Using Dialog and Human Observations to Dictate Tasks to a Learning Robot Assistant

Paul E. Rybski, Jeremy Stolarz, Kevin Yoon,
Manuela Veloso
Carnegie Mellon University
School of Computer Science
Pittsburgh, PA, 15213
{prybski,jstolarz,kmy,mmv}@cs.cmu.edu

Abstract

Robot assistants need to interact with people in a natural way in order to be accepted into people's day-to-day lives. We have been researching robot assistants with capabilities that include visually tracking humans in the environment, identifying the context in which humans carry out their activities, understanding spoken language (with a fixed vocabulary), participating in spoken dialogs to resolve ambiguities, and learning task procedures. In this paper, we describe a robot task learning algorithm in which the human explicitly and interactively instructs a series of steps to the robot through spoken language. The training algorithm fuses the robot's perception of the human with the understood speech data, maps the spoken language to robotic actions, and follows the human to gather the action applicability state information. The robot represents the acquired task as a conditional procedure and engages the human in a spoken-language dialog to fill in information that the human may have omitted.

1 Introduction

For robots to be accepted in the home and in workspaces as useful assistants or partners, we believe that people will need to be able to interact with them using spoken language and physical actions (such as demonstration) rather than solely through traditional user interfaces such as a mouse or keyboard. To this end, we are actively researching ways for robots to interact with humans and have focused our attentions on human detection, speech understanding, and dialog processing.

In this paper, we describe our work in socially assistive robotics [9] which makes use of these different research areas and describe a set of algorithms for a mobile robot that allow it to learn a task from a human. Through a combination of processing a finite set of spoken commands, observation of a human performing that task, and engaging in simple spoken language dialog, the robot both appends new instructions to and verifies the learned task sequence. Our robot uses a combination of color vision and a laser range finder to track the position and location of a person. Additionally, our robot uses speech recognition to understand verbal instructions of what must be accomplished at each location visited by the human. After the task has been verbally described and demonstrated, the robot verifies the newly acquired task by engaging the human in spoken language dialog to query any unspecified effects of conditional (branch) points in the task. In this fashion, the robot actively participates in the training process rather than passively absorbing the information.

2 Related Work

Research into the issues involved with social robotics is important for the creation of robots that will operate alongside people and integrate themselves into human environments [10]. Some successful examples of social robots include tour guides in museums [6, 27], home health care assistants for the elderly [20], and socializing robots at conferences [24]. In these studies, robots carried out a specific set of actions, for which they had to actively interact with people, but where no explicit learning of new skills took place. In this paper, we are interested in a complementary robotic capability, whereby the robots actively learn new tasks from humans by listening to their speech and monitoring their movements.

Language (spoken or signed) is a crucial communication modality for humans [7]. In our research, we have chosen to focus on the use of spoken language understanding (and synthesis) as one of the primary means for humans to communicate with the robot. Speech as a robot interaction mechanism has been studied in a number of different scenarios. For instance, Martignoni and Smart use a restrictive grammar is used for describing robot control where objects, behavioral commands, and perceptual modifiers are all mapped directly from the parsed speech [15].

We also are interested in the use of spoken language dialog as a mechanism for the robot to actively query the human about specific aspects of its task that require further explanation where dialog with a human is used for understanding human perspectives and resolving linguistic ambiguities [26]. Spoken language dialog processing has been extensively studied by the Human Computer Interaction community [21, 3, 11, 12] and this is becoming more prevalent now in the robotics communities. A number of mechanisms for supporting spoken language dialog with robots have been explored recently, including how to describe spatial relationships to robots [25].

Similar dialog-driven interaction mechanisms have been developed in the area of plan recognition, though primarily in the Human-Computer Interaction areas, as opposed to Human-Robot Interaction, domain. In Lesh et al., characteristics of the collaborative setting are exploited to reduce the amount of input required of the user [14]. This recognition strategy, however, requires some prior knowledge in the form of shared plans (or mutually-believed goals, actions, intentions) and a set of recipes (or action plans for achieving goals). This work differs from ours in that the goal is to help the user accomplish tasks according to perceived intent whereas as we are striving to teach a robot new tasks.

In Oblinger et al. an augmentation-based learning approach is described where the task structure is inferred from user demonstration [18]. In this work, manual edits can also be made to fix incorrect task structures and constrain the induction procedure on subsequent demonstrations. Again, this approach is explored in the software application domain and there is no effort to conduct a collaborative discourse with the user for natural interaction. Additionally, in our work, branching structures are explicitly and quickly communicated by the user, rather than being inferred over multiple demonstrations.

This paper describes a mechanism for actively teaching a task to a robot through the process of observing a person doing the task, as well as listening to spoken commands. Human demonstration for humanoid robotics, for instance, has been successfully employed for task learning [8] and action generation [2] respectively. Similar methods have been reported where a human teacher constructs a hierarchical description of sequences, tasks, and behaviors for robots [23].

Our work is closely related to the research of Nicolescu and Matarić, where a mobile robot observes a human doing a task and learns to associate specific behaviors with the actions that the human is performing [17]. Mechanisms for generalization of the task from the observation of multiple runs are also included. However, our approach is to focus on the use of spoken language understanding to help learn the initial task and then have the robot engage the human in active dialog to verify that the task information has been correctly transferred. Our work is also related to the plan learning/generalization research reported in [28]. Our efforts focus on learning a specific instance of a task interactively in an on-line fashion and could serve as the input source to this general off-line plan learning system.

Other work includes research in which a stationary humanoid that understands speech, though is unable to speak itself, learns tasks by communicating with the human through gestures and facial expressions [4]. We take a complementary approach in which our robot is expressionless but instead communicates understanding and/or queries through spoken language.

3 Task Training

One of the most important methods that humans have for communicating with each other is spoken language. We believe that an interesting challenge for a successful robot assistant is to allow it to be taught a task (or a set of tasks) by a human through the use of speech. When considering how we as humans might teach tasks to each other, we note that a combination of both demonstration and verbal instructions can be used. If the teacher and the learner both have shared knowledge of all of the concepts (spatial, semantic, etc...) that are referenced, then the teacher only needs to verbally describe the task sequence. However, as is more often the case, such complete shared knowledge is not available and instead a combination of both demonstration and verbal descriptions can be used in order to successfully teach a new task.

Our task training architecture supports a combination of these methodologies. In order to derive a mapping from human actions and speech to robot actions, the robot requires both a set of behaviors that will allow it to perform the tasks required of it as well as behaviors necessary for doing the learning. Additionally, the robot must be able to differentiate between those language utterances which specify specific behaviors that

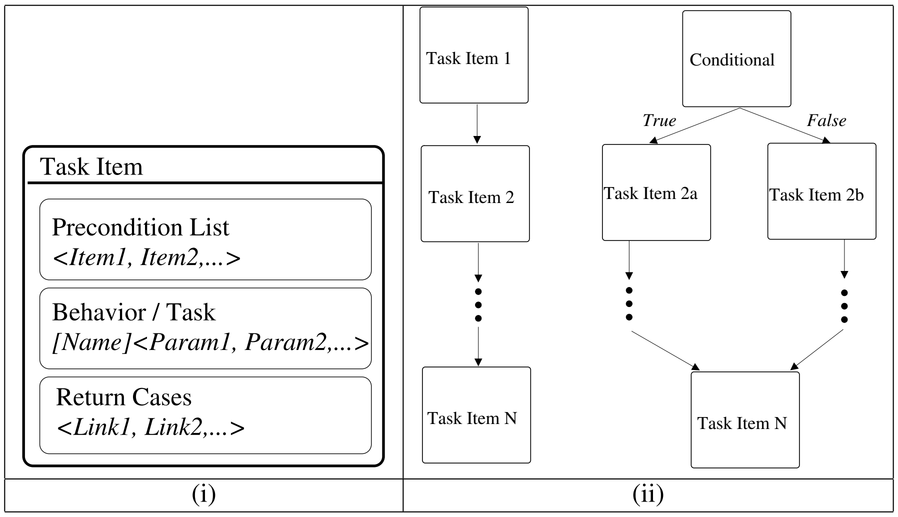


Figure 1: (i) The task item, the basic task building block. (ii) Two examples of tasks. On the left is a linear sequence and on the right is a sequence with a conditional branch.

should be executed by the robot, and those utterances which refer specifically to the structure of the task to execute.

The fundamental building block of our robot's control system are functions, referred to as *behaviors*, that map a set of inputs, including sensor information as well as derived state information, to a set of actions [1]. Internally, every behavior is defined as a finite state machine with an explicit start state and potentially multiple termination states, depending on whether the behavior was successful in achieving its goals [13]. Behaviors are responsible for relatively simple control operations such as tracking a person or navigating between waypoints. On termination, a behavior will report whether it was successful or whether it had failed. One such failure condition includes not being able to reach the goal in a timely fashion (timeout).

Behaviors are invoked by a structure called a *task item*. Task items are chained together in a directed graph structure where each link in the graph represents the transition from one task item to the next. A task item, illustrated in Figure 1(i), includes three different components: (1) A list of preconditions that must be true before that task item can be evaluated; (2) the name of a behavior or another task to be executed when the preconditions are all true; and (3) links to additional task items that will be executed based on the return status of the task's behavior. Typically, the outcomes of a particular task are either `Success` or `Failure` which is equivalent to an `If` conditional branching statement in a declarative programming language, but an arbitrary number of potential exit conditions is allowed.

A *task* consists of a set of task items that are linked in the form of a directed acyclic graph (DAG) [16], as shown in Figure 1(ii). The links that connect individual task items are directional and indicate that the first task must execute before the second. The "root" node of the graph is the initial starting state. There can be an arbitrary number of potential end states, which are specified as the leaf nodes with no additional nodes connected to them afterward. When added to the system, all tasks are given a name so that they can be referred to by having the human speak their name. Because named tasks can be referred to in the behavior/task slot of a task item, tasks can be nested in a hierarchical fashion that can be arbitrarily deep. Tasks that have been defined and exist in the robot's repertoire can be re-used in the construction of new tasks (an example of which is shown in Section 5.

A task item's preconditions must be satisfied in order for the task item's behavior to be executed. When the preconditions for a given task item do not match the perceived state of the robot's world, the behavior that is associated with that task item is not executed and the preconditions of the next task item are evaluated. The precise nature of the preconditions is domain-specific and must be pre-defined before the robot can be taught a new task. Preconditions may, for example, be based on previously commanded tasks and need not necessarily have any fundamental bearing on the robot's ability to perform the current task item. For example, assume a "goto the nursery" command is followed by "sing a lullaby." It may be the case that the first command imposes on the second the required precondition that the robot is in the nursery. This would make sense since the obvious intent is for the robot to sing only if it reaches the nursery, though its actual ability to sing a lullaby is independent of its location. In this way, preconditions can provide some protection against undesired actions when behaviors fail.

We have developed an algorithm for our robots where the robots process spoken language utterances as well as visually observe a person's motions to learn how to do a task. For a person to use our algorithm with a robot, they will need to actively initiate task learning with the robot and be aware of the robot's physical capabilities (e.g. what the robot can and cannot physically accomplish), the robot's perceptual modalities, and the subset of language that the robot understands. The sequence of operations for this algorithm includes:

1. **Training:** The human shows the robot how to do the task through a combination of demonstration and spoken instructions (see Algorithm 1).

2. **Verification:** The robot analyzes the task for completeness and asks the human for more information as needed (see Algorithm 2).

3. **Execution:** The robot executes the task (see Algorithm 3) whenever so requested by the human.

Finally, a robot that uses this algorithm must be capable of the following:

1. Detect and track a person such that it is always well-positioned to hear and observe the human.

2. Understand a subset of spoken language that lets the human describe basic commands that the robot must perform.

3. Communicate via spoken language back to the human to query them for more information as needed.

Algorithm 1 LearnTask()

```
1: Init task graph $\tau = \{\}$, which is eventually to be added to the robot's list $\beta$ of known behaviors and
   tasks.
2: Start human-tracking behavior(s)
3: **while** Training Loop active **do**
4:     $msg \leftarrow$ SpeechInput()
5:     **if** Understood($msg$) **then**
6:         $p$ = preconditions from previous action and current robot state (if any)
7:         $c$ = behavior/task necessary to satisfy $p$ (if any)
8:         Append $c$ to $\tau$
9:         **if** $msg ==$ "*Thank you*" **then**
10:            add $\tau$ to $\beta$ and exit loop
11:        **end if**
12:        **if** $msg ==$ "*Is that understood?*" **then**
13:            **if** Verify($\tau$) **then**
14:                Add $\tau$ to $\beta$ and exit loop
15:            **else**
16:                Say("*Ok, let's start again*")
17:                $\tau = \{\}$
18:            **end if**
19:        **else**
20:            Append $msg$ to $\tau$ with preconditions $p$
21:        **end if**
22:    **else**
23:        Say("*Please repeat command*")
24:    **end if**
25: **end while**
26: End human-tracking behavior(s)
```

3.1 Training

To start training the robot, the human starts the **LearnTask** behavior (Algorithm 1) by saying some invocation phrase that includes a label x, typically some imperative statement (e.g. x="dinner is ready"), that will be registered as the name for the task that is about to be trained. In our example the phrase that invokes the training procedure is "Let me show you what to do when I say..." Note that this is hard-coded and could potentially be any phrase.

Whenever the person speaks a command that is understood by the robot's grammar, the robot appends the corresponding action to the task structure along with appropriate preconditions and also prepends any actions deemed necessary to satisfy these preconditions. The preconditions are defined ahead of time based on what the robot can sense about the state of the environment as well as its own state. For instance, if the person moves to a particular location or room in the environment, the robot observes the location and the robot being in that location becomes a precondition for all tasks that follow in the training sequence until the person moves to a new location (see Section 5 for an example of this.) If the robot does not understand what was said, it notifies the user and asks to have the command repeated. When finished training, the user can either say "Thank you," ending the training behavior, or he/she can ask "Is that understood?" prompting the robot to verbally confirm the task sequence. The user then states whether the robot correctly recorded the steps, and, if it did not, repeats the sequence of steps the robot is supposed to carry out.

While giving the command sequence, the user can also make conditional statements by saying "if *y*", where y is some statement (e.g. "Kevin is present"), followed by a sequence of commands and an optional "otherwise" clause for giving instructions when the specified condition is not satisfied. Note that there is no limit to the number of tasks that can be spoken in the "if" or "otherwise" blocks. Nested "if" statements are also supported and discussed in more detail below.

Algorithm 2 Verify(Task τ)

```
1: Repeat task as it was dictated by traversing the graph $\tau$.
2: Ask("Is this correct?")
3: if msg == "yes" then
4:     for all if-nodes, $f$, with unspecified "otherwise" cases do
5:         Ask("if <f.condition> is false, I will <f.false.action>. Is this correct?")
6:         if $msg$ == "yes" then
7:             continue
8:         end if
9:         if $msg$ == "no" then
10:            Ask("What should I do when <f.condition> is false? Say done to end")
11:            while $msg$ != "done" do
12:                Append $msg$ to $\tau$ with preconditions $f.precondition$
13:            end while
14:        end if
15:    end for
16:    return True
17:    if $msg$ == "no" then
18:        return False
19:    end if
20: end if
```

3.2 Verification

A task can contain potentially many conditional branches. However, when dictating a task to a robot, only a single traversal from the start task item to an ending task item is needed at any given time. For instance, the teaching human only needs to specify what to do if any given conditional statement is true. When the robot is asked to verify the task (see Algorithm 2), it will traverse the graph and look for any conditional statements that do not have an "otherwise" (or false) condition associated with them. In this case, the robot will notify the human and inquire whether this was intentional. If the human intended more action to be taken at this point, then task training is resumed and the human can either speak a new set of commands, or teach the robot with a combination of spoken commands and demonstrated actions. This process is repeated for every "if" in the task that does not have an explicit "otherwise" case.

Tasks with very large numbers of nodes and numbers of conditional states can have their complexity mitigated by breaking them into a group of smaller tasks instead. Because task items can reference behaviors or other tasks, a group of smaller tasks could be dictated and assembled into a larger task. We will show an example of this in section 5.

3.3 Execution

To execute a learned task, the human simply utters the phrase x that was specified in the training stage and the task is invoked (see Algorithm 3). Execution of the task consists of recursively traversing the individual task items, checking whether their preconditions hold and executing their contents (a behavior or another task) if they hold true, or skipping that task item if they are false. Whether the preconditions are true or false are determined by the state of the world as perceived through the robot's sensors.

Algorithm 3 ExecuteTask(b)

```
 1: if $b$ is a Behavior then
 2:     Do($b$)
 3: end if
 4: if $b$ is a Task then
 5:     $t_{curr}$ = $b.root\_node$
 6:     while ExecuteLoop active do
 7:         if $t_{curr} ==$ NULL then
 8:             exit ExecuteLoop
 9:         end if
10:         if $t_{curr} ==$ "if <condition>" then
11:             if <condition>is satisfied then
12:                 $t_{curr}$ = $t_{curr}.true$
13:             else
14:                 $t_{curr}$ = $t_{curr}.false$
15:             end if
16:         else
17:             if $t_{curr}$ preconditions satisfied then
18:                 ExecuteTask($t_{curr}$)
19:             end if
20:             $t_{curr}$ = $t_{curr}.next$
21:         end if
22:     end while
23: end if
```

4 Robotic Implementation

We have implemented an instance of the task training algorithms described in the previous section and have conducted experiments on our CMAssist [1] robots - one of which is shown in Figure 2 - that were developed as platforms for researching the use of robots as assistants. Our robots, (shown in Figure 2), have a CAMEO [22] omnidirectional camera rig mounted on the top of their sensor mast. People are identified and tracked through the use of color histograms, similar to that described in [5]. The robots use a stereo camera for obstacle avoidance as well as to assist with tracking people. A laser range finder at the base of the robot is used for localization within a known map (trained ahead of time). Computational power is provided by two Pentium-M laptops.

Understanding of human speech is done in two parts. First, IBM ViaVoice is used for the initial capture and processing of the spoken utterances. A natural language

[1] http://www.cs.cmu.edu/∼coral/cmassist

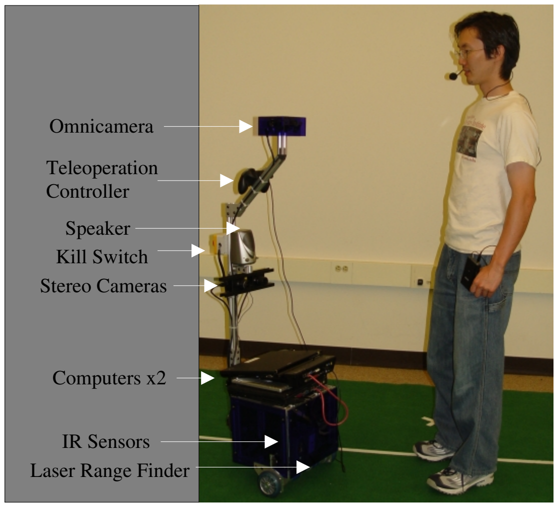


Figure 2: Our robot interacts with a person.

processing system called NAUTILUS [19], developed by the Naval Research Labs (NRL), is used to process the utterances and match them against an *a priori* grammar that represents what the robot can understand. These programs are run on a third laptop that the robots connect to wirelessly.

A list of relevant behaviors used by our robots is as follows:

- **Goto(name)** Allows the robot to navigate safely from its current position to some named location.
- **Say(s)/Ask(s,p)** Synthesizes speech and plays it through the robot's speaker. The **Say(s)** form causes the robot to speak the utterance **s**. The **Ask(s,p)** form causes the robot to find a particular person **p** and speak the utterance **s** and then wait for an appropriate response.
- **Follow(p)** Causes the robot to locate and follow person **p** at a safe distance of a few meters.
- **FollowLearnTask** Our system-specific implementation of **LearnTask** (Algorithm 1). This behavior invokes the **Follow** behavior to drive after the teacher, thus allowing it to infer locational preconditions from the teacher's position (Algorithm 1, line 2).

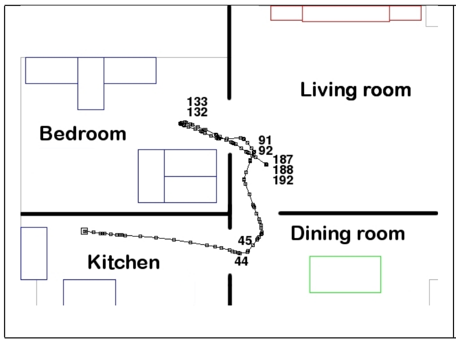
In order for the locations in the environment to be semantically meaningful as part of the training process, a map of the environment is provided to the robot which contains linguistic information regarding physical locations. For instance, the locations of named objects such as "couch", "table", and "television" can be added to the map as well as general locations of rooms such as “lab” or “living room.” This *a priori* information is used to ground locations that are either mentioned in the human’s speech or are visited as the human walks about the environment.

5 Illustrative Example

As an illustrative example, we describe how our robots are interactively taught how to do a task. In this task, the human shows the robot the steps necessary to execute the task of bringing the family together for dinner. In this example, the human that performs the training walks around the home environment with the robot following them. The individual commands that are dictated to the robot are associated with the specific room that the person is in when they give the command. In our example, the location of the person when they state the command forms the specific precondition for each of those commands. Thus the robot will attempt to reach the specific locations where it heard the command before attempting to execute it.

The task training procedure is started with the statement “Let me show you what to do when I say dinner is ready” Training is only started when the phrase uttered by the human starts with the words “Let me show you what to do when I say”. In this case, the words “dinner is ready” is the phrase registered to the task that is to be trained and correspond to the value of x in Algorithm 1. The robot responds with an affirmative “okay” and invokes **FollowLearnTask**.

5.1 Training and Verifying the Task



Task: dinner is ready

44	goto dining room
45	“tell Jeremy set the table”
91	goto living room
92	“tell Kevin come to dinner”
132	goto bedroom
133	“say turn off the television”
187	goto living room
188	“say task complete”
192	“thank you”

Figure 3: Top-down views of three different paths traversed by the robot. In all of the figures, dark lines represent walls and boxes represent furniture such as tables and shelves. The numbers next to the path indicate the locations where specific commands were issued to the robot. Numbers directly on top of each other signify the same location. This figure shows the path taken by the robot as it followed the human through the environment during task training. The larger square indicates the starting location.

Figure 3 illustrates the path that the robot took as it followed the human around the environment and learned the steps of this task. The numbers next to the different path items are timestamps (in seconds) and correspond to the locations of specific task items that the robot is to execute in those locations.

Commands explicitly stated by the human are in quotes. Unquoted commands indicate commands that are implied in order to satisfy the locational preconditions that **FollowLearnTask** assumes for all actions. That is, the robot assumes it is not to perform an action until it is at the place where the action was commanded (such as the living room or bedroom as in the example above). So when the human said "tell Jeremy set the table"—another task to be described later—in the dining room, the robot prepended the "goto dining room" command. This insertion corresponds to lines 6-8 of Algorithm 1.

In this example, two sub-tasks are being invoked. These sub-tasks were defined before the "dinner is ready" task was trained. The specific training dialog that defined these two sub-tasks is shown in Figure 4. Note how in the task "tell Kevin come to dinner", the **Verify** routine prompted the teacher for instructions for the case when Kevin is not there, whereas no further clarification was required for the other task. The task items that make up these two tasks do not have location-specific preconditions as these tasks were taught using the **LearnTask** behavior and not the **FollowLearnTask** behavior. However, note that these task items are referred to in the higher-level "dinner is ready" task which does have location-specific preconditions.

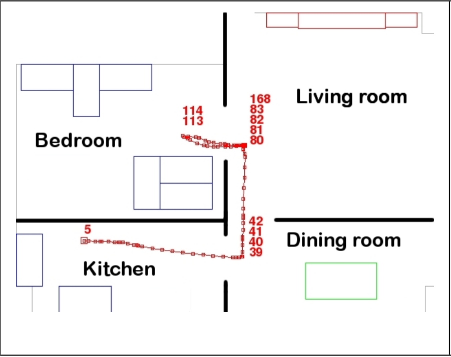
Task: tell Jeremy set the table	**Task: tell Kevin come to dinner**
"if Jeremy is present" "say Jeremy set the table" "otherwise" "say cannot find Jeremy" "Is that understood?" *<Yes, you said if Jeremy is present say set the table otherwise say cannot find Jeremy. Is this correct?>* "Yes" *<Task training complete>*	"if Kevin is present" "say Kevin come to dinner" "Is that understood? *<Yes, you said if Kevin is present say come to dinner. Is this correct?>* "Yes" *<What should I do when Kevin is present is false? Say done to end.>* "say cannot find Kevin. done" *<Task training complete>*

Figure 4: The following transcripts show the training and verification dialogs of the tasks "tell Jeremy set the table" and "tell Kevin come to dinner". What the robot says is indicated in <>.

Finally, the task item specified by the phrase "if *name* is present" is a behavior which uses the robots sensors to determine whether the named person is there. For our example, the robot knows what the people look like ahead of time by the color of their clothing.

5.2 Successful Execution

Figure 5 shows the robot successfully carrying out this task that it was taught previously. In the execution of this task, both Jeremy and Kevin were present in the



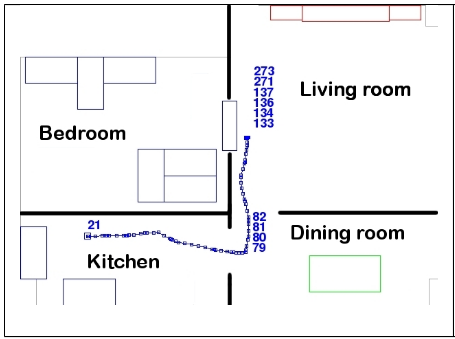
5	goto dining room
39	(starting *tell Jeremy set the table*)
40	checking to see if Jeremy is present
41	say <*set the table*>
42	goto living room
80	(starting *tell Kevin come to dinner*)
81	checking to see if Kevin is present
82	say <*come to dinner*>
83	goto bedroom
113	say <*turn off the television*>
114	goto living room
168	say <*task complete*>

Figure 5: Path taken by the robot as it traveled through the environment successfully executing the learned task.

environment and successfully detected by the robot.

Note that the "goto" behaviors are invoked when the robot must travel to that location. Thus the position in the path where these are active is the start of the traversal, whereas during training, they are recognized at the end of the traversal when the robot learns where it is supposed to go.

5.3 Unsuccessful Execution



21	goto dining room
79	(starting *tell Jeremy set the table*)
80	checking to see if Jeremy is present
81	say <*cannot find Jeremy*>
82	goto living room
133	(starting *tell Kevin come to dinner*)
134	checking to see if Kevin is present
136	say <*cannot find Kevin*>
137	goto bedroom
271	goto living room
273	say <*task complete*>

Figure 6: Path taken by the robot as it traveled through the environment and tried to execute the learned task. In this case, the bedroom was made inaccessible, preventing the robot from reaching its intended location.

In contrast, Figure 6 illustrates an example when the robot is unable to complete the task. We artificially blocked off a segment of the environment so that the robot could not reach the bedroom. Additionally, none of the people that the robot was meant to find were present.

In this case, the robot was unable to find either of the people, and, because it could not reach the bedroom (within a given timeout period), the locational precondition that the “goto bedroom” command imposed upon the subsequent task item “say turn off the television” was evaluated false causing the task item not to be executed. The execution procedure continued to traverse the task graph until it reached a task item whose preconditions were met. In this case, it was the next “goto living room” command since “goto” commands do not have locational preconditions.

5.4 Discussion

This empirical example demonstrates the task training, verification, and execution algorithms as we have instantiated them on our mobile robot. Preconditions that the robot can detect with its sensors include its location, the location of a person, and the identity of the person (pre-trained based on the color of their clothes). When dictating the task to the robot, the person walks through the environment with the robot following them. Spoken commands are associated with the location of the person when they speak them. As a result, a task that is taught to the robot is specific to this environment as well. If the robot were to change to a new environment, it would require a new map and a new training session to learn the task. The verification algorithm was demonstrated briefly in the case when the otherwise part of a conditional was not stated during the initial training. Finally, when executing the dictated task and the robot encountered an obstruction which caused a given task item’s behavior to fail, the next task item in the sequence was evaluated. Because the preconditions for these example tasks included the physical location of the robot, all behaviors that were to be executed in a specific location that could not be reached by the robot were bypassed. This demonstrates the capacity of the robot to be able to short-circuit sets of task items that would otherwise not be executable because their preconditions for execution do not hold true.

6 Summary

In this paper, we have described an algorithm for dictating tasks to a mobile robot assistant which involves the combination of spoken language understanding, dialog, and physical demonstration. This allows a human to interact with a robot in a through a subset of spoken English language in order to train it on a new task (it is assumed that the human is aware of what the robot can understand). We have developed a specific instance of this algorithm and deployed it on a real mobile robot platform. Finally, we have demonstrated how this algorithm can be used to build tasks from previously-learned sub-tasks and how the execution of the learned task is robust to failures of individual task items.

7 Acknowledgments

We would like to thank the Naval Research Labs for the NAUTILUS natural language understanding system and for their assistance in getting it to work with our robots.

References

[1] R. Arkin. Motor schema based navigation for a mobile robot: An approach to programming by behavior. In *Proceedings of the 1987 IEEE International Conference on Robotics and Automation*, pages 264–271, 1987.

[2] D. Bentivegna, C. Atkeson, and G. Cheng. Learning from observation and practice at the action generation level. In *IEEE International Conference on Humanoid Robots*, Karlsruhe and Munich, Germany, September/October 2003.

[3] A. W. Biermann, C. I. Guinn, and D. R. H. R. W. Smith. Efficient collaborative discourse: A theory and its implementation. In *Proc. ARPA Human Language Technology Workshop '93*, pages 177–182, Princeton, NJ, 1994.

[4] C. Breazeal, G. Hoffman, and A. Lockerd. Teaching and working with robots as a collaboration. In *The Third International Conference on Autonomous Agents and Multi-Agent Systems AAMAS 2004*, pages 1028–1035, New York, NY, July 2004.

[5] J. Bruce and M. Veloso. Fast and accurate vision-based pattern detection and identification. In *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, Taiwan, May 2003.

[6] W. Burgard, A. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. Experiences with an interactive museum tour-guide robot. *Artificial Intelligence*, 114(1–2):3–55, 1999.

[7] H. H. Clarke. *Using Language*. Cambridge University Press, New York, NY, 1996.

[8] M. Ehrenmass, R. Zöllner, O. Rogalla, S. Vacek, and R. Dillmann. Observation in programming by demonstration: Training and execution environment. In *IEEE International Conference on Humanoid Robots*, Karlsruhe and Munich, Germany, September/October 2003.

[9] D. J. Feil-Seifer and M. J. Matarić. Defining socially assistive robotics. In *International Conference on Rehabilitation Robotics*, pages 465–468, Chicago, IL, June–July 2005.

[10] T. Fong, I. Nourbakhsh, and K. Dautenhahn. A survey of socially interactive robots. *Robotics and Autonomous Systems*, 42:143–166, 2003.

[11] C. I. Guinn. Mechanisms for mixed-initiative human-computer collaborative discourse. In A. Joshi and M. Palmer, editors, *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, pages 278–285, San Francisco, 1996. Morgan Kaufmann Publishers.

[12] C. I. Guinn. An analysis of initiative selection in collaborative task-oriented discourse. *User Modeling and User-Adapted Interaction*, 8(3-4):255–314, 1998.

[13] S. Lenser, J. Bruce, and M. Veloso. A modular hierarchical behavior-based architecture. In A. Birk, S. Coradeschi, and S. Tadokoro, editors, *RoboCup-2001: The Fifth RoboCup Competitions and Conferences*. Springer Verlag, Berlin, 2002.

[14] N. Lesh, C. Rich, and C. Sidner. Using plan recognition in human-computer collaboration. In *Proceedings of the Seventh International Conference on User Modeling*, 1999.

[15] A. J. Martignoni and W. D. Smart. Programming robots using high-level task descriptions. In *Supervisory Control of Learning and Adaptive Systems: Papers from the 2004 AAAI Workshop*, pages 49–54, 2004.

[16] M. Nicolescu and M. Matarić. Experience-based representation construction: Learning from human and robot teachers. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 740–745, Maui, Hawaii, USA, October 2001.

[17] M. Nicolescu and M. Matarić. Natural methods for robot task learning: Instructive demonstration, generalization and practice. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems*, Melbourne, Australia, July 2003.

[18] D. Oblinger, V. Castelli, and L. Bergman. Augmentation-based learning: combining observations and user edits for programming by demonstration. In *Proceedings of the International Conference on Intelligent User Interfaces*, pages 202–209, 2006.

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

[20] J. Pineau, M. Montemerlo, M. Pollack, N. Roy, and S. Thrun. Towards robotic assistants in nursing homes: challenges and results. *Robotics and Autonomous Systems*, 42(31):271–281, March 2003.

[21] C. Rich and C. L. Sidner. COLLAGEN: A collaboration manager for software interface agents. *User Modeling and User-Adapted Interaction*, 8(3–4):315–350, 1998.

[22] P. E. Rybski, F. de la Torre, R. Patil, C. Vallespi, M. M. Veloso, and B. Browning. Cameo: The camera assisted meeting event observer. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, New Orleans, April 2004.

[23] J. Saunders, C. L. Nehaniv, and K. Dautenhahn. Teaching robots by moulding behavior and scaffolding the environment. In *Human-Robot Interaction*, Salt Lake City, Utah, March 2006.

[24] R. Simmons, D. Goldberg, A. Goode, M. Montemerlo, N. Roy, B. Sellner, C. Urmson, A. Schultz, M. Abramson, W. Adams, A. Atrash, M. Bugajska, M. Coblenz, M. MacMahon, D. Perzanowski, I. Horswill, R. Zubek, D. Kortenkamp, B. Wolfe, T. Milam, and B. Maxwell. Grace: an autonomous robot for the aaai robot challenge. *AI Magazine*, 42(2):51–72, Summer 2003.

[25] M. Skubic, D. Perzanowski, S. Blisard, A. Schultz, W. Adams, M. Bugajska, and D. Brock. Spatial language for human-robot dialogs. *IEEE Transactions on Systems, Man and Cybernetics, Part C*, 34(2):154–167, May 2004.

[26] D. Sofge, J. G. Trafton, N. Cassimatis, D. Perzanowski, M. Bugajska, W. Adams, and A. C. Schultz. Human-robot collaboration and cognition with an autonomous mobile robot. In F. Groen, N. Amato, A. Bonarini, E. Yoshida, and B. Kröse, editors, *In Proceedings of the 8th Conference on Intelligent Autonomous Systems (IAS-8)*, pages 80–87. IOS Press, March 2004.

[27] T. Willeke, C. Kunz, and I. Nourbakhsh. The history of the mobot museum robot series: An evolutionary study. In *Proceedings of FLAIRS 2001*, May 2001.

[28] E. Winner and M. Veloso. Analyzing plans with conditional effects. In *Proceedings of the Sixth International Conference on Artificial Intelligence Planning Systems*, Toulouse, France, April 2002.