TASK TRANSPARENCY IN LEARNING BY DEMONSTRATION : GAZE, POINTING, AND DIALOG

A Thesis
Presented to
The Academic Faculty

by

Nicholas Brian dePalma

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in the
College of Computing

Georgia Institute of Technology
August 2010

TASK TRANSPARENCY IN LEARNING BY DEMONSTRATION : GAZE, POINTING, AND DIALOG

Approved by:

Professor Andrea Thomaz, Advisor
School of Interactive Computing
Georgia Institute of Technology

Professor Rosa Arriaga
School of Interactive Computing
Georgia Institute of Technology

Professor Mike Stilman
Robotics and Intelligent Machines
Georgia Institute of Technology

Date Approved: July 1, 2010

This is dedicated to my mother and father who provide the most loving and supportive environment a son could ask for, and to the love of my life, Kate dePalma, whose patience and understanding absolutely means the world to me.

ACKNOWLEDGEMENTS

My thesis could not have happened without the gracious and thought-provoking conversations I've had with my advisor, Dr. Andrea Thomaz. Without her guidance, I would truly be lost.

I have also been fortunate to have some of the most patient faculty on my committee. Dr. Mike Stilman and Dr. Rosa Arriaga have truly challenged me to keep my eye on the big picture and together have helped to shape my philosophy.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

This body of work explores an emerging aspect of human-robot interaction, transparency. Socially guided machine learning has proven that highly immersive robotic behaviors have yielded better results than lesser interactive behaviors for performance and shorter training time. While other work explores this transparency in learning by demonstration using non-verbal cues to point out the importance or preference users may have towards behaviors, my work follows this argument and attempts to extend it by offering cues to the internal task representation.

What I show is that task-transparency, or the ability to connect and discuss the task in a fluent way implores the user to shape and correct the learned goal in ways that may be impossible by other present day learning by demonstration methods. Additionally, some participants are shown to prefer task-transparent robots which appear to have the ability of "introspection" in which it can modify the learned goal by other methods than just demonstration.

CHAPTER I

INTRODUCTION

Learning by demonstration that takes advantage of the natural social interaction between human and robot is a burgeoning area of study. In previous work, Thomaz and Breazeal laid the foundation for leveraging human interaction with robot transparency to further interactive machine learning [34, 33]. Here, one of the most compelling arguments about interactive machine learning with robots is made; they show that, instead of strictly taking input from the human, the robot can use gesture and social cues to inform the human about the internal state of the robot. These behaviors are *transparent* in that they indicate internal state. This is in contrast with more "opaque" behaviors. The work points out an important observation; namely that we can look to developmental learning as a hint on what kind of cues humans use to inform one another. It was shown that gesture and gaze are just a few major social cues that can be leveraged. Argyle's treatise[3] on gaze provides a great reference for how gaze can be characterized and used or leveraged. Inspired by this as well as other social cues that have been shown to be useful, such as pointing [8, 7] and natural language [29], I designed a study to explore task-transparency or ways to explicitly make available the task to the human in ways that are familiar to the teacher.

Transparency in human-robot interaction is the communication that facilitates an inference, or a way of guessing, to the internal state of the humanoid. Fundamentally an human-computer interaction term, transparency in general is the ability for a device to be so intuitive as to blend into our daily lives. In human-robot interaction, transparency has been linked to natural behaviors that facilitate interaction between human and robot. It has been shown in previous research that humans that work together share what they believe to be a shared task representation to complete the task in unison. This was studied in detail with the work of Bratman [6]. While Bratman's work focuses on shared task planning, joint activity and a commitment to jointly supporting the task; his work provides strong evidence that through commitment, humans actively facilitate the learning process by supporting the efforts of others. Further evidence that supports this but argues too of the subtle interactions between collaborators can be found in the work of Baron-Cohen [4]. Here too, pointing is identified as an important and interesting way that humans interact with one another for communicative reasons. The importance of these social cues has been argued for by others but has only recently gotten the attention it deserves in robotics. I further support this in my work and further extend the importance of it in robot task learning.

Learning by demonstration is a specific sub field of robotics that enables a dream of many roboticists to build robots that dynamically add new tasks to its repertoire. To design adaptable robots, engineers will need to consider the same environment and modalities that humans utilize and provide enough primary functionality to survive in our environment. Learning by demonstration studies are usually performed under a controlled environment in which little noise gets in the way of what is learned. Breazeal et al.[7] point out that inference is not straight forward and that many times the shared representation becomes out of sync due to sensor error or otherwise. These errors that are produced should be corrected as soon as they are discovered which motivates much of the work in robot transparency; revealing state early and often provides enough feedback so that the collaborator can correct these minor errors. My intuition and the focus of my study attempts to provide a framework such that these small deviations can be corrected quickly. By making the argument that task transparency will reveal and alleviate any sort of symbolic error early, I expected the human demonstrator to modify the internal goal state. Many may argue that a good saliency model, like the ones found in [20, 21] will help alleviate much of the noise in the demonstration. I don't disagree, but supervised learning (in the broadest sense) can always help correct implementation problems or errors in the model as long as the supervisor has modification access to the error in the representation. Making a step in this direction, I am attempting to take a different view and ask what I can make available to the supervisor to aid learning by demonstration toward becoming more accurate. To do this, I am allowing the user to directly clarify specific goals that were taught rather than specifying entirely new demonstrations or by constructing goal sequences. To achieve this, I provide the robot with a speech interface that allows the robot to explore its own representation while hypothesizing that humans will commit to modifying that representation through dialog. Following in this line of thought, I provided a transparency mechanism to allow the human to detect when something has deviated from the intended goals that the human has taught and provide a mechanism for the learned goals to be refined and fixed in a partial order plan ad hoc and on the spot.

By revealing the internal state, or more specifically the learned goal that was previously taught, I hypothesize and show that humans correct the goal and significantly improve the accuracy of the learned goal. I explore the importance of task transparency to goal accuracy during teaching. Previous work has indeed shown the subtle effects of transparency on teaching but by also explicitly making symbolic goals available to the robot to reveal at all times, I expect the accuracy of the goals post interaction to significantly improve in accuracy with respect to the intended goal. Using our robot, Simon, I set up a study to explore two different interactions that were designed to explore task transparency in further detail. I designed the interaction during the baseline interaction to elicit very little feedback from the robot, using very few, or if they were needed, the most basic social cues while the experimental group received an interaction that contained social cues that will discuss later (gaze, pointing and a basic dialog system) and show that these basic techniques will allow the human teacher to shape the goal better than the baseline interaction alone. I used two tasks for each group, cleaning up the table and building a door to remove demonstration bias. And finally, I built a framework for learning first order relations from observation. Though, these relations were grounded by an expert, myself, from the robot's sensors and used to build a representation of the task. I present the results of a user study that shows a predilection of humans to view machines that can discuss their own learned task model as more intelligent while also showing that the resulting learned task is more accurately transferred from human to machine. By providing a basic dialog system along with the first order task learning framework, task learning accuracy was shown to significantly improve over the baseline.

It is important to clarify at this point that, ethically, the goal of learning by demonstration for human-robot interaction is not to create perfect replicas of human learning in robots but to facilitate human interaction and leverage the interaction to maximize the accuracy of what is learned. This human-centered learning process is a cooperative and social activity. Thomaz[33] has called this approach Socially Guided Machine Learning or SG-ML. These theories inform the design of interactive learning systems, not by emulating biological mechanisms but by taking advantage of them to facilitate the interaction. In fact, there have been many interactive machine learning studies showing that interactive learning that takes advantage of these natural modes of communication have advantages over batch learning [38], or that transparency in active learning improves performance [10].

Task transparency having such an important role for human learning, as I hypothesize, should translate and provide better performance for robot learning. I show that this is the case, pointing to improvements in accuracy with each clarification that the human gives.

CHAPTER II

EXAMPLE SCENARIOS

In the near future, robots will soon be asked to perform minor household tasks. This may include doing the laundry, doing the dishes, or simply cleaning up. The role of robots in our society won't be restricted to the home; but also to our workplaces as robots are asked to build products along-side or in lieu of human labor. As such, I designed two scenarios to represent these types of tasks that the robot will be asked to perform in the future. The first scenario is cleaning up, wherein the robot is asked to sort, organize, and place objects in the world around it. I explore these situations in more detail as a theme throughout my thesis.

2.1 Building a Door

The first example is from a domain of one of our collaborators, General Motors™involves a task that can only be executed by humans at the moment. The basic task of building a door is very complex on the floor of major manufacturers, simultaneously requiring solutions to major unsolved problems in computer vision, precision, and manipulation. On the manufacturing floor, every part of the car is moving on an assembly line and all assembly actions are performed by a human worker on the door while it is moving. Involving robots in this task requires solutions to human-robot collaboration on the assembly floor that incorporate manipulation in a dynamic environment.

The exact sequence of events of one particular worker on the floor was presented to us by GM as particularly challenging. I explore this task in detail as a case-study for learning by demonstration, demanding that the robot learn the actions and goals that need to be satisfied on a car door by the worker before it leaves his hands on the manufacturing floor. The sequence from the workers standpoint is thus: pick up the

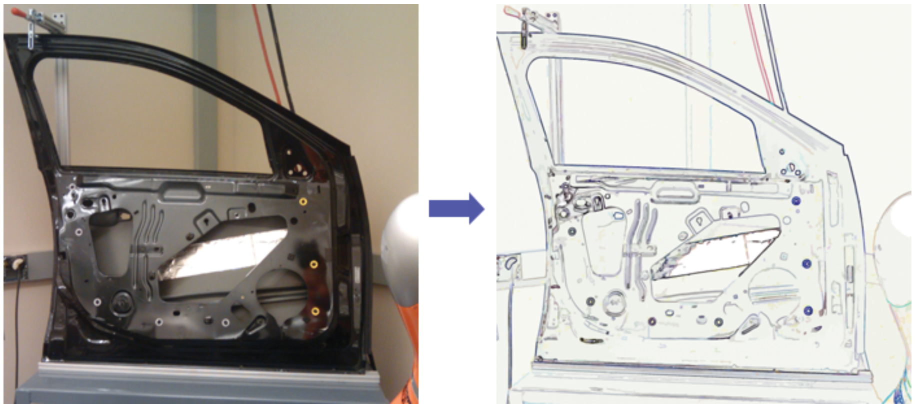

Figure 1: Creating the abstract door for the door task

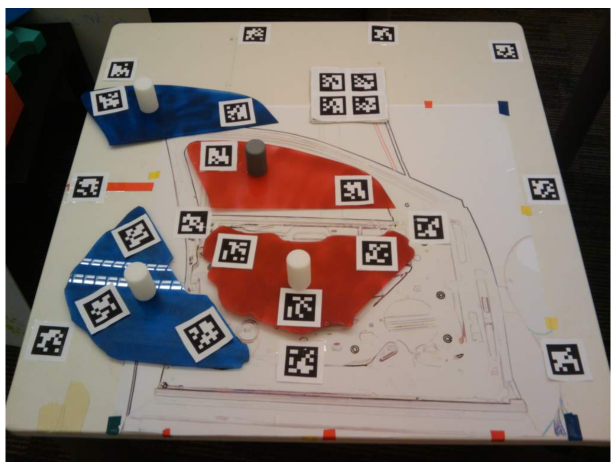

Figure 2: The “build a door” task

door lock and insert it inside the hull of the door (which is subsequently hollow) place it by feeling around for the holes, then align the door lock by peering through the screw hole in the side of the door, then bolting it from the outside. An attached cable is run through a hole in the interior face of the hull of the door. Finally, place the panel over the cable and seal the hot glue with a roller. This includes many challenges for robots including precision of manipulation in a cramped hollow door, bi-manual manipulation for holding and bolting the door lock, tracking major features on the door, and timing the seal of the panel. I simplify this task by first building an abstract door as seen in Figure 1.

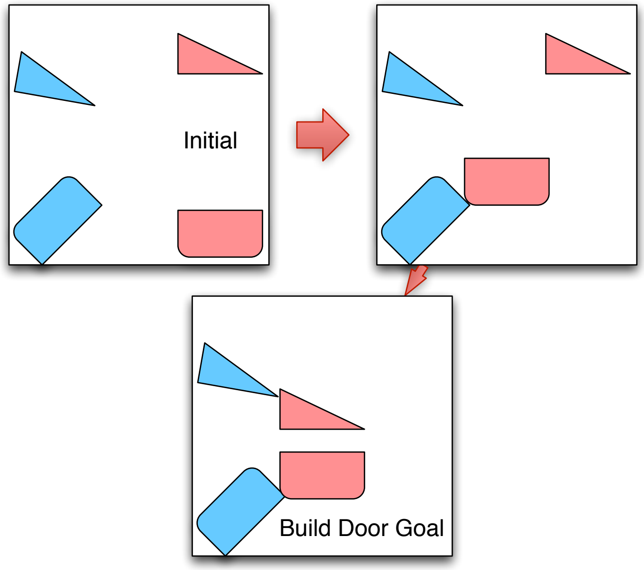

Figure 3: Step by Step solution to the build a door task

The parts in the task were Plexiglass (cut using an Epilog laser cutter) with 3D printed handles made with a Dimension 3D printer. The goal of the task was to learn how to place and orient the panel and window.

The door building task involves moving two pieces into place. One example completion is shown in Figure 3. In the presence of other parts and door features, the task was to learn that the panel is aligned to the bottom of the door and the window to the top.

The robot's objective is then to simply learn to align and place the objects correctly by demonstration. I will use this example task throughout this paper and is the center of discussion for the study as well.

2.2 Cleaning Up

For the other task that was used, I structured the task to have the human collaborators teach the robot to "clean up" which I defined simply as putting like colored objects together into small groups. Again, in this task, I abstracted the clean-up task

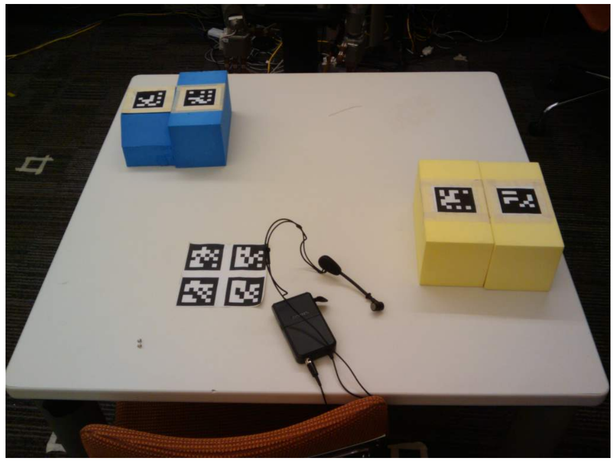

Figure 4: The “clean up” task

from a typical cleanup scenario. Cleaning up fundamentally involves understanding a few concepts: orienting, ordering, and placing. For instance, these abstractions are analogous to collecting dishes and placing them near the sink or collecting books and placing them on a book shelf, right side up in alphabetical order. I explore this task only as a placement exercise.

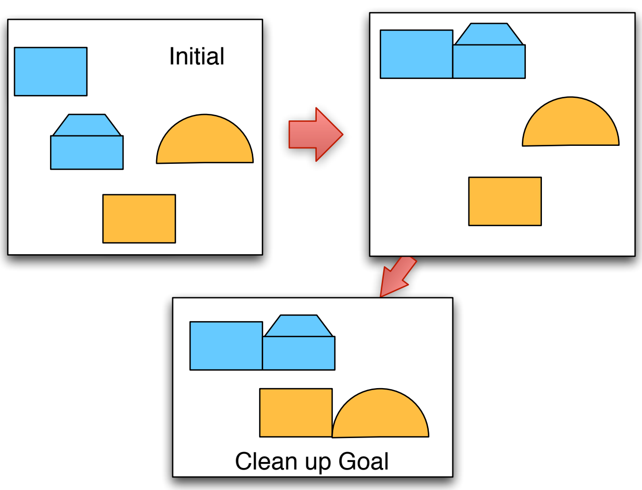

Figure 5: Step by step solution for the clean up task

The task is to place like colored objects near one another as shown in Figure 5. In this figure, the yellow arch is denoted with a half circle, yellow and blue blocks with the rectangle, and blue keystone with the polygon. The objects are arranged, by an expert, in the same placement for every user on the table. The goal of this task is to move the objects in two moves to cleanup the table by color. Completion occurs in two moves: move the yellow block next to the yellow arch, followed by the blue keystone next to the blue block in any order. Redundant or extraneous subtasks may be encountered such as having the first move, placing the blue keystone to the right of the blue block discover that the blue keystone is also above the yellow arch. This could be considered too specific an objective for that action. This is meant to be fixed in later demonstrations or through some clarification mechanism.

CHAPTER III

BACKGROUND AND RELATED WORK

My work could be placed closest to the work of Thomaz and Breazeal. Their work on establishing transparency, or the broad class of communicative acts that facilitate and aid in revealing the internal state of the robot collaborator as important to supervised learning motivates more investigation. Task-transparency is inspired by my interest in human robot interaction with respect to learning by demonstration (LbD). In this chapter, I will explore some experiments in LbD and transparency.

3.1 Transparency

> "[Transparency] would say that there is nothing in the state of the system that cannot be inferred from the display. If there are any modes, then these must have a visual indication; if there are any differences in behavior between the displayed shapes, then there must be some corresponding visual difference."
>
> -Dix [14]

Transparency is historically a term used in human-computer interaction. The term is applied towards mechanisms that allow the user to "peer" into the internal working state of the machine and provides the ability to modify that state. More specifically, some part of the internal working state is made visible by formatting or translating the internal state for the user. Transparency has become a central heuristic for users in devices and has born out its need in human computer interface design.

Donald Norman implores us to consider the emotional machine in robot design[30]:

> "... the robot should display its emotional state, much as a person does...so that the people with whom it is interacting can tell when a request is understood, when it is something easy to do, difficult to do, or perhaps even when the robot judges it to be inappropriate."

His emotional machine idea reveals something fundamental to design, that well designed devices in the future will use, what some may consider, unorthodox communicative channels. Emotional robot design is controversial. Aside from the debate is a steady stream of evidence that by leveraging our nonverbal communicatory channels, supervised interactive machine learning performance can be significantly improved. Thomaz[25, 33, 34] has demonstrated this result previously and Breazeal et. al. [7] demonstrates how fundamental these cues are for human robot teamwork. Furthermore, Thomaz further shows[33] evidence that to support the idea that robot transparency improves interaction by 1) reducing the total time spent with the robot, and 2) detecting and reducing errors.

Early work in transparency and learning by demonstration pointed out similar results. Thomaz and Breazeal[25] show increased accuracy in Q-Learning attributed to transparency. This study uses speech, gaze, and gesture to help guide the teaching process and was an early indicator of transparency's usefulness in learning by demonstration.

The broad concept of transparency is emerging as a qualitatively interesting method of using social cues to maintain synchronicity of state. It affords an affective channel that is recognized by users as a preferred method of peripheral communication. Recent work by Mutlu[27], uses one type of transparency, gaze, to establish footing in conversation and can successfully establish role in conversation. This was shown to be an effective method of manipulating the roles of the participants in the study also show that transparency can be used to internalize roles established by the robot agent.

I am interested in using transparency as a method of imploring the user to play a role as teacher. Similar to the work of Thomaz' Sophie's Kitchen experiment [34] in which participants were able to successfully utilize subtle gaze cues to provide guidance and improve performance of the virtual agent, I show performance improvement with regard to task accuracy.

3.2 Learning by Demonstration

Learning by demonstration has been surveyed fairly recently [2] and provides many important studies in the field. They define learning by demonstration as a subset of supervised learning that incorporates observing a number of demonstrations, D, from a teacher using observations of pairs in its state and action space to generate some policy. Some of the earliest work has been under the name "Programming by Demonstration", or PbD, in which a set of actions and parameters are used to construct a behaviors that can be executed arbitrarily many times. Friedrich and Dillmann[17] demonstrate a system that begins with a set of skeleton programs and macros that can be arranged through example to construct interesting behaviors. This allows the user to construct some permutable number of possible plans using a given set of actions and objects that the robot knows how to deal with. PbD, one of the original types of LbD has, as of recently, been updated to use newer forms of machine learning and vision techniques to allow teachers to construct far more complicated plans [29, 10], to teach specific actions [9, 23], and to teach the robot about specific discrete and/or continuous features about the object or the goal. Advances have also allowed the teacher to further provide generalizations about the task [13, 8]. More recent advances have incorporated many new modalities that have been afforded by modern advances in the state of the art such as speech recognition and speech synthesis for dialog, teleoperation using gestures[16], and shadow teleoperation using feedback control[19].

All of these techniques have been under the guise of one name or another, whether it be programming by demonstration, learning by demonstration, apprenticeship learning [1], or socially guided machine learning [33]. Interesting directions LbD and PbD have taken in the past range from, leveraging new technologies to incorporating more abstract concepts that have been well studied in theory of mind research, such as saliency models[12], and ontology building[17], that further allow for the robot to build a repertoire given a primitive set and to have a model of attention that resembles the human.

While much of their work has revolved around learning by demonstration, some recent work has provided more information to the user than just the execution of the learned task or behavior. The work of Mataric and Nicolescu has provided a mechanism that allows the user to understand what the robot has learned and allows the user, through execution only, to stop the robot and correct the currently executing plan by specifying changes using speech commands or new positive demonstrations that clarifies the erroneous goal of the learned task [29]. Chernova's LbD system[11] allows for the robot to ask for a completely new demonstration. Both of these implementations provide an incremental architecture to integrate newly observed demonstrations into a less precise or an erroneous task representation to build a more perfect representation that the robot intends to satisfy by executing the task.

My work extends work from Thomaz, Breazeal, and Chao [34, 33, 10, 7] in the direction of Nicolescu [29] by additionally focusing on a correction mechanism through dialog. My work stands in contrast to each of these for the following reasons: 1) *I provide a task-transparent mechanism that is specific to learned goals*, 2) *I am identifying and studying the accuracy of the learned goal*, and 3) *I am allowing the robot to modify the goals without restriction*. These objectives stand in support of the work of Nicolescu[29] whose work with dialog and partial order plans shows positive results but was focused on planning and was not explicitly considered transparent or evaluated with naive users.

CHAPTER IV

ROBOT PLATFORM

I took advantage of the Socially Intelligent Machines Lab's resources for my investigation. The specifics of what already exists are separated into the robot hardware that I used, the sensory system that I set up, and the architecture that was used to program our robot, Simon.

4.1 Robot Hardware

For my study, I have taken advantage of our robotic platform, Simon, an humanoid upper-torso robot with seven degrees of freedom per arm and four for each hand. The torso has two degrees of freedom, pitch and yaw, and a highly expressive head. Simon was designed specifically to work with humans; it was designed to be friendly, having non-rigid compliant arms and a childlike voice.

The torso and arms were designed and manufactured by Meka. The torso features two compliant arms that are safe to use around humans. The motors are built with custom series elastic actuator technology[15] run on an ethercat bus attached to a Linux real time operating system. Simon is attached to a metal pole to prevent it

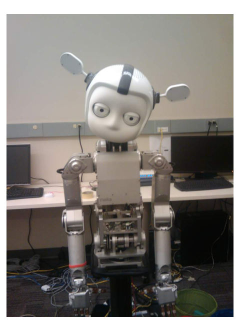

Figure 6: Simon the Robot

from falling over and to aid in calibration. The entire system runs on a custom Meka controller that maintains internal motor state for the robot.

4.2 Sensory Environment

The sensory environment for my study was a combination of ARToolkit+[22] to track objects and the Windows 7 speech API for speech recognition. All of my abstractions involved using different fiducials that are recognized using a package called ARToolkit+. ARToolkit+ uses small fiducial tags to recognize an object's identity, position, and orientation. These barcode-like tags can be seen in Figure 2 or in Figure 4 in Chapter 2. ARtoolkit+ is run using a dual camera mount hanging from the ceiling pointing at a table situated between the robot, Simon, and the participant. This provides a mechanism to map the barcode id to the color and shape of the object while also providing detail on its position and orientation.

The Windows 7 speech API software was used for recognition. It requires a grammar that includes the phrases in Figure 1 for recognition and is based on custom implementations of Microsoft software. The recognition software provides a limited vocabulary that when organized in a certain way, produces sentence tags. These tags represent the semantic meaning of each sentence recognized. For instance "Simon, forget about the blue window" will return the tag FORGET:BLUE-WINDOW. The tag is parsed into a declarative command and a referent after each spoken phrase.

I am also taking advantage of speech synthesis using the default "Junior" voice from Apple's OS-X operating system. This provides a method for Simon to communicate in an unassuming child-like voice. We are interfacing with this using a system pipe to a command line and sentences are put together using a formula that is described later in Chapter 6.

4.3 Software Cognitive Architecture

In addition, I took advantage of our software platform, c6m (based on a the creatures architecture, see [5]). This package uses an inverse kinematics package that implements CCD[37] to control the arm in the workspace; this package side-by-side with an animation system allows Simon to play a number of animations, designed in 3D CAD software, that involve the head, the arms, and the hands as well as the capability to enter into interactive modes interchangeably with playing animations. It is built into a small number of major components that can be described visually in Figure 7.

Each perception module is connected with c6m through a small subnet. c6m has a custom network stack on top of UDP called IRCP (detailed more in Hancher's MS thesis [18]). Every incoming observation becomes a set of percepts $P = \{p_1, ..., p_n\}$ where each $p \in P$ is is an atomic classification and is aggregated in a perception system to be merged with other percepts by using match values $p(o) = m$, where $m \in [0, 1]$ and o is some percept observation. These observations are merged together in the perception system to later become derived percepts.

External Modules send data to our percept system over the network. Sensory data is captured by packet handlers that are merged into percepts and added to a "percept tree" where primitive features in the world captured by external modules (i.e. vision, speech, ARToolkit) and added to the base perceptual level of the tree. For each time, t, this root percept data gets refreshed by each module and based on the structure of the tree, other percepts that may use that data are updated appropriately. These percepts are called "derived percepts" and are meta information such as "most salient" object which is a percept that contains information about about the object of attention. Other derived percepts include any sort of meta information from the lower level tags. For instance, in my study, ARToolkit+ may only send packet data about the location and the IDs of the tag but higher level knowledge such as color and shape may be mapped to these tags as a way of simplifying the vision problem.

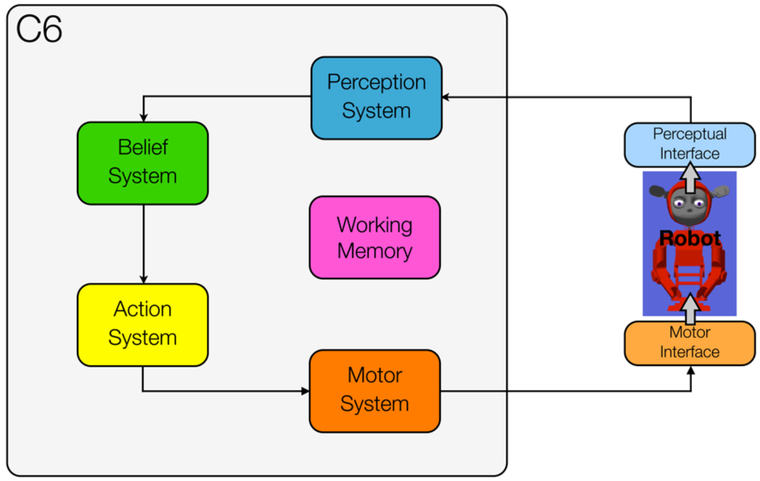

Figure 7: c6m Architecture

This meta data may understand or perceive that “ID 5” is really just “blue block” and those derived percepts are updated on the tree.

The belief system moderates a belief set B between each time frame and merges the current percept set into the previous beliefs. I wrote a number of similarity metrics to merge and aggregate percept data into beliefs appropriately and these belief objects detail the perceived state of the world classified and arranged by ARToolkit+ tag ID and type. This is usually analogous to objects in the world. For instance, a single belief may contain percepts about its color, it’s location, shape, size, and any other features that may be important to learn about. Each belief is then used to make decisions about next actions based on a set of hierarchical action tuples that require preconditions, a small set of execution parameters, and postconditions. Animations are triggered through these action tuples as well as our inverse kinematics code that is integrated as one of the interactive action tuples. After each high level action begins to run, the lower level joint trajectories are rendered into our simulated framework to be visualized. The motor system watches this data for changes and sends it to our controller, a Meka real-time operating system that manages Simon’s motors via an ethercat bus.

Learning in c6m is based on and inspired by previous and some unpublished work of Thomaz [25, 33] and others. The focus of the learning framework is on goal learning in which a number of demonstrations are given, observed and stored by the robot. Goals are represented by the consistent logical sentences (more specifically relations based percept data, see 5.1) that make up the end state for every demonstration and action. The demonstration, a capture of the belief system after every demonstration and every action at the end of the action, are merged by determining the frequency of the observed percepts in each belief and determining whether not or it is consistent with every other demonstration it has observed. If, for any reason, the objectives aren't consistent, then it is considered unimportant and thrown out as a goal condition. Algorithm 4.1 describes how this works in pseudocode. The goal is then the set of constraints put forth by the consistent perceivables for all of the demonstrations. This method was based on similar work by Thomaz[33] and Chao[10].

consistent_goals = list()
action_list $\leftarrow G_1 ... G_n$
for G_i in action_list **do**
  for relation in G_i **do**
    count = count_exist($G_1 ... G_n$)
    if $\frac{count}{n} > thresh$ **then**
      push(consistent_goal, relation)
    end if
  end for
end for

Algorithm 4.1: Determine consistent goal constraints

Algorithm 4.1 builds basic "move" actions based on the goals in each movement. By identifying the object (ARToolkit tag) that changed the most, a pseudo-action is built : move(<object>). This makes correspondence between out of order demonstrations possible. The input for this algorithm is the set of n actions goals or objectives ($G_1 ... G_n$) that are deemed to correspond. Once these are aggregated, they are sent to the algorithm (4.1).

The two previously mentioned implementations have been made that use a version space to more formally generalize. Most of my work extends this portion of the c6m.

CHAPTER V

TASK LEARNING AND PLANNING

A number of contributions were made to the framework to support the requirements in the door building task. First, I developed a relational learning subsystem for of our task learning framework so that Simon can learn about spatial relations based on the percept and belief data. I also developed an implementation of a partial order plan learner which discovers sequence constraints on the system and finally a few needed interaction components based on the relation subsystem and the partial order plan subsystem. A dialog subsystem was built to use the relations and allow Simon to create a small dialog about the goals that were discovered from the original learning framework using synthesis and recognition. This was also followed up with a method of using traditional optimization to solve first order logical sentences with grounded relations.

5.1 Relational Symbol Grounding

The relational grounded learning revolves around one continuous feature from the perception system to provide higher level reasoning symbols: location. I focus mainly on spatial relational learning for my task learning. Grounded symbols are labels such as $Left\_Of\,(A, B)$ or $Right\_Of\,(A, B)$ or even $Next\_To\,(A, B)$. (Background regarding first order logic can be found in appendix A.2.) I reduce the symbols representation to a uni-modal Gaussian by taking the two locations of the referents and subtracting one from the other as the sample data for the learner.

Figure 8 and 9 visualize a simple relational example: $Left\_Of$. In Figure 9, positive examples of “this object is to the left of this object” are used to find the mean and covariance of the sample set. Once they are calculated, the model is used

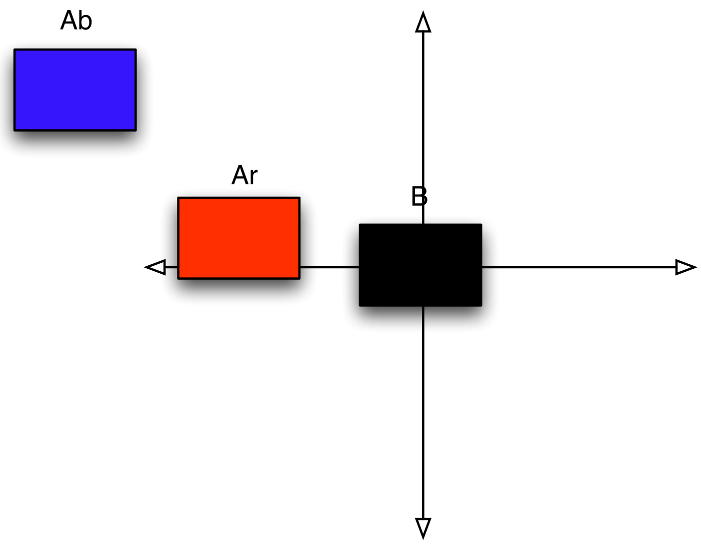

Figure 8: Example situation: Object Ab or Ar to the left of Object B?

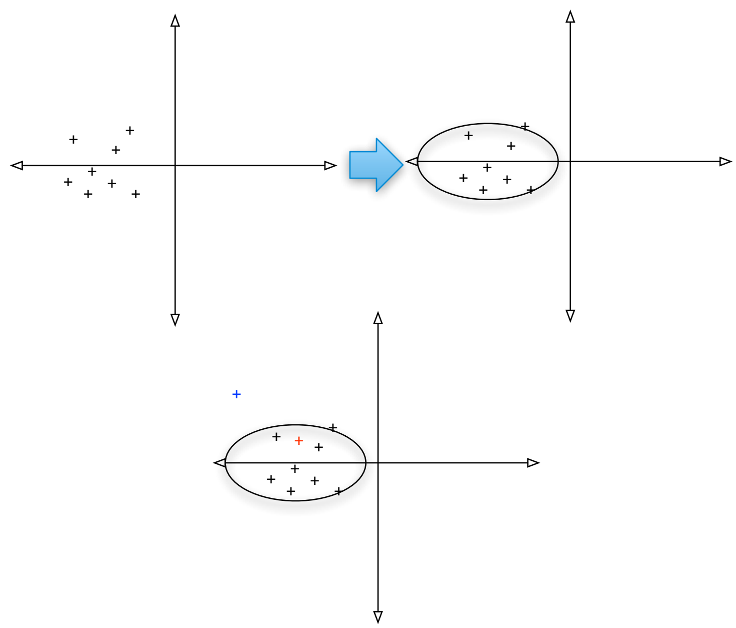

Figure 9: Example grounded relation: Left_Of

as a discriminator to test whether or not the blue sample fits. In this case, it doesn't, it is an outlier and thus the symbol is not activated. On the other hand, the red symbol fits the model, $Left\_Of\,(Ar, B)$, learned from the training data and is returned as a positive fit. So for the blue sample, the position of some object B is not to the left of some object A but for the red sample, we can safely say it is true.

The relational position, $\vec{v}_r = \vec{v}_A - \vec{v}_B$ is used to fit a Gaussian by finding the mean and covariance. With n number of examples and v_{r_i}, the relational position of the ith sample then the mean and covariance are found as usual:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} \vec{v}_{r_i}$$
$$\Sigma^* = \frac{1}{n-1}\sum_{i=1}^{n} \left(\vec{v}_{r_i} - \mu\right)\left(\vec{v}_{r_i} - \mu\right)^T$$

Similar to the perception system, grounded symbols are evaluated on a tree in which primitive relational features are first computed. The basic percepts that are available to all of the higher level spatial percepts are *relational distance* and *relational position*. These are special features that are not trained and are special in that they are provided to the framework as-is. The symbols used in my study are the following:

- $Relational\_Position\,(A, B)$ with value type (x,y,z) or $\vec{v}_r$
- $Relational\_Distance\,(A, B)$ with value type scalar distance
- $Left\_Of\,(A, B)$ with value type : confidence
- $Right\_Of\,(A, B)$ with value type : confidence
- $Top\_Of\,(A, B)$ with value type : confidence
- $Bottom\_Of\,(A, B)$ with value type : confidence
- $Next\_To\,(A, B)$ with value type : confidence

The relational position, $\vec{v}_r$ and L2 norm relational distance, $|\vec{v}_r|_2$, provide the dependency data data required for the uni-modal Gaussians to properly determine their confidence. Some examples of of higher level relational features that require the primitive features include $Left\_Of$ which requires a multidimensional evaluation on the position of the objects with respect to one another and $Next\_To$ which could be described with a multidimensional Gaussian but is trained as a one dimensional Gaussian.

This model, once trained, is labeled and new data can be determined to fit this model. Discrimination is a simple thresholded p-value. So for some Γ_R relational alphabet (in our case, the taxonomy above) and some symbol $\gamma \in \Gamma$, the mechanism finds the distance γ^* of some sample $\vec{v}$,

$$\vec{v}_d = \vec{v} - \mu_\gamma,$$
$$\gamma^* = \vec{v}_d \cdot \frac{1}{\sqrt{\left|\Sigma_\gamma \left(\frac{1}{|\vec{v}_d|_1}\vec{v}_d\right)\right|}}$$

where $|\vec{v}_d|_1$ is the L1 Norm. This determines distance from mean value in units of deviation. We can then threshold the value and decide whether or not the evaluated symbol, $\gamma = Left\_Of$ for instance, with the specified parameters (object Ar and B from the example) can be confidently said to be true using this measure of confidence. In my case, I threshold the value to two units to evaluate whether γ is true when it is observed in the environment.

These relations were trained synthetically by an expert, myself, before the study began and all use location as a primary feature from the belief system.

5.2 Grounded Relational Task Learning

At a high level, I define a relation to be some type of grounded symbol between two objects. These trained relations, once grounded and observed, are aggregated into

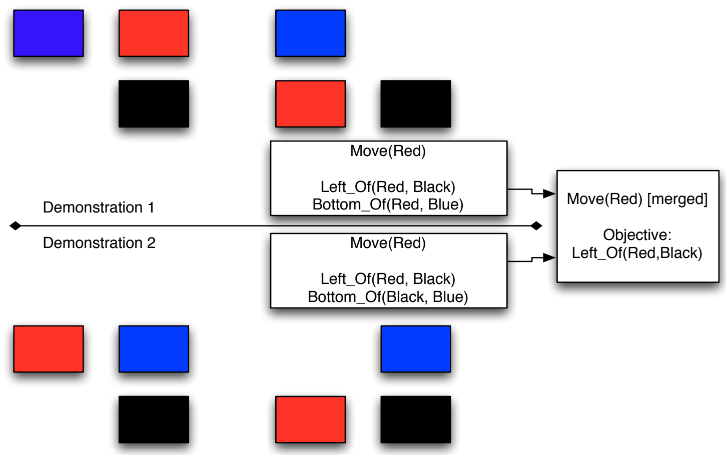

Figure 10: Example of merging actions to determine objective

each action's objective set and are merged into a single list of potential objectives of that action (see Chapter 4 and the pseudocode for the merge in Algorithm 4.1 for more detail) that were asked to merge. If the action consistently produces the same results, then that is the objective.

I define all actions in the task learning system to be primitive "move" actions. The example in Figure 10 shows a simple example from the clean up task. This example reduces the problem a bit to aid explanation. In the real task, the move action would also contain shape data and the colors would be yellow and blue. In this example, the changed state of the world is found to be that the red object moved. An action is created called "Move(Red)" with the objective of that action becoming just the changed relations for that action. When the actions are merged in the partial order plan learning, the objectives are merged but the sequence remains. Anything that is consistent in every action in every demonstration remains as an objective for that action. This method was inspired by previous work as well [33]. In previous work, a far more complete solution is provided in which the criteria and expectations of a goal are maintained. An expectation is the desired feature value while the criteria are the beliefs that are relevant to apply that expectations to. These are maintained to determine whether it can proceed with the task. Also in the original version, as well as newer implementations[10], an entire version space is enumerated during generalization.

5.3 Learning and Planning in Partial Order Plans

Since a partial order plan (or POP for short) is a directed graph in nature, (see appendix B), there are some interesting characteristics, namely it becomes easier to linearize POP trees. Child nodes in my notation precede parent nodes in demonstration. Since the POP is learned from demonstration, all of the starting positions represent some initial state in a demonstration. Starting at these leaf nodes, we can be guaranteed to terminate. For every node that has multiple children, then as long as the child sequences connect in the future, we must satisfy all of the child sequences. Luckily, this is never encountered for my implementation since my representation can not learn partial order plans this way so the linearization algorithm (presented later) does not take this into account even though it is possible. Learning happens using a variant of most common subsequence (see Figure 11) and my particular linearization algorithm uses a modified depth first search (or an algorithm that iterates over the tree, most recent child found first) to generate a random linearization from the partial order plan. The intuition is similar to that of a threading model - forking and joining - except that each "thread" is queued until the main thread is complete. Any action that has multiple children need all child paths satisfied before continuing with the linearization. Sibling nodes are queued and not executed until active child sequence reaches the join stage in the future, which the parent with multiple children can proceed. My particular algorithm (Algorithm 5.1), a particular variant of the linearization method found in [32], focuses on a single task that must join in the future.

Since we are learning the partial order plan, and the objectives simultaneously

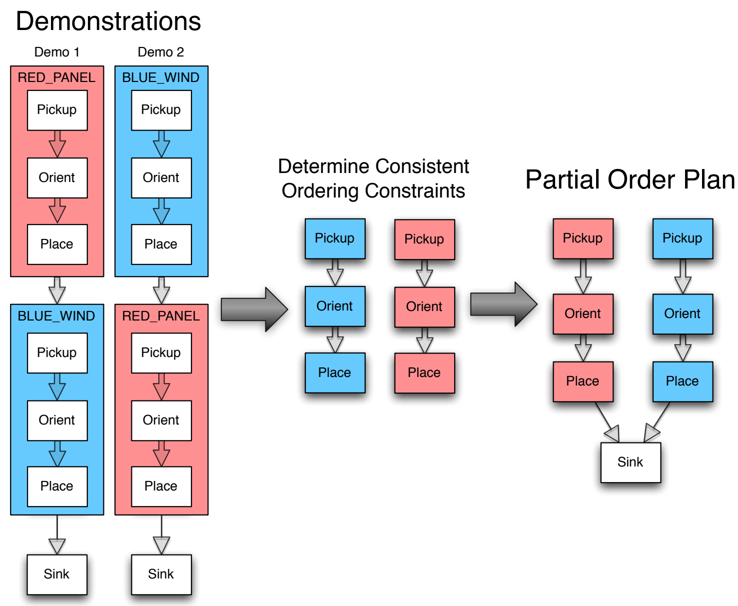

Figure 11: Most Common Subsequence POP Learning from the door task

then correspondence becomes a challenge between demonstrations. This is resolved using a constraint to the user that each demonstration only gets to move an object once and that correspondence is solved by using the object of attention to do action matching before merging and determining the consistency of each action. Once the actions correspond, then they are sequenced by the partial order plan learner. These actions are exactly represented by something that looks like: "Move(<object>)". The goal of the task ends up being the sum total of the objectives in the linearization.

Learning the partial order plan has been explored a few times in LbD. Famous architectures include the PRODIGY system [36] which was a fully integrated planning system that included partial ordering constraints on the plans generated. Though this is different than ours in that it was the sole focus and the learning aspect was not grounded in dynamic symbols. Also UCPOP is an earlier famous work that was able to handle actions that produced conditional effects [31]. My partial order plan learning uses a simple and proven algorithm - that of "most common sub sequence", which had success in earlier work in natural methods of robot POP learning [29]. Most

```
startnodes ← all_leaf_nodes(task_tree)
push_bottom(stack,startnodes)
visited_children(startnodes)
while notempty(stack) do
  currentAction ← pop(stack)
  execute(currentAction)
  for child in children(currentAction) do
    visited_child(child)
    if num_parents(child) == num_visited_child(child) then
      push_top(stack, child)
    else
      push_bottom(stack, child)
    end if
  end for
end while
```

Algorithm 5.1: Linearization of a Partial Order Plan

common subsequence POP learners consider every action preceding a future action to be a potential sequential constraint of the future action. Once enumerated, these are filtered by consistency and linked into a tree structure. The consistent ordering constraints are then linked together to form the partial order plan as in Figure 11. My variant POP learner algorithm can be found in Figures 5.2 and 5.3 where each demonstration is a list of all transitions from a preceding node a node that follows it in the demonstration sequence.

```
function CountParents(demonstrations) : sink_node
for demo in demonstrations do
  for transition in demo do
    if transition exists in all other demonstrations then
      map this transition as child → all potential parents
    end if
  end for
end for
return FindStructure(alltransitions, sink, emptyset)
end function
```

Algorithm 5.2: Preparing demonstrations for sequencing the Partial Order Plan

Once the linearization of the partial order plan is created, the objectives need to be satisfied. I developed a method for satisfying the discrete objectives based on the

```
function FindStructure(data, node) : node
remove node as parent in all transitions in data
for transition in data do
  if transition has zero parents left then
    add as child to node
    remove transition from data
  end if
end for
for child in node do
  FindStructure(data, child)
end for
return node
end function
```

Algorithm 5.3: Sequencing the Partial Order Plan

$$PDF = P(R1|pos) * P(R2|pos) * ...$$

$$\gamma^* = \prod_{i=1}^{R} \gamma^{R_i}$$

Figure 12: Objective function used during optimization

underlying distribution using an optimization procedure. Given some set of symbols, say $Left\_Of(A, B)$ and $Top\_Of(A, C)$ then by maximizing the p-value in the symbols domain, we can find the best position to place the object of interest to satisfy the most symbols in the set. In this example, it is possible to satisfy both of them, but in some cases, the algorithm could be given a set to satisfy that is unsatisfiable. This algorithm still attempts to solve it as best as it can given its limitations. I use the Nelder-Mead algorithm [28] to find a local maximum given a start position which I seed to be the average means of the first two first order relations in the objective. The objective function I use is simply the joint probability distribution function of the relations. Figure 12 illustrates this objective function.

Nelder-Mead's simplex algorithm simply returns the position of the local maxima that satisfies some subset of of the relational symbols. This is then used as the expected position of the object of interest for that move action.

CHAPTER VI

TRANSPARENT INTERACTION

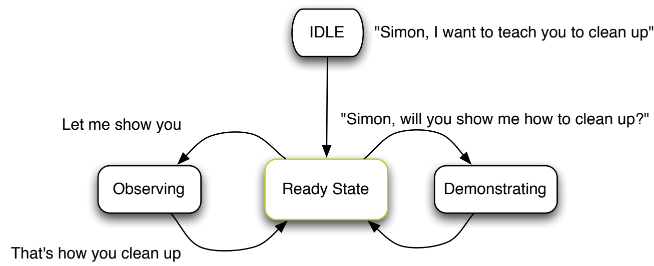

Figure 13: Interaction Diagram

Focusing on task transparency, I wanted to study how a robot can reveal its internal representation using a few mechanisms: pointing, dialog, and gaze. These social cues were used as transparency mechanisms that have been shown to be useful in the past [7, 27]. I coded a grammar that was given to the robot and sentence tagging, provided in the Speech API, was used to extract the commands about objects and relations that the recognition software knew about to generate goals and plans to execute. My particular implementation combines these ideas with our learning system to learn and build complex goals that include both action and sequence constraints (see Section 5.3 and Section 5.2).

I avoided the problem of manipulation of the objects when executing the task to simplify the problem. In the two scenarios in Chapter 2, once the observing phase completes (Figure 13), then the robot will need to demonstrate its understood objectives of the task. In the case of moving the objects for each scenario (panels for the door, blocks for cleanup), I programmed Simon to ask the participant to move, align, and place the objects in lieu of the robot. Figure 14 explores the differences

Non task-transparent	User Action	Task-transparent
Looking straight ahead		Looking at head height, gently looking around
↓	“Will you show me how to build a door”	↓
”Pickup the <object>”		”Pickup the <object> and looks in that direction”
↓	“Okay Simon, I did that.”	↓
“Put it here”		“Put it here”
<Points to position>		<Points to position>
↓		“I believe that was because...”
	<dialog>	<dialog>
	Repeats until user satisfied	
	“Okay Simon, let’s move on”	

Repeats until task complete

Figure 14: Interaction scenario during participant demonstration between robot and human

between a transparent interaction interaction and what is required of the user for manipulation.

In the example of building a door, one sequence of events may be that the participant requests that the robot enter a learning state in which the robot will learn to build a door. The participant may then give a demonstration and complete the learning process (see Chapter 5). After the demonstration is complete, the user may enter a demonstration mode. This interaction, in the task-transparent case, will do so while conversing about its (potentially) overly specific goal. The user may then have the opportunity to correct the robot’s internal goal.

I defined task transparency to have three major components: Dialog, Pointing, and Gazing.

6.1 Robot Pointing and Gazing

To maintain a fluent interaction, robot pointing was only used to clarify some continuous destination for an object. Since manipulation is not implemented, the robot had a need to request that the human place an object of reference (again from the action in the POP, see 5.3), at a specific location. When the request arose for Simon, the robot would ask that the "<object>" be placed "here" which is followed by a pointing gesture to the location. It is important to note here that pointing is a challenging manipulation problem. For the purposes of my experiment, I designed a set of pointing animations using a 3D animation tool, Maya, and the position that is requested by the optimization procedure is mapped to the closest animation that points in that direction.

The gazing behavior is a challenging inverse kinematics problem. Our codebase, c6m, uses a modified CCD implementation that allows me to make the robot look at a position in the distance without reaching out for it with is neck. I took advantage of this in my code and had it look at the ARToolkit+ tags. The tags were calibrated into Simon's frame before having it gaze at the tags. The gazing behavior focuses on the object of interest during the task or it is not used at all in the case of a non-transparent interaction.

6.2 Task-transparent Dialog

I presented a script to the user that allowed them to communicate their intentions to the robot. Table 1 gives a basic overview regarding the interaction commands that are allowed by the participant. The structure of the interaction is managed by a finite state machine that maintains and controls the interaction.

One example of a basic interaction is as follows. The user approaches the robot and begins a task "I want to teach you to build a door" which puts the robot into learning mode and begins a new task. The new task is created in its memory and a snapshot of the current belief system is taken to compare later. This snapshot is used to build a consistent goal for the task to be relevant when further demonstrations are given. This process was documented earlier in Chapter 4. The user then creates a demonstration or sequence of actions by beginning and ending actions repeatedly using the script in Table 1.

1. The user proclaims "I am moving an object", which begins a new primitive "move" action and capture the current beliefs to compare later.

2. The user then moves the object and completes the action by saying, "Did you see that, Simon?".

3. This ends the action, takes a snapshot of the belief system and adds the action to the partial order plan to be later sequenced.

4. Steps 1-3 are repeated until the demonstration is completed.

5. Upon completion of the entire demonstration, the user finalizes the sequence of actions, or the demonstration, by telling the robot, "That's it Simon, that's how you build a door". This finalizes the task by capturing the relations from the belief system for the demonstration itself.

6. Subsequent demonstrations can be merged into the task by using the command "Simon, let me show you again" which recalls the most current task from memory and begins another demonstration. This demonstration is appended to the set of observed demonstrations and the interaction repeats.

The merging of the actions is documented in section 5.3 which create not only the partial order plan constraints but also merges the action set into a minimum number of actions.

After the participant builds the initial understanding of the task using demonstrations, a human teacher can modify the task using more demonstrations or if presented with the opportunity, can have a dialog with Simon to further clarify the task. Once the robot is asked, "Simon, will you show me how to <task>?", the robot begins to demonstrate the task, revealing what it did and did not understand about the demonstrations. During this exercise, the human discovers whether or not Simon has correctly understood the task. The robot will then linearize the partial order plan and given its observations (which for our purposes, starts the plan off in the initial state of the demonstration), would start iterating through its plan. For each action that it encounters, it asks to place the object of attention at a particular spot that it points at (explored in section 6.1). At this point, the robot, in the task transparent interaction will go into a state of dialog. The dialog interaction allows the user to modify the internal objectives of that interaction using a few commands, again found in Table 1. Since there is the possibility that the interaction involves many possible relations (for example, the door task which regularly produces more than 30 relations), a set of five objectives of that action were randomly sampled and expressed to the user in the form of "I believe the objective is for <object> to be to the <relation> of <object> and for ...".

Simon is able to reveal its objective using this formulaic sentence. For instance, in the case of the clean-up task (Figure 5), the objective may be: "I believe the objective is for the blue block to be next to the blue keystone and the yellow arch to be next to the yellow block.". In this case, the objective is correct, but in the case that it is too specific, it may also include the case "The blue block is on top of the yellow block" which is irrelevant to the goal but was present in the demonstration. In this case, the user is allowed to modify the objectives by following the script found in Table 1.

For instance, a user may request that the robot "forget about the blue block" which may be irrelevant to the current action's objectives. These clarifications were shown to significantly modify the learned goal as the dialog proceeded.

Table 1: Dialog commands cheatsheet for the user

"I want to teach you to build a door" "I want to teach you to clean up"	Start the task <X>
"I am moving an object"	Begin an action
"Did you see that, Simon?"	End an action
"That's it Simon, that's how you build a door" "That's it Simon, that's how you clean up"	End the task <X>
"Will you show me how to <X>"	Have the robot execute the most current task
"Let me show you again."	Repeat a demonstration of the most current task in memory
"Okay Simon, I did that." "Okay Simon, let's move on."	Move on to the next action in the Partial Order Plan
"Forget about the <object>"	Filter out all objectives of current action with <object> as one of the referents
"Also consider <object1> <relation> <object2>"	Add an explicit relation with <object1> and <object2> as the referents
"Only consider <object>"	Similar to the *forget* action, this command filters out all objectives where <object> is not one of the referents.

6.2.1 Clarifications

The clarifications have very specific effects on the objective. I defined the following three clarification mechanisms:

- "Forget" <object>: filters the objectives. For example, for all A and B of relation(A,B) in objectives, if A or B is <object>, then remove that objective
- "Also consider <object1> <relation> <object2>": explicitly add <relation>(<object1>, <object2>) to the objectives
- "Only consider <object>": filter the objectives. For example, for all A and B of relation(A,B) in objectives, if A or B is <object>, then keep that objective, otherwise remove

In the door task, there were twenty-one tags on the table at all times. In this example, if the participants were teaching with red only, the participants may find it easy to start by saying "Forget about the blue panel" and "Forget about the blue window". This will remove all of the overly specific objectives that surround the blue objects and generalize the task quickly by allowing the user to understand and modify the internal representation without going back to demonstration.

CHAPTER VII

EVALUATION

My hypothesis is that *task-transparency aids in the accuracy of the learned goal to the mental model of the teacher*. As such, the null hypothesis is simply that task-transparency has no effect on the accuracy of the goal. I tested my interaction on eighteen subjects from both outside the Georgia Institute of Technology and from within. Seven of these participants were robotics students from within the college and had experience with robots, four were from within the College of Computing and were considered to have intermediate experience and seven were considered novices. Eight students were in group A and ten students were in group B. Each of the participants received a "cheatsheet" of the speech commands that the robot was able to understand along with very specific representations of the task that explained the relations that the robot understands as well as an explanation of the goal that they are to teach. They were provided assistance if they had questions and were allowed to ask questions until they felt comfortable with the task before the study began. After the study began, they were only allowed to ask questions related to the functioning of the speech recognition system since recognition had a high miss rate for certain dialects. I tested two interactions on two tasks, pictured in Table 2.[1]

Table 2: Group design

	NTT	TT
Door Task	Group A	Group B
Clean Up	Group B	Group A
Time	Step 1	Step 2

[1]NTT: non-task transparent interaction, TT:task-transparent interaction

NTT	User Action	TT
Looking straight ahead		Looking at head height, gently looking around
↓	“I’m moving an Object”	↓
↓	<Moving object>	Trying to follow movement
↓	“Did you see that Simon?”	↓

Repeats until demonstration completed

Figure 15: Interaction scenario during participant observation between robot and human

In Table 2, Group A received the non-task-transparent interaction with the door task before they received the task-transparent interaction with the clean up task. Thus our independent variable is whether or not the human is training the robot with the task-transparent interaction or with the non-transparent interaction. This within study was counter-balanced and corrected for task familiarity by providing two different interactions with different two tasks. To ensure that task preference didn’t bias the results, I collected results to validate that task preference did not contribute to the qualitative results. The results can be found in Figure 19. While they show a small bias toward the clean up task, it isn’t significant enough to warrant worry.

7.1 Experimental Design

My study involves two phases, teaching and demonstrating.For the teaching phase, Simon observes the participant’s actions and learns from a structured interaction. The sequence diagram for the learning phase can be found on Figure 15.

For the learning phase, there are two possible interactions, non task-transparent (NTT) and task-transparent (TT) which yield familiar but subtly different interactions. For the baseline, NTT interaction, Simon gives little indication that it is observing or paying attention while the TT interaction provides a small and subtle gazing interaction. For both of the interactions, Simon provides verbal feedback in the form of “Okay” to acknowledge that the beginning of the action was received

NTT	User Action	TT
Looking straight ahead		Looking at head height, gently looking around
↓	“Will you show me how to build a door”	↓
”Pickup the <object>”		”Pickup the <object> and looks in that direction”
↓	“Okay Simon, I did that.”	↓
“Put it here”		“Put it here”
<Points to position>		<Points to position>
↓		“I believe that was because...”
	<dialog>	<dialog>
	Repeats until user satisfied	
	“Okay Simon, let’s move on”	

Repeats until POP satisfied

Figure 16: Interaction scenario during participant demonstration between robot and human (repeated)

followed by a “Yes, I saw that” when the action is completed and Simon has properly recognized the end of action phrase. For the NTT interaction, the participant was allowed to return to this phase as a way of clarification to take advantage of the strict learning by demonstration input.

The learning phase uses the same task learning code and, with the same input, gives the same results. It is in the demonstration phase where things change significantly. In the NTT interaction, speech synthesis and pointing are used to communicate actions and intentions but not the internal representation of the task while the task-transparent version expressed the learned task by utilizing gaze, pointing, and a dialog interaction that afforded the user a clarification mechanism. In the NTT interaction, the user was allowed to repeat the demonstration until they felt Simon had successfully learned the task. In the task-transparent interaction, they were asked to teach Simon the task followed by a dialog in which Simon was able to express the learned task and provided the dialog mechanism to modify the internal representation.

The NTT interaction is meant to reflect a somewhat typical interaction in LbD systems. Simon looked straight ahead to avoid gazing at the wrong objects, commanded the user using speech synthesis to perform the action and only pointed when the objective could not be explained using speech synthesis. When clarifications needed to be made, the user would go back to the learning phase to do it.

On the other hand, the TT interaction enabled Simon to point (as in the NTT version), to look at the objects of interest, and a dialog system. This is considered the task-transparent interaction since the internal objective is revealed through a few interactive channels. I show that task-transparent interactions improve accuracy in the learned task as well as being considered more intelligent compared to the non task-transparent interactions.

7.1.1 Data collection

During the interaction, data was collected from the user for further analysis. My dependent variable, accuracy, was measured during the interaction according to our distance metric (Section 7.2). The objective of our study was to determine the effects of task transparency on goal accuracy. As such, I collected data after each demonstration and after each clarification. This afforded the granularity per clarification of the objectives. For instance, the baseline interaction allowed the participant to teach as many times as required but was unable to make any direct clarifications to the learned goal since it was not revealed. The participants task and belief system is captured after each command. More specifically, the current action's objectives are captured to analyze what the participants attended to in the clarification.

7.2 Distance Metric

In order to compare and show a difference between our experimental conditions (NTT vs TT), I use the dependent variable of goal accuracy.I define the accuracy of a goal to be the hamming distance from one goal to another with the cardinality of the union set as the maximum distance.

The distance measure between the goals of an action or task from the expert, G_E, and the goals of the respective action or task from the naive user, G_P, is my measured dependent variable. The distance defines the accuracy of what was intended to be taught (through the task-transparent framework) by the expert to what was actually taught by the participant in that particular interaction. The distance was defined to be:

$$d = len\,(G_E \cup G_P) - len\,(G_E \cap G_P) \tag{1}$$

Note that $len\,(G_U \cup G_P) \geq len\,(G_E \cap G_P)$ thus the distance metric is never negative, i.e. $d \in [0, len\,(G_E \cup G_P)]$. Where zero is the same and $len\,(G_E \cup G_P)$ is perfectly different.

In the case where the $len\,(G_E \cup G_P) > len\,(G_E \cap G_P)$, then the extraneous objectives grow linearly with each wrong objective and don't grow at all with each objective that is in line with what the expert taught. Special code was written to normalize the relational equivalencies. For instance "left AND bottom" between two referents is equivalent to "right AND top" with the referents in opposite order. These identities were taken into account when calculating the distances in the code. This was used to measure the accuracy of the goal both when the task was complete and when each clarification was made. My hypothesis was that between the independent variable (TT vs NTT), the accuracy is smaller for the TT interaction. Also of interest was to characterize how the user affected the accuracy for each clarification during the demonstration phase.

Table 3: Expertly trained goals for the two tasks

Clean up Task	
$G_1 =$	$\{Next\_To$(Yellow_Block, Yellow_Arch), $Next\_To$(Blue_Block, Blue_Keystone)$\}$
Build door Task	
$G_2 =$	$\{Left\_Of$(Panel,Tag#17), Top_Of(Window,Panel), $Bottom\_Of$(Panel, Tag#17), Top_Of(Panel,Tag#15), $Bottom\_Of$(Panel, Tag#18), Top_Of(Window, Tag#18), $Right\_Of$(Panel, Tag#18), Top_Of(Window, Tag#17), $Left\_Of$(Window, Tag#17), $Right\_Of$(Window, Tag#18)$\}$

7.3 Task Scenarios

Each user was asked to teach the robot how to clean up and to build a door. See Chapter 2 for basic details. Simon learned with the following grounded relations (trained by an expert): $Left\_Of$, $Right\_Of$, Top_Of, $Bottom\_Of$, and $Next\_To$. The goals for each task (Table 3) are those that are consistently met after all demonstrations and clarifications are complete. This was given to the user for reference if asked.

With the goal accuracy not normalized in my distance metric, it is impossible to compare distances between these goals. This was intentional. The larger the cardinality of the set $G_P \cup G_E$ (goal of the participant & goal of the expert), the smaller the effect of differences between the expert's goal and the participant's goal. This means that for simple tasks, such as the "clean up" task, a single difference in the task's goal representation can change the normalized value by larger amounts. Or in other words, for x,y as the cardinality of the two respective tasks, one difference is larger than the other, $\frac{1}{x} < \frac{1}{y}$, when $x > y$. For this reason, I analyze the task data separately between my independent variable for each task.

Table 4: Clean Up Task Accuracy

Clean Up		
	Non-Task-Transparent	Task-Transparent
Mean Distance	1.8	0.63
Median Distance	2	1
p-value	$< .001$	

7.4 Results

I analyzed the dependent variable, goal accuracy, between the NTT interaction and the TT interaction for each groups data. Since, the task accuracy can't be compared between tasks, the data had to be split into four sets. Data from each group representing the NTT and TT interactions for each task. I analyze the independent variable by task by doing two separate one-tailed student's t-test for the independent variable, NTT/TT, for each task to determine significance.

In the case of determining accuracy with respect to the action during clarification, the object of interest is used to determine the goal of just that object. The object of interest is defined to be the object that was moved during the primitive move action. Once we determine the goal of just that action, the distance is calculated with respect to just that action to the experts action.

7.4.1 Task Accuracy

Task accuracy based on the previous metric, Section 7.2, where a smaller value is better (shorter distance to expert demonstration) yields positive results. In fact, for both tasks, the accuracy yields better and more accurate tasks to what the user was told to teach. For both the complicated door building task and the simple clean up task, the task is modified in a statistically significant way to reduce the more specific learned goal to something that is more general in more situations. See table 4 and table 5.

Table 5: Build a Door Task Accuracy

Door Task		
	Non-Task-Transparent	Task-Transparent
Mean Distance	80.38	33.2
Median Distance	77.5	27
p-value	$< .001$	

Task-transparency aids the user in shaping what is learned by the robot by clarifying and reducing the specificity of the goals produced with learning by demonstration alone. Each grounded symbol that is consistent in the learned task is intrinsic to the set of demonstrations learned and is actually a constraint on the optimization. The explicit modifications allow the user to remove extraneous constraints on the task and generalize further past what was taught by the demonstrations.

Analysis of experience to accuracy reveal that experience did not in general affect accuracy results. Pairwise t-tests reveal that all p-values between accuracy in each group for each level of experience (robotics student, computer experience, or novice) are all ≥ 0.09, which I consider insignificant since all values are > 0.05.

For all of the users who performed the door task, Simon reproduced the task exactly and the user chose not to repeat it. For the cleanup task, six of the eight users performed only once and the remaining three users in the NTT group demonstrated the twice to clarify. For most of the extraneous constraints, the tags that were removed from the demonstration for the door task were immutable table fiducials. Despite all corrections in the demonstration set, these can not be generalized away using traditional learning by demonstration methods given the task environment. Any number of demonstrations cannot remove immutable always-there features if the demonstrations must happen in the presence of such features. By using task-transparency, the user is able to remove them and provide a more accurate representation of the task to Simon.

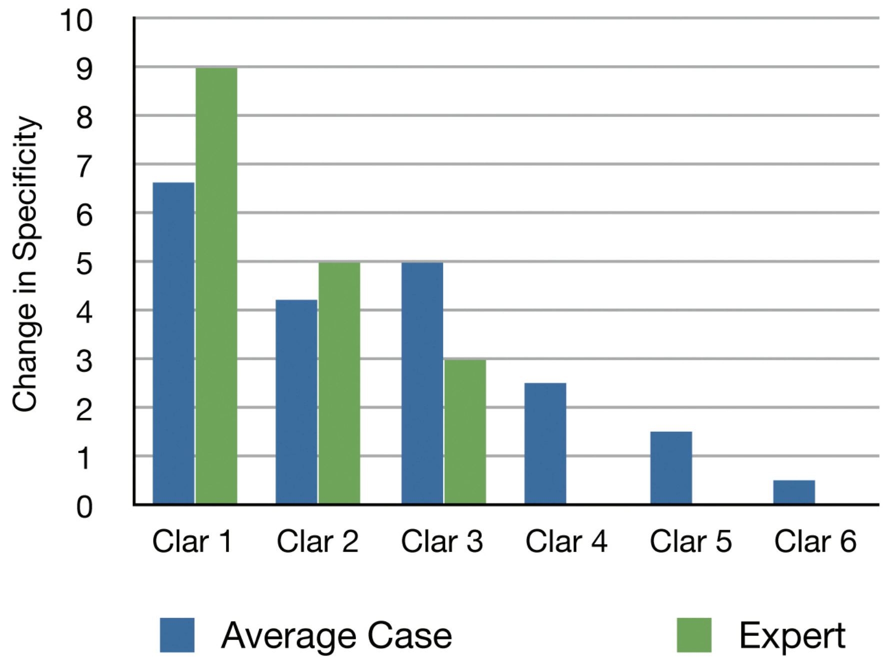

Figure 17: Average change in specificity of the goal after each clarification

7.4.2 Clarification Analysis

One interesting consequence of the dialog is that we can see how each clarification changes the specificity of the goal (defined by the number of features that changed or the delta). For instance, if a single clarification is made, then we'd expect the number of features to either increase or decrease in value. The most interesting case is with respect to building a door. In Figure 17, I've aligned each action to the expert's equivalent action and found the difference of the clarified series. We can see that the first clarification changes the specificity the most. For instance, if the user understands that Simon has somehow mischaracterized the goal by including some extraneous blue window when the real objective only involved the red panel, then by removing the blue window, six relations were removed from the goal, thus reducing the specificity[2]. As time went on, the effect of each clarification was, on average, reduced.

[2]See Chapter 6, Section 6.2.1 for the detailed taxonomy of possible corrections and their effects

Finally, I wanted to analyze how long people were willing to interact with Simon. On average, for the complicated door task, the participant was willing to provide 4.2 clarifications on average while for the more simpler clean up task, 1.13 clarifications was sufficient.

7.4.3 Qualitative Response

In addition to the quantitative results, a questionnaire was given to allow a free response about the interaction between the robot and the human. The participants also received questions focusing on preference and perception. The basic questions that were asked are the following:

- "If you were to interact and teach with Simon on a daily basis - which of the two would you be willing to work with daily?"
- "Was there a difference in intelligence level between the two studies? If so: which did you perceive to be 'more intelligent' and if not: just mark 'same level of intelligence."

7.4.3.1 Perceived Intelligence

Of the seven responses that were received, six claimed that the task-transparent interaction was more intelligent despite the challenging interaction. Some select comments about this interaction can be found in Figure 18.

"I feel like study two[TT] is higher level because Simon talked about my objective which I can discuss about that."
"I was more impressed with what Simon could do in the 2nd study."
"I think that the asking and answering of questions showed a level of intelligence that was not present in the first study."

Figure 18: Select comments regarding perceived intelligence

7.4.3.2 Preference

Out of eighteen participants, nine preferred the task-transparent version and nine preferred the non-task-transparent version. Out of these, Figure 19, shows that they were, for the most part, not biased towards any one task.

An equal number of subjects preferred the TT interaction to the NTT interaction and the NTT interaction to the TT interaction. Analysis of the accuracy to the NTT

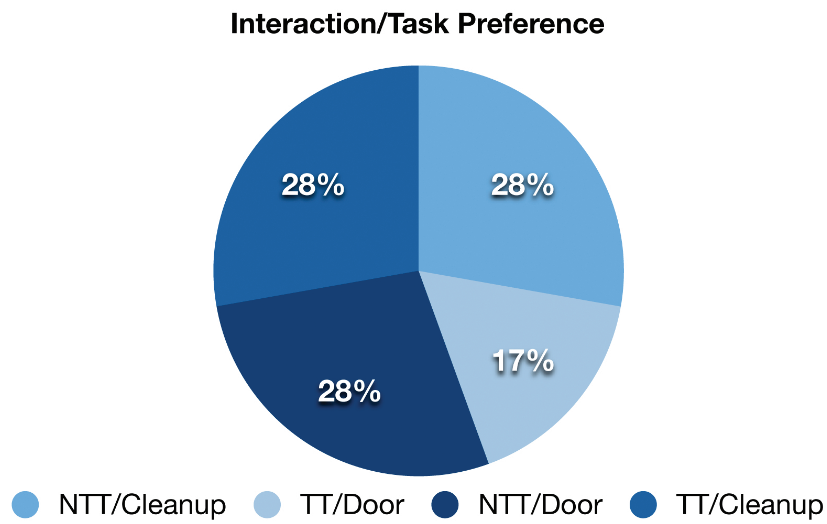

Figure 19: Task Preference out of Eighteen participants. Between Group A (NTT/D,TT/C) and Group B(NTT/C,TT/D), p-value: $0.31 > 0.05$

and TT preference reveals that there is no significant difference in accuracy between those who preferred the NTT interaction to those that preferred the TT interaction. Accuracy p-values for Group A, between those that preferred NTT to TT are 0.5 and 0.15 for the door building accuracy and cleanup accuracy respectively while Group B had p-values of 0.39 and 0.17. This also points out that preferring the NTT preference was not correlated with accuracy. The comments reveal just why participants had no preference toward the interaction type. The comments for these questions reveal that the subjects sometimes prefer the robot to not be so pedantic. One participant that prefers the intelligent interaction said, "I found it interesting having to correct and work with Simon as he learned and then made mistakes and then continued learning." said one participant who preferred the TT interaction. While others preferred the hands off approach of the NTT interaction, "It was [an] easy task, and I don't have to specify all [of the] rules." and "Even though in the second one, Simon was trying to tell me the goal of the task, but the way it presented [the task] is confusing and not obvious to me. So I prefer the first one." were just some of the responses of some of the users. The interaction may need some work to make it more amenable to human interaction but the goal accuracy is a positive metric that shows the advantage of the task-transparent version which may be useful for future task learning interactions.

7.4.3.3 Feedback regarding the verbal interaction

The verbal interaction that followed the execution was an important part of the experiment. I found that the implementation challenged the users who felt constrained by the script and confused by the speed of the speech. Out of the twelve responses regarding the verbal interaction, eight cited it positively and four cited it negatively. This reveals that there is nothing conclusive about this split opinion. One observation from the comments seems to suggest that those who had negative comments regarding the dialog pointed to Simon speaking too quickly and having a hard time understanding and keeping up with the relations and referents in Simon's frame. Just some of the feedback can be found in Figure 20.

Positive
• "I liked interacting with the robot and asking him questions." • "because i liked that Simon was talking to me more."
Negative
• "I feel like study two[TT] is higher level because Simon talked about my objective which I can discuss about that." • "I thought the explanations Simon gave were simpler. It was hard to follow all the detail Simon gave about the positioning of the window/panel in the second study. It was less confusing."

Figure 20: Select comments regarding the dialog interaction

7.5 *Discussion*

Learning by demonstration requires a very complicated coupled interaction that demands each participant to carry a large burden in maintaining a common experience. This puts the burden on the roboticist to design behaviors that facilitate this biological desire humans have developed and make available enough information to facilitate these social learning mechanisms, like those described in Tomasello[35]. Utilizing this information, the human teacher can transfer goals and tasks more effectively from human to machine. But most importantly, designing a coupled transparent algorithm is challenging. From an interactive standpoint, transparency in human computer interaction has always focused on making devices that don't reveal its internals but provides this information in ways that make sense for humans. Thomaz and Breazeal[33, 7] have had the most success in coupling these transparent robot learning mechanisms with learning algorithms. While my work shows that task-transparency provides good results, my study also shows that this interaction should be natural and needs to be a designed in a principled way, by designing algorithms that utilize transparency as a critical consideration. One of the basic findings of my study's discourse interaction was that my particular implementation spoke too fast, spoke in a way that made it hard to follow (in other words the referents were sometimes not labeled well enough), and that participants prefer this interaction to be available throughout the entire learning interaction. By measuring the accuracy per clarification, I have also characterized the clarification process as making only minor changes as further clarifications are provided indicating perhaps that further interactions have less of an impact on accuracy.

My hope is that, in the future, better design considerations will be enumerated that will provide a useful taxonomy for building learning interactions in socially guided machine learning. Thomaz, Breazeal, and Chao [33, 7, 10] have clearly provided a number of design considerations including gaze, pointing, gesture, and introspection (utilizing active learning). In this way, my work stands alongside theirs by additionally considering dialog. What I show in my study is that within the transparent design pillar, providing an interface to the symbolic goals itself in a way that feels natural to the human is as important of a design consideration as many of the non-verbal communicative acts have been shown to be.

CHAPTER VIII

CONTRIBUTIONS

Task transparency in robot learning through demonstration has been shown to improve task accuracy. Having seen positive results from my study, I can conclude a few things; 1) task transparency is an important behavioral design consideration in LbD, 2) users would prefer a less obtrusive transparency mechanism, and 3) the way users clarified the task takes on the strategy of making the largest changes to the specification first, followed by subsequent detailed clarifications.

In my study, I made an observation that in the door task, the features that required the most attention are the immutable features on the table. This is an important distinction between the task-transparent interaction and the non-task transparent interaction. The task transparent interaction provided a mechanism that allowed the participant to remove features that were unimportant. With traditional learning by demonstration, demonstrations may have the condition that they be trained in an environment that can not be modified. In this case, the task's criteria may be overly specific and require that the task only be performed in the same type of environment that it was trained in. What the task-transparent interaction provided was a mechanism to further generalize past the limitations of the environment that it was trained in.

To ensure that the users prefer one interaction over the other, my study made it obvious that the behavior needs to reveal the task in a way that makes sense for human teachers. My particular implementation was not transparent enough in the most traditional human computer interface terminology. In fact, the mechanism revealed far too much internal state and this got in the way of the natural interaction that the participants expected. My study emphasizes the need for the design of an unobtrusive discourse between human and robot about the internalized task to improve task learning accuracy and aid generalization.

APPENDIX A

LOGIC

A cursory explanation of logic follows for brevity. Further study in propositional logic, first order logic, and second order logic can be found in many fundamental Artificial Intelligence sources [32, 26, 24] I will only cover propositional and first order logic.

A.1 Propositional logic

Propositional logic is one of the most fundamental logics. It provides a mechanism for declarative statements that have some truth value and has traditionally been used to represent world state and can be used in planning and knowledge representation. Propositional logic includes some alphabet, Γ, or in our case, some number of features. It also uses a number of symbols that represents relationships between the features. These rules may include $P \rightarrow Q$ represents an "if-then" relationship where Q is true if P is true. This says nothing of Q's effect on P. Other basic symbols include $\neg P \rightarrow Q$, which uses a new symbol, $\neg$ that represents a negation, or in other words, when P is false, then Q is true. This only represents a single direction of inference. So if Q is known, we still can't say anything about P. We can directly link their relationship by using something called a biconditional, $\leftrightarrow$ that represents a direct relationship to each other. So $P \leftrightarrow Q$ will represent an "identity" where the value of P is always the same as Q. In general, a fully-defined propositional logic is some language that includes some alphabet and rules, or $\mathcal{L} = (\Gamma, \Omega, \zeta)$ where the Γ is our alphabet, for instance ($P, Q, Yellow\_Block, Yellow\_Arch$, etc), $\Omega = \{\rightarrow, \leftrightarrow, \neg, \wedge, \vee, etc\}$ represents the symbols we use to build our current statements, and finally, ζ represents our rules, for instance $P \wedge Q \rightarrow \neg T$ may represent some relationship or rule between the alphabet

Rules(ζ)	Example	Discussion
P	$Block$	A block object exists
$P \wedge Q$	$Yellow\_Block \wedge Yellow\_Arch$	Both the yellow block and yellow arch exist
$P \rightarrow Q$	$Arch \rightarrow Yellow\_Arch$	If an Arch exists, then a Yellow Arch also exists
$P \rightarrow \neg Q$	$Keystone \rightarrow \neg Yellow$	Since no keystone objects are yellow, by knowing if it is a keystone, it implies that it is yellow, the converse is not necessarily true.
$P \leftrightarrow Q$	$Yellow\_Block \leftrightarrow Blue\_Block$	If the yellow block exists, so does the blue block. If the blue block exists, the yellow block also exists.

Figure 21: Rough interpretation of propositional logic in the task-transparency study

that is observed or found to be consistent.

A.2 First Order Logic

In earnest, my task transparency study uses more formal first order logic than propositional logic. First order logic extends propositional logic to include predicates and quantification [32]. Predicates, in our usage, can help describe attributes or relations between symbols. For instance, {*IsLarge(a), IsBlue(a), IsYellow(a), Next_To(a, b)*} may represent some attribute of some variable, a. So we can create interesting rules such as *IsYellow(a)*$\rightarrow$*IsLarge(a)* then we can create rules that imply that allow yellow objects are also large. If we design $a \in \{Block\}$ and $b \in \}Keystone\}$ then we can even say that the block is next to the keystone using *Next_To(a,b)*. If a contained both $Keystone$ and $Block$ and *Next_To(a,a)* then we may have problems since a in this case could be *Next_To(Keystone,Keystone)*. This represents a need for quantification in the representation that helps generalize and constrain the variables usage. Generalization and quantification help define the limitations of our variable a in our examples. Two possible quantification symbols exist such as $\exists$ and $\forall$, which represent "there exists" and "for all" respectively. "There exists" quantifies the situation

Rules(ζ)	Example	Discussion
Next_To(a, b)	*Next_To(Yellow_Block, Yellow_Arch)*	The yellow block is next to the yellow arch
$\exists a$*IsKeystone(a)*	*IsKeystone(Blue_Keystone)*	At least one of the objects in the world is a keystone. In this case, there exists a blue keystone.

Figure 22: Rough interpretation of first order logic in the task-transparency study

such that, given some set of possible objects, say $a \in \{Arch, Block\}$ in our previous example, then we can safely say that within the set, a, the number of possible arches that exist is at least one. See Figure 22 for a few symbols that I use in my study.

APPENDIX B

PARTIAL ORDER PLANS

*Partial Order Plan*s (or POP for short) have successfully been used to represent sequential task constraints. When a task is executed by an agent, a concise definition can be helpful as a heuristic for its planner. A planner that sufficiently represents its state and its actions preconditions and postconditions may have the ability to plan its way through sequential constraints but by explicitly explaining sequential constraints, a useful heuristic may emerge. Classically, partial ordered plans are represented with some type of propositional or first order logic [32]. My example partial order plan can be found in Figure 23. In this example, you must place the window and the panel into place before you are finished with the task. But first, to accomplish that, the robot must execute both paths in some sequence. In this example, the plan points out the fact that you can orient and place the window and the panel in different orders but they must happen before the task is complete.

In this example, the objective is for both pieces to be placed (WindowPlaced, PanelPlaced) by first picking up the piece and then orienting it correctly followed by releasing it from the robots hand. A *linearization* of this partial order plan is one particular sequenced list of actions to take to accomplish the task. For instance, in our example one particular linearization is that you need to Pickup, Orient, and Release the window before the panel. Another linearization of this plan may be that you need to perform those actions on the panel before the door. Both are valid linearizations. More formally, most partial order plans can execute them in parallel. My execution code linearizes the partial order plan given the robot's limitations; but one particular correct linearization may be that the robot picks up the panel, orients it, picks up

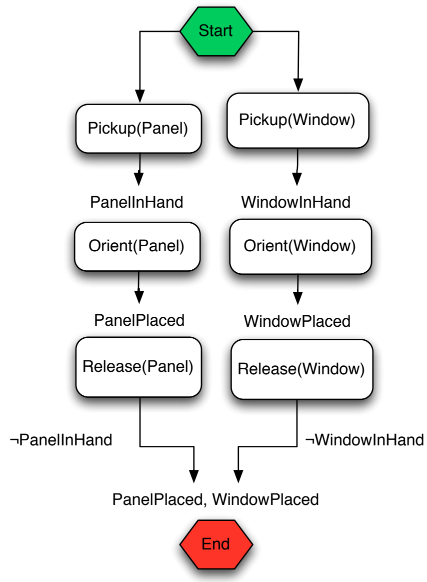

Figure 23: Example door building partial order plan

the window, places the panel, and finally orienting and placing the window. This is a valid linearization of the partial order plan but requires more capability than has been given to Simon by this study. After each action, a particular change in the world state is activated or deactivated. In the example, after a pickup action is executed, the symbol $WindowInHand$ becomes activated and needs to be deactivated to successfully accomplish the task. The objectives of an executed action need to be satisfied before the action is considered complete.

Partial order plans provide a good framework for building sequential constraints of a plan. While my implementation is not a complete solution, POPs have provided a framework for handling constraints in a formal way that make planning clear and unobtuse.

REFERENCES

[1] Abbeel, P. and Ng, A., "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*, p. 1, ACM, 2004.

[2] Argall, B., Chernova, S., Veloso, M., and Browning, B., "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009.

[3] Argyle, M., Ingham, R., and McCallin, M., "The different functions of gaze," *Semiotica*, pp. 19–32, 1973.

[4] Baron-Cohen, S., "Precursors to a theory of mind: Understanding attention in others," *Natural theories of mind: Evolution, development and simulation of everyday mindreading*, pp. 233–251, 1991.

[5] Blumberg, B., Downie, M., Ivanov, Y., Berlin, M., Johnson, M., and Tomlinson, B., "Integrated learning for interactive synthetic characters," in *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, 2002.

[6] Bratman, M., "Shared cooperative activity," *The Philosophical Review*, vol. 101, no. 2, pp. 327–341, 1992.

[7] Breazeal, C., Kidd, C., Thomaz, A., Hoffman, G., and Berlin, M., "Effects of nonverbal communication on efficiency and robustness in human-robot teamwork," *Proceedings of IEEE/RSJ International Conference on Intelligent Robotics and Systems*, 2005.

[8] Breazeal, C. and Thomaz, A., "Learning from human teachers with socially guided exploration," in *Proceedings of the International Conference on Robots and Automation (ICRA)*, 2008.

[9] Calinon, S., Guenter, F., and Billard, A., "On learning, representing, and generalizing a task in a humanoid robot," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 37, no. 2, pp. 286–298, 2007.

[10] Chao, C., Cakmak, M., and Thomaz, A., "Transparent active learning for robots," in *Proceeding of the 5th ACM/IEEE international conference on Human-robot interaction*, pp. 317–324, ACM, 2010.

[11] Chernova, S. and Veloso, M., "Confidence-based policy learning from demonstration using gaussian mixture models," in *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, p. 233, ACM, 2007.

[12] Demiris, Y. and Khadhouri, B., "Hierarchical attentive multiple models for execution and recognition of actions," *Robotics and Autonomous Systems*, vol. 54, no. 5, pp. 361–369, 2006.

[13] Dillmann, R., "Teaching and learning of robot tasks via observation of human performance," *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 109–116, 2004.

[14] Dix, A., Finlay, J., and Abowd, G., *Human-computer interaction.* Prentice hall, 2004.

[15] Edsinger, A., *Robot manipulation in human environments.* PhD thesis, Massachusetts Institute of Technology, 2007.

[16] Fong, T., Conti, F., Grange, S., and Baur, C., "Novel interfaces for remote driving: gesture, haptic and PDA," *SPIE Telemanipulator and Telepresence Technologies VII, Boston, MA*, 2000.

[17] Friedrich, H., Mnch, S., Dillmann, R., Bocionek, S., and Sassin, M., "Robot programming by demonstration (RPD): Supporting the induction by human interaction," *Machine Learning*, vol. 23, no. 2, pp. 163–189, 1996.

[18] Hancher, M. D., "A Motor Control Framework for Many-Axis Interactive Robots," Master's thesis, Massachusetts Institute of Technology, 2003.

[19] Hersch, M., Guenter, F., Calinon, S., and Billard, A., "Dynamical system modulation for robot learning via kinesthetic demonstrations," *IEEE Trans. on Robotics*, vol. 24, no. 6, pp. 1463–1467, 2008.

[20] Itti, L. and Koch, C., "Computational modelling of visual attention," *Nature Reviews Neuroscience*, vol. 2, no. 3, pp. 194–203, 2001.

[21] Itti, L., Koch, C., and Niebur, E., "A model of saliency-based visual attention for rapid scene analysis," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 20, no. 11, pp. 1254–1259, 1998.

[22] Kato, H. and Billinghurst, M., "Marker tracking and hmd calibration for a video-based augmented reality conferencing system," in *iwar*, p. 85, Published by the IEEE Computer Society, 1999.

[23] Kulic, D., Lee, D., Ott, C., and Nakamura, Y., "Incremental learning of full body motion primitives for humanoid robots," in *8th IEEE-RAS International Conference on Humanoid Robots, 2008. Humanoids 2008*, pp. 326–332, 2008.

[24] LaValle, S., *Planning algorithms.* Cambridge Univ Pr, 2006.

[25] Lockerd, A. and Breazeal, C., "Tutelage and socially guided robot learning," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings*, vol. 4, 2004.

[26] Luger, G., *Artificial Intelligence: Structures and Strategies for Complex Problem Solving, 5/e.* Pearson Education India, 2002.

[27] Mutlu, B., Shiwa, T., Kanda, T., Ishiguro, H., and Hagita, N., "Footing in human-robot conversations: how robots might shape participant roles using gaze cues," in *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, pp. 61–68, ACM, 2009.

[28] Nelder, J. and Mead, R., "A simplex method for function minimization," *The computer journal*, vol. 7, no. 4, p. 308, 1965.

[29] Nicolescu, M. and Mataric, M., "Natural methods for robot task learning: instructive demonstrations, generalization and practice," in *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pp. 241–248, ACM, 2003.

[30] Norman, D., *Emotional design.* Basic Books, 2004.

[31] Penberthy, J. and Weld, D., "UCPOP: A sound, complete, partial order planner for ADL," in *proceedings of the third international conference on knowledge representation and reasoning*, pp. 103–114, 1992.

[32] Russell, S., Norvig, P., Canny, J., Malik, J., and Edwards, D., *Artificial intelligence: a modern approach.* Prentice hall Englewood Cliffs, NJ, 1995.

[33] Thomaz, A., *Socially guided machine learning.* PhD thesis, Massachusetts Institute of Technology, 2006.

[34] Thomaz, A. and Breazeal, C., "Transparency and socially guided machine learning," in *5th Intl. Conf. on Development and Learning (ICDL)*, 2006.

[35] Tomasello, M., *The cultural origins of human cognition.* Harvard Univ Pr, 1999.

[36] Veloso, M., Carbonell, J., Perez, A., Borrajo, D., Fink, E., and Blythe, J., "Integrating planning and learning: The PRODIGY architecture," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 7, no. 1, pp. 81–120, 1995.

[37] Wang, L. and Chen, C., "A combined optimization method for solving the inverse kinematics problem of mechanical manipulators," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 4, pp. 489–499, 1991.

[38] Zang, P., Tian, R., Isbell, C., and Thomaz, A., "Batch versus Interactive Learning by Demonstration," in *IEEE 9th International Conference on Development and Learning, 2010. ICDL 2010*, 2010.