree are located and possible matching frames for the verb are identified. The match that expresses the most semantic roles is selected, so when there is ambiguity the most specific possible interpretation is used.

Grounding. Given the semantic interpretation, we must generate a reference to a specific entity or location in the environment to compile the action into the planning logic.

To resolve these exophoric references, we make an assumption that the commander will only refer to objects or locations they know about. Under this assumption, we implement a mutual knowledge system (Clark 1981) by storing both the objects that the robot knows of and the objects the robot thinks the commander knows of. The supposed commander knowledge is updated by combining the robot's knowledge with its awareness of the commander's location.

Locations are resolved by finding a room labeled with the location extracted from the semantic parsing. To resolve object references, we first check whether the argument to the action is something that can be sensed by the robot, and generate a message if it is not. If the argument is an indefinite reference, such as "any bombs," the action is added as a standing order. If the argument is a pronoun, we try to resolve the anaphoric reference through the history of discourse between the robot and commander. If the argument is a definite reference, we check the mutual knowledge base to find a matching object. Finally, if we cannot resolve the reference, then the robot returns an error response.

4.3 High Level Planner

The ultimate goal of the natural language interface is to produce a continuous controller for the robot. This controller should decide, based on information from the robot's sensors, the actions the robot should take in order to correctly achieve its goals. To produce the controller from natural language, semantic representations of the operators orders are converted into a set of linear temporal logic (LTL) formulas. The logical form is then automatically synthesized into a correct-by-construction finite state automaton (FSA) using LTLMoP (Finucane, Jing, and Kress-Gazit 2010). This FSA is then used as the robot's controller.

4.4 Robot Controller

The controller generated from the language deploys different low-level robot behaviors based on the goals and the state of the robot and the environment. This controller automatically reacts to different environment events, as perceived by the robot's sensors.

State and Sensor Management. One of the responsibilities of the robot is presenting information about the world as perceived by its sensors to the generated controller. The granularity of the raw sensor data is too fine for the controller, thus the data is abstracted into discrete events. Sensor output is filtered and fused with other data to create a more concise description of the world. This is done by taking pieces of raw data and interpreting them into various types of information such as abstract location, for example which room the robot is in, and what agents and objects are present in the current room. The interpreted sensor data, along with information about the location of the robot and the current behaviors being executed, form the system's state, which is reported to the controller.

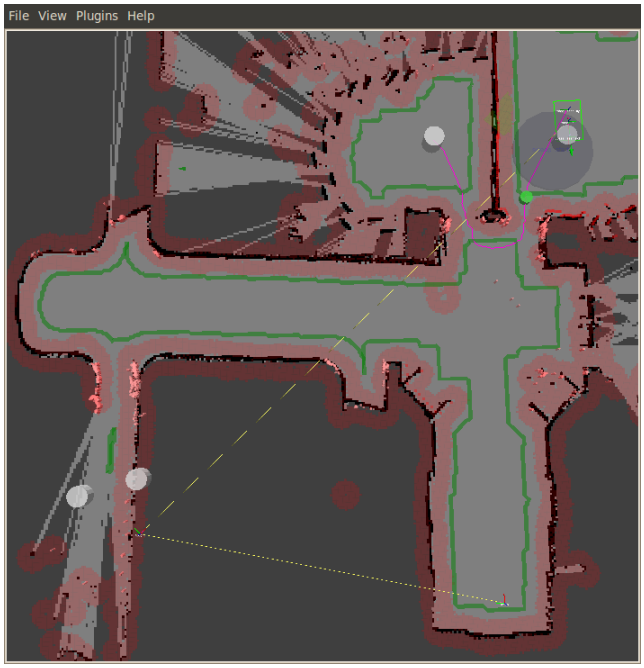

Figure 4: The robot generates a map of the environment in a fronter-based exploration scenario.

Low Level Implementation. The robot is currently capable of six *behaviors*, including driving to a location, exploring (map building), performing a generic search of an area, following a person, retrieving (asking for) objects, and (simulated) disarming explosives. A behavior is composed of a set of rules for starting, stopping, and suspending execution, along with logic that controls the execution of one or more *actions*. An action is an activity, atomic at the planner's level, that the robot can perform. There are currently four actions implemented: drive, area sweep, explore, and follow. Actions react dynamically to the environment, can keep state, and even perform some lower level planning. However, they do not take into account the state of the overall system or what other actions are currently running. Actions often make use of shared resources such as drive train motors and can also be used by multiple behaviors.

Mapping, Region Discovery, and Exploration. An important aspect of being able to effectively communicate with the system is the ability for both the operator and robot to accurately refer to places in the world using names. This system supports this feature through the use of either static or dynamically generated maps. In some cases, the map of the area in which the robot is operating is known in advance,

through the existence of building plans or previous experience. These maps can be preloaded into the system before deployment, which gives the operator the advantage of being able to refer to places in their instructions. However, in many cases the layout of the world is not known ahead of time, and a map must be generated. This can be achieved through the use of a strategy for frontier-based exploration (Yamauchi 1997) and simultaneous localization and mapping (Grisetti, Stachniss, and Burgard 2007). We analyze the structure of the world by drawing polygon outlines on the map to denote *regions*, which are assigned names. In addition, we map which polygons can be reached from other polygons using a connected graph. The polygon definitions, names, and connected graph is stored as a topological graph which can be used by the system for planning. We created an automated system to identify regions using an algorithm (Fabrizi and Saffiotti 2000) which uses a process of dilation and erosion of walls to determine distinct places within the world, and a process called water-shedding to determine their connectivity to each other. An example of such regions being identified as the robot explores is shown in Figure 4.

5 Examples of System Usage

We now demonstrate the system in a scenario where a person acting as the commander used a real robot to simulate a search and rescue scenario. Before retrieving the hostages, the commander needed to search the building for bombs. Hostages, bombs, and users were represented as boxes with fiducial markers in the scenario to provide a simple simulation of perception.

5.1 Known Map

The interaction shown in Figure 5 demonstrates how the commander specified a plan to the robot and received feedback as the robot understood each command, with corresponding line numbers. The commander first issued a standing order for the robot to notify the commander when any hostages were seen (line 01). When the robot was told to look for a particular human collaborator, the robot interpreted the commands as requiring a search of all rooms in which the user could be present. The request to get the defuser from the the user and bring it back here shows how the system can give information about how it resolved commands to the commander. When the commander says "here," the robot resolved it to its current location, the hall. This inference is made explicit so the commander has the ability to correct any misunderstandings.

When the commander completed giving orders (line 07), the robot formed a plan and began to execute it. The robot did not inform the commander of every action taken, instead only notifying when it was explicitly asked to (i.e., hostages), if it acted on a standing order, or when it completed its mission. When the robot identified the user and requested the defuser, the commander was notified. After completing the mission, the robot informed the commander that it was idle.

Not every interaction results in successful understanding. Once the robot had the bomb defuser, the commander needed to instruct the robot to use it to defuse bombs. In cases where the system was able to extract nothing of use from the utterance, in this case because VerbNet did not contain an appropriate form of *disarm*, the system reported that it did not understand the utterance at all (line 12). Another possible failure mode is that the system extracted the semantic structure but it did not understand how to carry out the command. In the case of the second command tried, *defuse the dynamite*, the system recognized that the verb was *defuse* and the object was *the dynamite* but it did not understand that dynamite is the same thing as a bomb. The responses the system gives help to make it apparent when the commander's word choice is at fault; the system is able to express that it understood the syntax but does not know the object in this example. With such feedback, the operator may adjust its vocabulary when repeating the command (line 15).

The robot's standing orders remain in effect across plans

01 **Commander**: Tell me if you see any hostages.
02 **Robot:** I'll let you know if I see a hostage.
03 **C:** Look for User 2 in the library and the classroom.
04 **R:** Got it. I'll search the library and search the classroom.
05 **C:** Get the defuser from User 2 and come back here.
06 **R:** Got it. I'll retrieve the defuser from user_2 and go to the hall.
07 **C:** Make it so.
08 **R:**Understood. I'm carrying out your orders now.
The robot searches the classroom and does not find the user. It then searches the library, identifies the user, and begins to get the defuser.
09 **R:** I'm now going to get the defuser in the library.
In addition to the language notification, an icon showing the location of the user is shown on the operator interface along with a picture of the user captured by the robot. After getting the defuser, the robot returns to the hall.
10 **R:** I'm done, and I'm in the hall.
11 **C:** Disarm the dynamite.
12 **R:** I'm sorry, I didn't understand that at all.
13 **C:** Defuse the dynamite.
14 **R:** Sorry, I don't know anything about dynamite.
15 **C:** Defuse the bombs.
16 **R:** Got it. I'll defuse all bombs.
17 **C:** Search the lab.
18 **R:** Got it. I'll search the lab.
19 **C:** Make it so.
20 **R:** Understood. I'm carrying out your orders now.
The robot drives to the lab and searches it, finding a hostage but no bombs. The robot displays a picture of the hostage on the operator's interface.
21 **R:** I see a hostage.
22 **R:** I'm done, and I'm in the lab.
23 **C:** Search the office.
The robot drives to the office and searches it, finding a bomb during the search. It displays a picture of the bomb on the operator's interface.
24 **R:** I'm now going to defuse in the office.
The robot defuses the bomb and then completes its search of the room.
25 **R:** I'm done, and I'm in the office.

Figure 5: Interaction with the robot before and during plan execution.

01 **Commander:** Go to the classroom.
02 **Robot:** Got it. I'll go to the classroom.
03 **C:** Make it so.
04 **R:** Understood. I'm carrying out your orders now.
The robot begins driving to the lab.
05 **C:** Stop.
06 **R:** Aye sir, full stop. I'm in the hall.
The robot stops immediately.
07 **C:** Go to the lab.
08 **R:** Got it. I'll go to the lab.
09 **C:** Nevermind
10 **R:** Okay, I'll belay those orders.
The robot clears the "go to the lab" command from its orders.
11 **C:** Go to the lounge.
12 **R:** Got it. I'll go to the lounge.
13 **C:** Make it so.
14 **R:** Understood. I'm carrying out your orders now.
The robot begins driving to the lounge.

Figure 6: Controlling the robot's plan execution.

so that the commander does not need to specify them again for each set of commands. The standing order of informing the commander when a hostage is seen was still valid and was carried over into a new plan, as shown by the notification upon seeing a hostage (line 21). In addition, a new standing order to defuse bombs was given now that the robot has the bomb defuser. As the robot searched the lab and recognized the hostage, it notified the commander and then waited for new orders after its search was complete. The commander provided new orders to send it to the office (line 23), where it performed a search, pausing to defuse a bomb, and then completed its search of the room. The ability to remain reactive to the environment while performing actions is an important feature of the planner in a system that operates in dangerous environments where immediate reaction is critical.

In Figure 6, we give examples of how the commander was able to interrupt plans currently being executed and change the robot's plan. The robot can be told to stop in mid-plan execution (line 05), in which case the plan is discarded. If orders have been given but the commander changes his or her mind before telling the robot to execute the commands, the commander may clear the list of orders by giving a command such as *nevermind* (line 09).

5.2 Frontier-based Exploration

In Figure 7, we show an interaction between the commander and the robot when the robot was used to explore an unknown area. The robot was asked to inform the commander about hostages and ordered to begin constructing a map of the floor (lines 01-04). However, once the commander noticed a bomb appear on the operator interface, the commander gave an additional standing order to defuse all bombs and ordered the robot to immediately create a new plan (lines 07-09). This caused the robot to pause exploring to defuse the bomb. The robot then returned to exploring once defusing was complete. The commander then assigned the name "hostage room" to the room the hostage was located in using the tablet interface, allowing future language interactions to use that name for that region.

01 **Commander:** Search the floor.
02 **Robot:** Got it. I'll search the floor.
03 **C:** Tell me if you see hostages.
04 **R:** I'll let you know if I see a hostage.
05 **C:** Make it so.
06 **R:** Understood. I'm carrying out your orders now.
The robot begins building a map of the floor. As it does, it senses a bomb, which appears on the commander's interface.
07 **C:** Defuse all the bombs.
08 **R:** Got it. I'll defuse all bombs.
09 **C:** Make it so.
10 **R:** Understood. I'm carrying out your orders now.
As the bomb is still active, the robot's new orders require it to defuse it before continuing exploring.
11 **R:** I'm now going to defuse in the 62500.0.
The robot refers to the current room by the arbitrary label assigned to it during map discovery. The robot finishes defusing and continues exploring the floor.
12 **R:** I see a hostage.
Commander renames the current room "hostage room" using the tablet interface.

Figure 7: Controlling the robot during frontier-based exploration.

6 Conclusions and Future Work

This paper presents a modular, end-to-end platform for research in human-robot interaction and a versatile test bed for evaluating improvements to individual modules. The deep integration of natural language into the system's interaction before, during, and after plan execution allows for the operator to stay in-the-loop with the system without having to continuously monitor the robot's actions. This allows for an operator to be notified of important events as they occur during plan execution as they would if another human were executing the plan. The use of general purpose natural language processing systems with limited reliance on domain-specific language specifications allow for the system to easily be extended to other domains. Such a platform can provide the capability for data collection to better understand robot-directed speech and serve as a mechanism for evaluating how operators adapt to the language capabilities of a system when the boundaries are discovered through interaction rather than specified as a system grammar.

Future work should extend the robot's language capabilities so that it can respond to queries about its actions and goals and participate in group briefings so it can help multiple teammates while executing its own plans. Additionally, future work could expand the system's ability to prevent and recover from errors that can occur during planning and execution.

7 Acknowledgements

This work was supported by Army Research Office MURI grant W911NF-07-1-0216.

References

Bikel, D. 2004. Intricacies of Collins' parsing model. *Computational Linguistics* 30(4):479–511.

Brooks, D.; Shultz, A.; Desai, M.; Kovac, P.; and Yanco, H. 2010. Towards State Summarization for Autonomous Robots. In *Proc. of the AAAI Fall Symposium on Dialog with Robots*.

Cantrell, R.; Scheutz, M.; Schermerhorn, P.; and Wu, X. 2010. Robust spoken instruction understanding for HRI. In *Proceedings of the 2010 Human-Robot Interaction Conference*.

Chernova, S.; DePalma, N.; Morant, E.; and Breazeal, C. 2011. Crowdsourcing human-robot interaction: Application from virtual to physical worlds. In *RO-MAN, 2011 IEEE*, 21 –26.

Clark, H. 1981. Definite reference and mutual knowledge. In *Elements of Discourse Understanding*. Cambridge University Press. 10–63.

Dzifcak, J.; Scheutz, M.; Baral, C.; and Schermerhorn, P. 2009. 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 *Proceedings of the 2009 IEEE International Conference on Robotics and Automation (ICRA '09)*.

Fabrizi, E., and Saffiotti, A. 2000. Extracting topology-based maps from gridmaps. *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on* 3:2972–2978 vol. 3.

Finucane, C.; Jing, G.; and Kress-Gazit, H. 2010. Ltlmop: Experimenting with language, temporal logic and robot control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1988–1993. IEEE.

Gabbard, R.; Marcus, M.; and Kulick, S. 2006. Fully parsing the penn treebank. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, 184–191. Association for Computational Linguistics.

Gat, E. 1998. Three-Layered Architectures. *AI-based Mobile Robots: Case Studies of Successful Robot Systems* 195–210. Chapter 8.

Grisetti, G.; Stachniss, C.; and Burgard, W. 2007. Improved techniques for grid mapping with rao-blackwellized particle filters. *Robotics, IEEE Transactions on* 23(1):34–46.

Jones, H., and Rock, S. 2002. Dialogue-based human-robot interaction for space construction teams. In *Aerospace Conference Proceedings, 2002. IEEE*, volume 7, 7–3645. IEEE.

Perzanowski, D.; Schultz, A.; Adams, W.; Marsh, E.; and Bugajska, M. 2001. Building a multimodal human-robot interface. *Intelligent Systems, IEEE* 16(1):16–21.

Quigley, M.; Gerkey, B.; Conley, K.; Faust, J.; Foote, T.; Leibs, J.; Berger, E.; Wheeler, R.; and Ng, A. 2009. ROS: an open-source Robot Operating System. *ICRA Workshop on Open Source Software*.

Ratnaparkhi, A., et al. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the conference on empirical methods in natural language processing*, volume 1, 133–142.

Schuler, K. 2005. *VerbNet: A broad-coverage, comprehensive verb lexicon*. Ph.D. Dissertation, University of Pennsylvania.

Shah, J.; Wiken, J.; Williams, B.; and Breazeal, C. 2011. Improved human-robot team performance using Chaski, a human-inspired plan execution system. In *Proceedings of the 6th international conference on Human-robot interaction*, 29–36. ACM.

Talamadupula, K.; Kambhampati, S.; Schermerhorn, P.; Benton, J.; and Scheutz, M. 2011. Planning for human-robot teaming. In *ICAPS 2011 Workshop on Scheduling and Planning Applications*.

Teixeira, A.; Lopes, L.; Ferreira, L.; Soares, P.; and Rodrigues, M. 2003. Recent developments on the spoken language human-robot interface of the robot carl. *Robotica*.

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

Yamauchi, B. 1997. A frontier-based approach for autonomous exploration. *Computational Intelligence in Robotics and Automation, 1997. CIRA'97., Proceedings., 1997 IEEE International Symposium on* 146–151.