Exploring Spoken Dialog Interaction in Human-Robot Teams

Matthew Marge, Aasish Pappu, Benjamin Frisch, Thomas K. Harris, and Alexander I. Rudnicky

Abstract—**We describe TeamTalk: A human-robot interface capable of interpreting spoken dialog interactions between humans and robots in consistent real-world and virtual-world scenarios. The system is used in real environments by human-robot teams to perform tasks associated with treasure hunting. In order to conduct research exploring spoken human-robot interaction, we have developed a virtual platform using USARSim. We describe the system, its use as a high-fidelity simulator with USARSim, and current experiments that benefit from a simulated environment and that would be difficult to implement in real-world scenarios.**

I. Introduction

An enduring challenge in human-robot interaction is creating interfaces that are both effective, in that proper behaviors occur, and natural, in that humans can interact with robots on a level closer to goals at hand. We are interested in the scenario where humans need to interact with multiple robots at the same time, a situation that particularly stresses the need for effective communication. At the core of the problem is the management of human-robot teams and sub-teams, where each has different roles and responsibilities. Spoken language has the potential to reduce the complexity of this interaction, by allowing humans and robots to communicate on a more abstract task level rather than in terms of a more structured operator/device relationship.

This paper describes TeamTalk: a human-robot interface capable of interpreting spoken dialog interactions between humans and robots in virtual and real-world spaces [1]. This system currently manages goal-oriented dialog in a search domain (which we will refer to as the "Treasure Hunt"). TeamTalk was developed using the Olympus architecture for the dialog interface [2]. The system incorporates live map updates in virtual and real environments and allows the execution of complex action sequences, called "plays". The PlayManager component is taken from RoboCup-related work [3]. Plays can be either individual or team-based actions, such as searching an area for victims or treasures. Low-level task allocation is managed via the TraderBot component [4]. These components were integrated into a coherent single application as part of the TeamTalk system [5]. More details on these components can be found in Section IV.

The complexity of the full system (including major software components developed in separate sub-projects, as well as 3-4 robots) presented a logistical challenge that led us to implement a simulation-based analog that would allow us to experiment with the interaction component. Development and testing was facilitated through the use of USARSim as a virtual testing platform [6]. USARSim is a simulation platform designed for evaluating and conducting research with urban search-and-rescue robots. The simulated environment and robot models are rendered with the Unreal Tournament 2004 game engine. We run simulated robots using MOAST's "SIMware" software, which serves as a replacement for real-world robotics hardware [7]. MOAST is a mobile robot framework that interfaces directly with USARSim.

Simulation allows us to make the development process more streamlined. At the same time, we maintained the same interface layer that is used for the real system, allowing us to move between the two environments with relative ease. Moreover, we designed the virtual world to mirror the environment in which the full system was being run.

TeamTalk with USARSim has also served as a robust testbed for conducting user studies that explore linguistic aspects of human-robot dialog. This virtual platform has a relatively low development and maintenance cost as compared to using actual robots. Current research focuses on multi-participant human-robot dialog, and on the interpretation of spatial language in human-robot dialog. Practical benefits of the virtual platform include the ability to perform user studies without incurring the logistical costs of a real-world full trial, and also reducing the chance that studies will experience technical difficulties. With TeamTalk interfaced to the virtual platform, we have collected several datasets of real-time spoken language interactions with virtual robots. By varying the scenario and mode of interaction, this platform permits us to explore what people say when speaking to robots in the context of goal-oriented tasks (e.g., exploring a room, moving to another teammate's location).

The organization of this paper is as follows. Section II elaborates on previous work with speech interfaces in USARSim. Section III describes the Treasure Hunt task. TeamTalk system features are discussed in Section IV. The

Manuscript received July 16, 2009. This work was supported in part by the Boeing Company Grant CMU-BA-GTA-1. The content of this paper does not necessarily reflect positions or policies of the Boeing Company and no official endorsement should be inferred.

Matthew Marge, Aasish Pappu, and Alexander I. Rudnicky are with the Language Technologies Institute at Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: mrmarge@cs.cmu.edu, aasish@cs.cmu.edu, air@cs.cmu.edu).

Benjamin Frisch is currently in the Computer Science Department at University of Wisconsin-Madison, Madison, WI 53706 USA (email: bfrisch@wisc.edu).

Thomas K. Harris is currently with EDalytics, LLC, Pittsburgh, PA 15217 USA (email: thomas@edalytics.com).

TeamTalk architecture, including its connection to the virtual platform, is described in Section V. Research projects making use of the virtual platform are presented in Section VI. We offer concluding thoughts in Section VII.

II. Related Work

Speech interfaces for USARSim-based systems have been previously explored. For example, a speech interface has been proposed for the airport tug domain, where robot tugs holding cargo could be commanded to move to different locations [8]. USARSim robots were used in simulation as part of a preliminary testing phase for cargo-moving tasks.

Other speech-based simulations have robots working as assistants to people. With the LiSA platform, robots transport small lab equipment around well-structured, delicate biological lab settings [9]. LiSA-based robots are interfaced through speech and touchscreen interactions. USARSim has been used as a simulator for their platform, but the interaction was designed to be between just a single robot and a single human.

"Smart home" robots have been developed using the Agent Development Environment (ADE), where agents such as robots share room-related information about the user [10]. Natural language dialog is made possible in ADE through the use of the SmartKom system [11]. ADE has used USARSim as part of a research platform that explores differences between human-robot interactions with virtual robots and with real-world robots.

TeamTalk extends the use of speech with USARSim by addressing communications within teams of robots and humans performing goal-oriented tasks.

III. The Treasure Hunt Domain

In the Treasure Hunt, a team consisting of robots and humans is tasked with locating "treasures" (color-coded objects) that are scattered throughout an indoor area. At the start of the task, the treasure locations are unknown and the area unexplored. In addition, team members may have roles assigned before the start of the task, based on their capabilities, such as exploration or treasure retrieval (the act of picking up a treasure and returning it to base). Teams may or may not have knowledge about the number of treasures in the environment, and are not aware of the hazards associated with the area. This makes the exploration task most appropriate for robot teammates. At any time, a human may query a team member's location and status. A robot team member would respond with its current assigned task, if any, and location (e.g., "Alphie here. I am moving to home.").

When a teammate locates a treasure, robots become aware of the location through internal communication while humans may view the treasure on their live map interface (displayed on a tablet computer). The Treasure Hunt GUI is shown in Figure 1. Currently, the treasure retrieval task is performed by a sub-team of a human (operating from a distance of about ten meters away from the remainder of the team), a navigation robot, and a robot specialized in locating

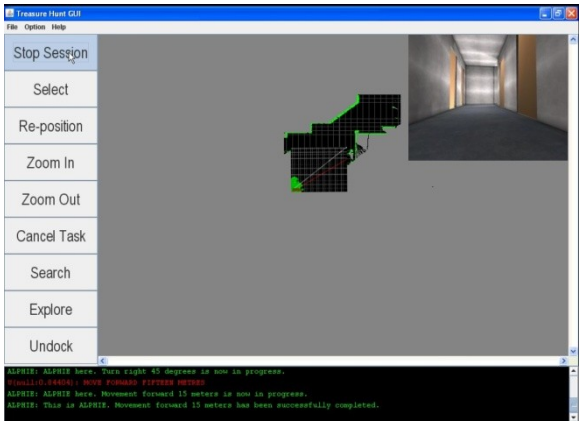

Fig. 1. The Treasure Hunt GUI, with a USARSim hallway overlaid in the upper-right corner.

the treasure. The team carrying a treasure must return to the home location to complete the task.

Humans have several roles: they must manage high-level goals to teams, query robot locations, and decide how robots should perform search tasks. At any time, a human can stop a current task or reassign a robot to a new task. Also, humans may also perform team-based subtasks, especially if they are most fit for the job, such as picking up a treasure from a safe location.

Robots have roles in the Treasure Hunt domain based on their capabilities. In both explorer and retrieval teams, Pioneer P2-DX robots traverse the area and collaboratively build an occupancy grid (using SICK laser range-finders) that notes obstructions, hallways, and doors. Currently, the Treasure Hunt scenario has Pioneers traverse a large open area used by several different robotics projects and thus a topography that varies over time. A low-level obstacle avoidance system prevents robots from colliding with other robots or with obstructions. The occupancy grid that teams of Pioneers build collaboratively is displayed on each human's Treasure Hunt GUI. This environment has been replicated in the TeamTalk virtual system using USARSim and Unreal Tournament map-building tools.

The treasure-hunting includes at least one Segway Robotic Mobility Platform (RMP). This robot is mounted with a high-resolution camera that allows it to follow a Pioneer robot and to spot treasures. As the Segway follows an exploring Pioneer, it shares its high-resolution images with the humans via the Treasure Hunt GUI. When a Segway locates a treasure, it notifies the team in two ways. First, it announces that a treasure has been found, via speech; the Treasure Hunt GUI also displays the treasure on a map. All of these roles are replicated in the USARSim-based virtual system.

IV. System Features

TeamTalk, the human-robot interface, is capable of interpreting speech, mouse clicks, and pen gestures from users. Although it is designed for tablet PCs, it may be run on any Windows-based platform. In the Treasure Hunt task, a human user operates TeamTalk with a tablet PC and an attached headset microphone. The user can instruct robots

and robot teams with speech or pen-based gestures. As an alternative, TeamTalk interprets typed text as a substitute for speech.

TeamTalk also displays a live representation of the robots' environment via its GUI. This includes an occupancy grid that is updated using information generated by the Pioneer robots as they traverse the environment. In the virtual platform, USARSim range scanner sensors are attached to the Pioneers and have access to the ground truth map. A live feed of high-resolution images from the Segway robot is also displayed on the GUI; this is replaced by USARSim images in the virtual environment. TeamTalk also displays the status of all robots involved in the task, along with a trace of the conversation history from the start of the task. Robot status information displayed on the GUI includes the robots' locations and orientations to the best of their knowledge and their current task assignment.

A user can instruct a robot or team of robots with varying levels of detail. At the high end of commands, robots can move to named locations (such as their starting point), explore an area, and search an area for treasure. In the exploring task, Pioneers traverse an area specified by pen gestures on the occupancy grid. As they traverse the area, the TeamTalk map is updated to account for obstacles, open spaces, and moving objects. The search task has Pioneer robots explore the area, with at least one Segway following, looking for treasure in the environment.

Robots are also able to process low-level "turn-by-turn" commands. At any time, a human can ask a robot to move a specified number of meters in any relative direction or cardinal direction (e.g., "Move forward five meters"). A human can also have a robot turn a specified number of degrees in any relative direction or cardinal direction (e.g., "Turn right ninety degrees"). TeamTalk is also capable of interpreting a natural language combination of moves and turns together in a single command. These may be used to get a robot out of a difficult situation or to navigate a delicate environment.

Consider a task where a robot must process a series of instructions. Such scenarios demand a robot's understanding of task division and input requirements for each component subtask. The robot instantiates a conversation to receive inputs from the user and acknowledge the in-progress subtask. If more information is necessary, such as if a user simply asked a robot to "move forward" without specifying a number of meters, the robot can prompt for this information.

We call this plan of tasks a "play," a term borrowed from the RoboCup domain. As defined in [12], a *play*, P, is a fixed team plan which consists of a set of applicability conditions, termination conditions, and N roles, one for each team member. Each role defines a sequence of tactics $\{T_1, T_2 \ldots\}$ and associated parameters to be performed by that role in the ordered sequence. Assignment of roles to team members is performed dynamically at run time. Upon role assignment, each robot is assigned its tactic T_i to execute from the current step of the sequence for that role. Tactics, therefore, form the action primitives for plays that influence

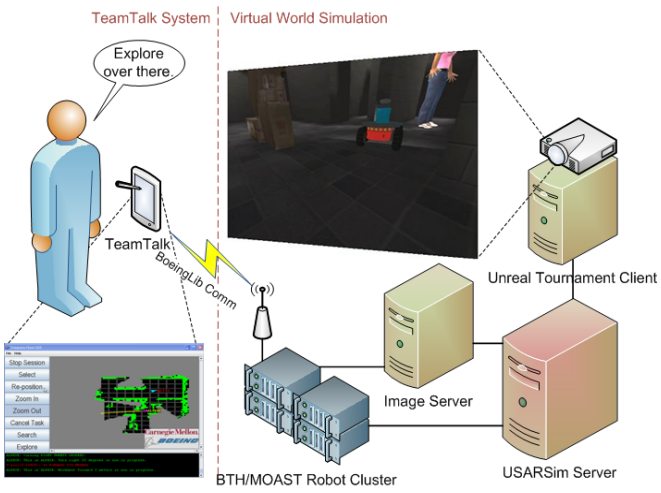

Fig. 2. Overview of TeamTalk with USARSim

the surrounding environment. PlayManager permits robots to perform these "plays" [13].

TraderBot is a low-level robot team management system that dynamically assigns roles in plays to different robots, using an auction mechanism. For example, if the task is to explore a specified area, robots in the team "bid" on the task, with the highest bidder being the one that is closest to the goal location and relatively idle. This low-level management is not directly controlled by the human user (nor is it intended to be). In principle the TraderBot mechanism allows tasks to be automatically rebid in case a team member drops out, and more generally permits robots to dynamically adjust their level of autonomy according to the given task [13]. TeamTalk, TraderBot, and PlayManager communicate directly with USARSim and MOAST components in the virtual system.

V. System Architecture

The TeamTalk system consists of several major sub-systems, which we now describe. TeamTalk coupled with the virtual system is no different than TeamTalk in real environments since the MOAST robots have been extended by incorporating the TraderBot and PlayManager components. The MOAST robots have been extended to use the BoeingLib communications protocol.

A. Treasure Hunt Multimodal Interface

The front end for TeamTalk handles user input and displays the status of each robot involved in the Treasure Hunt task. The Treasure Hunt GUI displays all the controls necessary to manage the robot team. It also displays an occupancy grid of the robots' shared representation of the environment, the locations of the robots, and the recent conversation history. Commands may either be spoken through a headset microphone connected to the computer running the Treasure Hunt GUI or by typing directly into the GUI itself. Upon initialization, the Treasure Hunt GUI reads a configuration file that specifies the IP addresses of the robots (either real or virtual) and the map server. Once the session has been initiated, the robots involved in the task report their status and are ready to begin the Treasure Hunt

task. As part of the Olympus Spoken Dialog Framework, the Treasure Hunt GUI uses the Galaxy Communicator architecture for message communication with Olympus-based dialog components.

B. Olympus Spoken Dialog Framework

This subsection elaborates on the underlying spoken dialog framework as shown in Figure 3. The speech component is implemented using Olympus [2]. It consists of all the processes that are necessary to maintain a task-based dialog with a human user. Among the components involved are those that keep track of conversation state, record speech, decode it (using Carnegie Mellon's PocketSphinx speech recognition engine), annotate speech for confidence, parse the input to extract its semantics, generate language and produce synthesized speech or display elements.

Olympus contains several components that handle the acquisition of user input. An audio server performs voice activity detection, acquires user input, sends the acoustic data to the PocketSphinx decoder, and collects recognition results. The Logios language model-building component is used to automatically create a dictionary and language model based on the grammar specified by the developer [14]. The Phoenix server parses decoded speech using a context-free grammar.

TeamTalk contains multiple domain-specific instances of dialog managers, each associated with a robot. Each robot internalizes its own conversation state processes with an instance of the RavenClaw dialog manager [15]. This allows for multi-agent interaction. A *dialog manager* models the state of the conversation between interlocutors at each step of the interaction and tracks the next task that the robot should perform. For the Treasure Hunt domain, we declare a number of *agencies*, or tasks, that a robot can perform. A robot's capabilities are incorporated into a dialog manager, including the ability to report a location, move to a destination, or turn, or undertake an activity such as search. These abilities are hierarchically organized into tasks and subtasks.

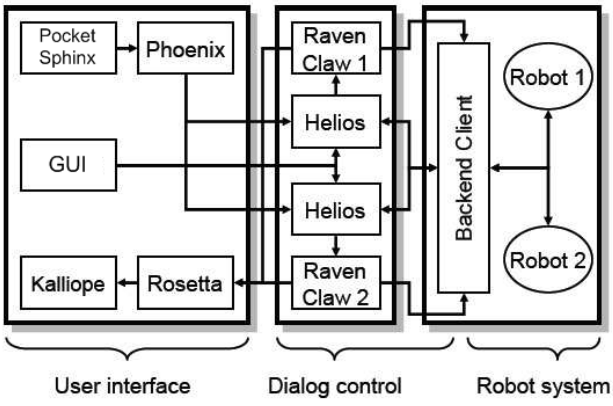

Fig. 3. TeamTalk system architecture, which uses the Olympus Spoken Dialog Framework.

Every task that is assigned to a robot has certain requirements and prerequisite conditions that need to be fulfilled [15]. For example, a robot may need to move a certain distance before it can turn to the right. Such requirements are part of the dialog task. A *dialog task* is an ordered list of agents that is used to dispatch inputs to appropriate subagents in the task. An *agent* is a particular type of handler to the ongoing conversation. For instance, a REQUEST agent asks and listens for certain user inputs that are relevant for the current task. Similarly, an INFORM agent generates follow-up prompts as an acknowledgement to user input. User input is bound to agree with a *concept type* (e.g., a `yes_no` question is bounded to a Boolean concept type; a `where_is`-type request is bounded to a string concept type). Additionally, a RavenClaw instance can handle complex concept types like arrays, frames and structures that are necessary when we are not informed about the size of list-type inputs.

Besides syntax-based bounding brought about by the speech recognizer's language model, RavenClaw allows us to restrict the scope of user input to certain semantic concepts using grammar mappings. Grammar mapping binds a particular user input as an instance of a semantic concept. For instance, if a user responded that his name is *John*, RavenClaw's grammar mapping binds user input to the semantic concept *name.* It is necessary to add an entry in the system vocabulary for the name *John* under the concept *name.* The TeamTalk backend is responsible for communicating with the robots involved in the task. When the dialog manager decides on the course of action that a robot should take, TeamTalk passes that message to the backend. The communication typically involves high- or low-level task instructions from the user. Messages are passed between TeamTalk and robot teammates via Boeing's communication libraries.

At any point in the dialog, a robot may have a concept it needs to convey to the user. This happens most often when a RavenClaw instance decides on the next task to perform. Rosetta-based natural language generation produces natural language text from RavenClaw dialog concepts. This component is customized for the Treasure Hunt domain, and consists of a set of templates with variables. We believe that this can be adapted to other domains, as the range of communications that a robot needs to convey is limited. Once natural language text is produced by Rosetta, Kalliope, the text-to-speech controller, synthesizes the text into speech and plays it in the user's headset. All components involved in the spoken dialog interaction are integrated in the Olympus architecture.

C. USARSim Configuration

An overview of USARSim integration with TeamTalk is shown in Figure 2. While robots are part of a treasure hunt rather than an urban search-and-rescue, we were able to modify the existing victim object (USARVictim) to be displayed as a color-coded treasure, which may be displayed in the virtual system. The virtual human could then pick up the treasure and return the item to home base.

To simplify development of the TeamTalk component, we developed a map simulating the actual environment to use with USARSim. Digital blueprints from the CMU Robotics Institute High Bay along with pictures with the locations of windows, stairs, bridges, and tables were used to create the

map. An example replication is shown in Figure 4. We measured each wall from the blueprint and created the proper areas, appropriately scaled to the USARSim environment. We were able to use the already-existing models in Unreal Tournament to add vehicles and stairs, which we also scaled to fit the map. The map also includes the stairs to the walkway on the second floor of the High Bay in addition to all of the rooms on the High Bay level of the building, providing us with an additional environment for testing. We also included an area representing the exterior in order to leave open the possibility of exploring outdoor scenarios. We found the tutorials on map development that came with the Unreal Tournament 2004 Editors Choice Edition (UT) to be sufficient to create the map. We also found that even though the map has many blocks and was slow to render, UT was able to display it at normal speeds on machines satisfying the typical UT system requirements (Pentium 4 processor, video card with 128MB of memory).

By default, USARSim starts the server in spectator-only mode. We modified the default game configuration (USARGame) and created a customized TreasureHuntGame which inserts an instance of a character into the simulation. A subclass of UnrealPawn, UT's player scripting class, was created that enlarged the human player mesh to the appropriate scale in USARSim and set the player's identification texture. Each time a human joins the environment, the TreasureHuntGame spawns a virtual character that has one of five uniform colors in a rotating order. A human player is necessary to open gates, doors, or the garage for the robots. The doors remain open for about five seconds to simulate real-world actions, but this length of time is configurable. The robots can then be asked to proceed through the opening. To make the player in the environment more appropriate for this task, we removed weapons attached to the player and in the environment. There is work in progress to add a tablet PC to the virtual character's hands. We found character creation to be a challenge due to the need to replicate human-like movements; most characters available on the web are creatures suitable for gaming and not for our kind of work. A repository of reasonable human figures would be an asset to USARSim-based research.

D. MOAST Integration

MOAST is configured to launch a user-configurable number of USARSim P2-DX robots inside of the simulated High Bay as a default UT Map. We are able to then connect a UT client to the USARSim Server to observe the robots, which are controlled by TeamTalk's connection to MOAST. We modified the USARSim game to load the virtual character model at an appropriate scale relative to the map. As a result, we are able to explore the room with the robots like a human teammate in the real-world scenario. We have also started work on a model of the Segway RMP. The USARSim manual clearly explained how to create a new robot, and the physics engine was able to keep the robot always upright.

Since the Treasure Hunt task typically requires the use of several robots, system resources can be demanding, even in virtual simulations. Initially, TeamTalk's MOAST configuration required one robot per computer. This proved ineffective as the number of robots outpaced the number of computers we had available. Currently, we use a single computer running VMWare, allowing us to instantiate multiple robots, each with its own network identity, maintaining compatibility with the real robot environment. Thus far, three robots can be instantiated on our equipment. Since each human runs the USARSim game within a UT client, any number of humans can be added to the environment. Two humans have successfully been added to the environment at any given time, though experiments with more humans are part of future plans.

Fig. 4. Comparison view of the real-world environment with the USARSim environment.

VI. Current Research

A. Multi-Participant Dialog

Dialog systems capable of handling multi-participant dialog may be beneficial to human robot interactions in domains such as treasure hunting and urban search-and-rescue. In fact, they become necessary when multiple teams are involved and are working towards a common goal. Therefore, assumptions that traditionally work for single-user dialog systems will fail. It becomes necessary to construct a policy that supports the needs of dynamic and asynchronous conversation between interlocutors.

USARSim provides us a reliable and relatively inexpensive manner in which to explore this challenge. Managing such conversations requires explicit

representation of the contexts and reasons behind the current dialog. To understand the relevance of the dialog with respect to context and situation, additional information is needed. This additional information includes the topology of the environment and history of events in the task. This data can be acquired through simulation testing. With USARSim, we are free to include a series of robots and humans in the Treasure Hunt map.

As can be observed in everyday team-based tasks, the number of addressees for an utterance can vary from one teammate to many. Adding to this complexity are dialogs within sub-teams, which tends to happen in this domain. At this time, we are beginning to address this issue by studying how turn-taking processes occur in human-human dialog [16]. For instance, a person who wishes to barge into another dialog should be dealt with carefully, without disturbing the ongoing conversation.

Consider the following scenario. Alice and Robot A comprise one subteam, and Bob and robot B comprise another subteam. If Alice gives an instruction to robot B at any time, robot B has to decide whether to ignore the instruction by Alice or suspend the latest instruction by Bob to follow the command issued by Alice. A conversation policy for turn taking helps robot B in the above-mentioned decision-making problem. Table 1 presents a few simple strategies that could govern turn-taking in multi-participant dialog.

TABLE I
POTENTIAL MULTI-PARTICIPANT DIALOGUE STRATEGIES

Strategy	Comments
Current speaker chooses the next speaker	This strategy is simple, but it curbs the freedom of other participants.
FIFO styled turn-taking	This strategy follows a typical meeting-style conversation policy.
Priority-centric yet dictated by external party.	This strategy comes with an assumption of task ranking and participant ranking, so that it allows multiple possibilities for the next turn, ranging from a low profiled participant reporting a less critical task to high profiled participant reporting a highly critical task. Also, this strategy leverages on the fact that real-life conversation between people takes place with respect to a priority hierarchy among them.

As a fallback to the above turn-taking mechanisms, embodiment of the robots and humans in the USARSim environment allows them to signal their willingness to take the conversational floor. These simulated embodiments act like a fabric to the turn-taking policy at the behavioral level.

B. *Spatial language*

An advantage to using a USARSim-based version of the Treasure Hunt task is the ability to rapidly obtain language data from users. Given the robustness of simulation testing, we are currently exploring spatial perspective-taking in human-robot dialog with USARSim. We intend to learn more about how people produce spatial language in reference to members of a human-robot team and later incorporate this knowledge into the TeamTalk platform. Such commands could include, for example, “Mok [a robot] move 4 meters to the right of Aki [another robot].” This has led us to design an experiment to assess how humans give simple dialog commands in reference to members of a human-robot team. We are in the process of extending spatial language processing to objects with associated ontologies.

In the study, the goal was to learn how people commanded a robot to move to a location in the 2-dimensional world relative to another robot using spatial language (e.g., “right” “left” “around”). Participants spoke their requests into a headset microphone and their speech was transcribed. Key findings from this study dealt with the varying configurations of the two robots in each scenario and the location of the goal point for the robots. The number and type of commands people gave varied based on the goal location. "Mok” was the robot that was commanded by participants to move to a location that was near “Aki,” the second robot. We found that people generally spoke in terms of 90° (e.g., “turn right”, “turn left”) or 180° turns (e.g., “turn around”). Also, we found that the orientation of Mok, the robot that needed to move, mattered most when it was facing right. This was also when it was directly facing two of the four potential goal locations.

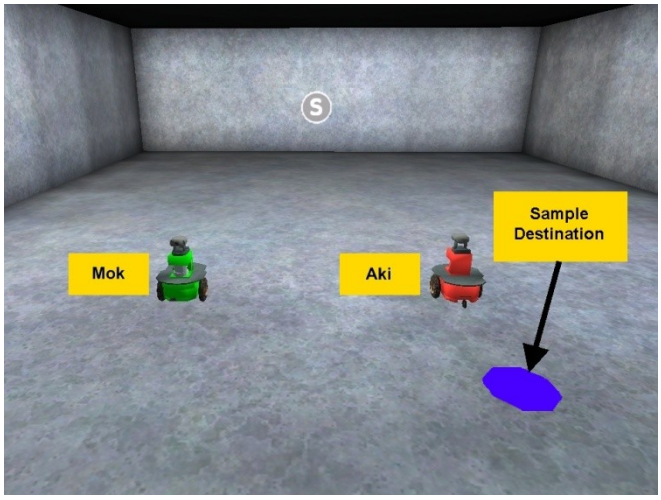

Fig. 5. Example stimuli from spatial language study. The goal destination is in purple.

Mok's other orientations required speakers to exert more effort, in terms of thinking time and the number of discrete steps spoken to move the robot to the destination. Here, we define a *discrete step* as one that causes or intends to cause a robot to move. Also, we found that Aki's orientation does not vary the commands; people treated it as a landmark. An analysis of the speech transcriptions from the study suggests that humans may use some form of internal grid to organize spatial-movement instructions. This is because participants use the length of Mok as a unit called a “step”.

Using the results from the exploratory analysis, we are currently conducting a follow-up study in a three-dimensional virtual setup with USARSim P2-DX robots, as shown in Figure 5. This required developing a USARSim

map that could permit participants to make decisions about moving Mok around a virtual environment while referring to Aki. The purpose of this second study is twofold: (1) to validate the results of the previous study and (2) to look into how the inclusion of metric units affects people's production of spatial language. We are in the process of conducting this follow-up study. The results of this study will provide us direction for developing a spatial reasoning component for TeamTalk that will be applicable to both real-world and USARSim-based situations.

VII. Conclusions and Future Work

By integrating a virtual system component into TeamTalk using USARSim, its usability as a research platform has improved. Additionally, the TeamTalk project development strategy has shifted since the integration of the USARSim virtual testing platform. Enhancements to TeamTalk are more easily tested by using the USARSim-based virtual simulation, as compared to testing with real robots. The simulation still maintains all the communication protocols that are used by robots "in the field". Furthermore, we can still elicit real-time human interactions with the virtual system. In addition, Unreal Tournament permits us to take the perspective of any robot or a bird's eye view of the scene in "Spectator Mode". We can also test outdoor scenarios with the virtual platform. Another benefit to using the virtual platform is that it facilitates collaboration with remote colleagues. Despite using the virtual system as our primary research platform, interesting results can still be validated in real-world studies, if necessary.

Future work in spoken language interaction with robots will involve using USARSim in Wizard-of-Oz (WoZ) experiments and ontology-driven robot navigation. Given the results from the spatial language acquisition studies, we anticipate developing varied forms of system responses, and testing these tuned responses in WoZ user studies. A researcher in these experiments will be controlling each robot's interactions with the user. Similarly, the aim of incorporating a Treasure Hunt ontology is to generate spatial representations that allow a human robot team to refer to objects using common sense in an environment. It is necessary to have a symbolic representation of the objects and their relationships with other objects. Conceptualization of these objects avails the opportunity for the robot to infer complex queries, such as "Face the door and walk until you find a window". Furthermore, we will assign attributes to each object to keep track of their status and update the sensory map, i.e., a robot's view of the world with the help of robot's sensory inputs. On the basis of detected objects (in the environment) and topological partitioning of the environment, a robot's knowledge about the world will be maintained.

Performing routine TeamTalk system checks is both necessary and relatively straightforward with USARSim. The transparency and flexibility of the USARSim project leads us to believe that we can expand the types of tasks associated with the TeamTalk project (e.g., collaborative problem-solving, robot learning by demonstration as done in [17]).

For more information on the TeamTalk project, please refer to the wiki, `http://wiki.speech.cs.cmu.edu/teamtalk`. The maps associated with the Treasure Hunt task are located at the TeamTalk project's Subversion repository, `http://trac.speech.cs.cmu.edu/repos/teamtalk/trunk/usarsim`.

Acknowledgments

The work described in this paper was supported by a university research grant from the Boeing Company. Bernadine Dias, Brett Browning, Brenna Argall, E. Gil Jones, Marc Zinck and Balajee Kannan collaborated with us on the overall Treasure Hunt project. We would like to thank Satanjeev Banerjee for comments on earlier drafts of this paper.

References

[1] T. K. Harris and A. I. Rudnicky, "TeamTalk: A platform for multi-human-robot dialog research in coherent real and virtual spaces," in *The Twenty-Second AAAI Conference on Artificial Intelligence*, 2007.

[2] D. Bohus, A. Raux, T. K. Harris, M. Eskanazi, and A. I. Rudnicky, "Olympus: an open-source framework for conversational spoken language interface research," in *HLT-NAACL 2007 workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technology*, 2007.

[3] E. Jones, et al., "Dynamically Formed Heterogeneous Robot Teams Performing Tightly-Coordinated Tasks," in *International Conference on Robotics and Automation*, 2006, pp. 570-575.

[4] M. B. Dias, R. M. Zlot, M. B. Zinck, J. P. Gonzalez, and A. Stentz, "A Versatile Implementation of the TraderBots Approach for Multirobot Coordination," in *International Conference on Intelligent Robots and Systems*, 2004.

[5] (2008, Dec.) Treasure Hunt Project. [Online]. http://www.cs.cmu.edu/~treasurehunt/

[6] S. Carpin, M. Lewis, J. Wang, S. Balakirsky, and C. Scrapper, "USARSim: a robot simulator for research and education," in *International Conference on Robotics and Automation*, 2007, pp. 1400-1405.

[7] S. Balakirsky, C. Scrapper, and E. Messina, "Mobility open architecture simulation and tools environment," in *International Conference on Integration of Knowledge Intensive Multi-Agent Systems*, 2005, pp. 175-180.

[8] A. M. Olney, "Multi-robot dispatch," in *International Joint Conference on Artificial Intelligence 5th workshop on Knowledge and Reasoning in Practical Dialogue Systems*, 2007, pp. 42-45.

[9] E. Schulenburg, et al., "LiSA: A Robot Assistant for Life Sciences," *Lecture Notes in Computer Science*, vol. 4667, pp. 502-505, 2007.

[10] P. Schermerhorn and M. Scheutz, "Natural Language Interactions in Distributed Networks of Smart Device," *International Journal of Semantic Computing*, vol. 2, no. 4, 2008.

[11] W. Wahlster, N. Reithinger, and A. Blocher, "SmartKom: Multimodal Communication with a Life- Like Character," in *7th European Conference on Speech Communication and Technology*, Aalborg, Denmark, 2006.

[12] B. Browning, J. Bruce, M. Bowling, and M. Veloso, "STP: Skills, tactics and plays for multi-robot control in adversarial environments," *IEEE Journal of Control and Systems Engineering*, vol. 219, pp. 33-52, 2005.

[13] M. B. Dias, et al., "Dynamically Formed Human-Robot Teams Performing Coordinated Tasks," in *AAAI Spring Symposium: To Boldly Go Where No Human-Robot Team Has Gone*, 2006.

[14] W. Ward and S. Issar, "Recent improvements in the CMU spoken language understanding system," in *ARPA Human Language Technology Workshop*, Plainsboro, NJ, 1994, pp. 213-216.

[15] D. Bohus and A. Rudnicky, "The RavenClaw dialog management framework: Architecture and systems," *Computer Speech & Language*, vol. 23, no. 3, pp. 332-361, 2009.

[16] H. Sacks, E. A. Schegloff, and G. Jefferson, "A Simplest Systematics for the Organization of Turn-Taking for Conversation," *Language*, vol. 50, no. 4, pp. 696-735, 1974.

[17] B. Argall, B. Browning, and M. Veloso, "Learning by Demonstration with Critique from a Human Teacher," in *Second Annual Conference on Human-Robot Interactions (HRI 2007)*, Washington, D.C., 2007.