HUMAN ROBOT INTERACTION THROUGH SEMANTIC INTEGRATION OF MULTIPLE MODALITIES, DIALOG MANAGEMENT, AND CONTEXTS

BY

David O. Johnson

Submitted to the graduate degree program in
Electrical Engineering and Computer Science
and the Graduate Faculty of the University of Kansas
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy.

_____
Dr. Arvin Agah, Chairperson

Committee members _____
Dr. Swapan Chakrabarti

_____
Dr. Xue-Wen Chen

_____
Dr. Brian Potetz

_____
Dr. Sara Wilson

Date defended: _____

The Dissertation Committee for David O. Johnson certifies
that this is the approved version of the following dissertation:

HUMAN ROBOT INTERACTION THROUGH SEMANTIC INTEGRATION OF MULTIPLE MODALITIES, DIALOG MANAGEMENT, AND CONTEXTS

Committee:

_____
Dr. Arvin Agah, Chairperson

_____
Dr. Swapan Chakrabarti

_____
Dr. Xue-Wen Chen

_____
Dr. Brian Potetz

_____
Dr. Sara Wilson

Date approved: _____

Abstract

The hypothesis for this research is that applying the Human Computer Interaction (HCI) concepts of using multiple modalities, dialog management, context, and semantics to Human Robot Interaction (HRI) will improve the performance of Instruction Based Learning (IBL) compared to only using speech. We tested the hypothesis by simulating a domestic robot that can be taught to clean a house using a multi-modal interface. We used a method of semantically integrating the inputs from multiple modalities and contexts that multiplies a confidence score for each input by a Fusion Weight, sums the products, and then uses the input with the highest product sum. We developed an algorithm for determining the Fusion Weights. We concluded that different modalities, contexts, and modes of dialog management impact human robot interaction; however, which combination is better depends on the importance of the accuracy of learning what is taught versus the succinctness of the dialog between the user and the robot.

Acknowledgements

I would like to thank my advisor, Dr. Arvin Agah, for his help, advice, knowledge, experience, encouragement, and support in completing this research. I could not have done it without him. I hope we can continue to collaborate on future research projects. I would also like to thank the current members of my committee, Dr. Swapan Chakrabarti, Dr. Xue-Wen Chen, Dr. Brian Potetz, and Dr. Sara Wilson, and the former members, Dr. John Gauch and Dr. Donna Havercamp, for their guidance and feedback on this research.

I would also like to thank Frans Flippo for letting me use his Framework software, which is the foundation for all the software I wrote for this research project. Frans is currently working in the Netherlands as a freelance software developer.

I also appreciate the efforts of Nickolay V. Shmyrev, a SourceForge developer, who spent many hours working with me to get the Sphinx 4.0 speech recognizer tuned for this application.

And finally, I want to thank my wife, Elaine, my son, Drew, and my daughter, Morgan for their unending support in my effort to finish what I started many years ago.

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Motivation

Today, robot manufacturers program their robots to accomplish a defined set of functions before they leave the factory. Although the functions are advanced when compared with what robots could do as little as ten years ago, they are very limited when compared with the wide range of activities that a human can perform. As the mechanical and sensory capabilities of robots continue to advance, they will be able to perform increasingly human-like functions. In fact, the Holy Grail for some robot designers is to design a robot that is as human like as possible. Future robots will include domestic robots to do household chores, such as cleaning a house; industrial robots to do tasks that are dangerous to humans or humans do not like to do; and exploration robots to explore other planets and uninhabitable places on earth.

According to Wolf and Bugmann [90], "Future service robots cannot be completely preprogrammed by the manufacturer. There are far too many possible tasks. In order for these robots to successfully learn and interact with people from the general public, they must be programmable by anybody (naïve users / without training) and not just by engineers, roboticists and computer scientists."

Bugmann [13] further elaborates by saying that "Any robot needs to be programmed by its user to become a functional device. For domestic robots, this poses a special problem, as their typical users are naïve in programming and unaware of mechanical and control issues. As for human servants, domestic robots need to learn the particular needs of their employers / owners and the particularities of their

environment. Various learning methods have been investigated, such as learning from demonstration, learning by reinforcement, etc. But none of these methods has the power that language has for communicating logical rules and procedural knowledge. Therefore, learning from verbal instructions will be an essential capability in future domestic robots." Bugmann calls programming a robot by verbal instruction, Instruction Based Learning (IBL) stating that, "Comparatively little research has been devoted to Instruction-based learning (IBL)" [13].

Although verbal instruction is an important part of how humans teach one another, it is not the only part. Humans use other modalities such as vision and gestures to teach each other. Therefore, it is logical that humans should use multiple modalities to teach future robots. For our research, we will expand Bugmann's definition of IBL to include other modalities.

1.2 Problem Statement

A great deal of research has been done on using multi-modal interfaces for computer systems, which is referred to as Human Computer Interaction (HCI). Table 1.1 illustrates how using multiple modalities, dialog management, context, and semantics improves recognition. Except for the research of Demirdjian *et al.* [25], those using multiple HCI techniques have lower error rates than those using only a single HCI technique. Even the research of Demirdjian *et al.* produced lower error rates when multiple techniques were used.

Researcher	HCI Techniques	Error Rate
Demirdjian *et al.* [25]	Vision & speech	0%
Demirdjian *et al.* [25]	Speech	5%
Demirdjian *et al.* [25]	Vision	8%
Morency *et al.* [61]	Gesture & dialog context	8%
Morency and Darrell [60]	Gestures & dialog state	9%
Quattoni *et al.* [67]	Vision & semantics	9%
Wang and Demirdjian [86]	Speech & gestures	12%
Webb *et al.* [87]	Speech & dialog state	17%
Metze *et al.* [59]	Speech, context, & gesture	17%
Morency *et al.* [61]	Gesture	22%
Saenko *et al.* [72]	Vision	34%
Eisenstein and Davis [31]	Linguistic context	34%
Bugmann [13]	Speech	40%

Table 1.1: Error Rates for Various HCI Techniques.

However, these techniques have not been applied to the same extent in Human Robot Interaction (HRI), or IBL. We believe that using these HCI techniques will improve IBL to some degree.

The hypothesis for this research is:

> **Applying the Human Computer Interaction (HCI) concepts of using multiple modalities, dialog management, context, and semantics to Human Robot Interaction (HRI) will improve the performance of Instruction Based Learning (IBL) compared to only using speech.**

We tested the hypothesis by simulating a domestic robot that can be taught to clean a house using a multi-modal interface.

What we are simulating does not exist today. We are looking in the future, when a user can go to a store in a mall and buy a domestic robot to bring home. Once the user gets it home, the first thing the user might say to the robot is, “Clean the house” and it is probably going to ask, “How do you want me to clean the house?” At this point, the

user is going to take the robot from room to room and tell it things like, “I want you to clean the sink, vacuum the floor, dust the cabinets, and then mop the floor once a week”. Today’s robots are not even close to having the capabilities of doing this. Today’s robots can take instructions verbally, but the verbal commands are simple, like: “Go to location A and pick up object B”. How do we get from the technology today to a robot that can dust a table when it is told, “Dust that table”? We propose it is by applying what we have learned about interacting with computer systems. Today, there are computer systems that respond to spoken commands. A lot of research has been done so that computer systems can take commands that are more sophisticated than the commands one can give to today’s robots. What we propose to do is apply these techniques, which have been successful at instructing a computer what to do and apply them to instructing a robot what to do.

1.3 Research Methodology

We tested the following modalities and contexts:

- Speech (e.g., “sweep the floor”)
- Pointing (what the user is pointing at, e.g., “broom”)
- Field of Vision (what the robot sees in the room, e.g., “broom”, “floor”, “mop”)
- Head Nodding (yes or no head nods)
- Real World Context (tasks the robot already knows how to perform, e.g., “sweep the floor with a vacuum once a week”)
- Dialog History (what the robot and user have been talking about recently, e.g., “clean the counters with a dry rag”)

Originally we planned to test the hypothesis with the following five scenarios:

1. Speech and Context (Real World and Dialog History) only
2. Speech and Context + Pointing (one type of modality)
3. Speech and Context + Field of Vision (another type of modality)
4. Speech and Context + Head Nodding (another type of modality)
5. Speech and Context + All

However, these do not test the impact of Context adequately. Therefore, we expanded the experiments to include seven scenarios:

1. Speech only (no modalities and no context)
2. Speech + Real World Context (one type of context)
3. Speech + Dialog History (another type of context)
4. Speech + Pointing (one type of modality)
5. Speech + Field of Vision (another type of modality)
6. Speech + Head Nodding (another type of modality)
7. Speech + All

During human-to-human interactions, the process of ensuring common understanding is called grounding. Clarifying questions are asked to ground the conversation. To test different modes of dialog management, we used three different strategies for grounding [83]:

1. **Optimistic** – The robot assumes grounding without any clarification. Optimism means that participants assume that their contributions have been understood.

2. **Cautious** – The robot engages in clarification when an utterance or gesture is ambiguous. Cautious participants wait until there is some kind of feedback (which may involve simply continuing with another relevant utterance) before assuming that their contributions have been understood**.**

3. **Pessimistic** – The participants always verify any answer that is not yes or no.

The purpose of this research is more than just demonstrating that a robot can be taught to perform a task. It is measuring how well the robot learned to perform the task, and how well it balanced asking questions with extrapolating from what the robot already knew.

Asking too many questions will frustrate the human, and will make the learning process longer. On the other hand, asking too few questions may result in the robot not learning how to perform the task correctly.

One can optimize the robot's learning process so that no clarifying questions are asked at the expense of not learning the task correctly; or one can optimize it so that the task is learned correctly at the expense of asking "too many" clarifying questions.

The learning process was evaluated using two metrics:

1. How many times did the robot ask a clarifying question?

2. How many times did the robot learn the task correctly?

All the 166 tasks in the corpus were tested in each of the 21 combinations of the three grounding strategies (i.e., Optimistic, Cautious, and Pessimistic) and seven scenarios of modality and context.

1.4 Organization

This dissertation is organized into eight chapters. Chapter 2 provides the background and reviews the relevant literature in the areas of HCI, HRI, and IBL. Chapter 3 presents the details of the research methodology that was adopted. Chapter 4 describes the system implementation in more detail. Chapter 5 explains the semantic integration of the modalities and contexts. Chapter 6 analyzes the experimental results and presents the findings. Chapter 7 examines using neural networks for semantic integration. Chapter 8 discusses the contributions of this research, limitations of the research, and future work.

Chapter 2 Background and Related Work

In this chapter, we will begin by discussing multi-modal Human Computer Interaction (HCI). Then, we will introduce Human Robot Interaction (HRI), which is a newer offshoot of HCI, and discuss how it is different from HCI. Next, we will introduce a specific taxonomy for classifying robots. We will use the taxonomy to identify robots where teaching the robot skills is important. After that, we will discuss the various methods of teaching the robot skills, including Instruction Based Learning (IBL). We will then focus on the limited research that has been done on IBL and the related challenges. Finally, we will explain how we believe that the techniques learned in HCI can be applied to address those limitations.

2.1 Human Computer Interaction (HCI)

Human Computer Interaction (HCI) has been around since the 1940's when the first computer, ENIAC, was built [88]. Much research has been done on HCI since then. Today, most people are familiar with HCI using a keyboard, mouse, and monitor. We will begin the discussion with multi-modal interfaces, where speech and vision modalities are introduced into HCI.

Pastra provides a theoretical basis for why multi-modal interfaces work [66]. She brings together Searle's theory of intentionality [73, 74], Harnad's symbol grounding problem [38], as well as arguments regarding the nature of images and language developed within different AI fields to produce a Double-Grounding theory that explains why speech and vision modalities work. Simply put, vision grounds speech

in the physical aspects of the world and speech grounds vision in the mental aspects of the world.

Bolt built the first known multi-modal interface in 1980 [5]. Bolt's interface could create, move, copy, remove, and name shapes using a combination of speech and pointing, e.g., "put that to the left of the green triangle," "copy that there," "call that the calendar," "move the calendar here". The system did fusion at the parse level. Every time the system recognized an anaphor or deictic reference, it would immediately see where the user was pointing and resolve the reference. The system also had an ability to learn new words. When the user said, "call that <name>," the system told the speech recognizer to switch from recognition mode to training mode so that the name that the user gave the object was learned. The system components were tightly integrated because of the way the system was designed. While performing fusion during speech, yields a straightforward implementation of fusion, gestures and speech are in general not synchronized, i.e., gesture precedes or follows a spoken reference. Assuming that they are synchronized forces the user to change his or her normal behavior to use the system. In other words, the system trains the user, which is not desirable.

Since Bolt's work, numerous multi-modal interfaces have been developed. A comprehensive summary of research in multi-modal computer interfaces conducted in 2004 listed 336 citations [57]. Several more recent research papers relevant to this dissertation are discussed in more detail below.

Chai *et al.* mapped gesture, dialog history, visible objects, and referring expressions from a speech utterance into four vectors and used these vectors to identify ambiguous references in speech [21]. In another effort, Chai *et al.* fused multi-modal inputs by using common data structures to represent utterances and gestures, and overlaying the data structures to disambiguate the utterance [20]. A variety of contexts, such as domain context and conversation context, were also used to enhance multi-modal interpretation. In a telephone banking call center application, Webb *et al.* used the state of the dialog to identify ambiguous references in speech, and achieved a 17% word error rate [87]. Wolf and Bugmann proposed using a touch screen to identify an ambiguous reference in speech in an application where users taught a computer to a play card game [90]. Morency and Darrell used the current dialog state of a Windows interface to clarify yes and no head nods with a 9% error rate [60]. Morency *et al.* also used dialog context to improve head nodding recognition of an embodied conversational agent from 78% to 92% [61]. A corollary of these results means that when expecting a yes or no answer, a robot could interpret yes and no head nods with 91% or 92% accuracy. Quattoni *et al.* developed a method of identifying objects (e.g., bed) in a scene (e.g., bedroom) using semantics to reduce the number of objects that need to be considered, with a 9% error rate [67]. For example, beds do not usually appear in bathrooms. A corollary of this is to use objects in a scene, or field of vision, to clarify the semantics of an utterance. Saenko *et al.* used a vision-only approach to detecting and recognizing spoken phrases (i.e., lip reading) to control a car stereo system with 66% accuracy [72].

Wang and Demirdjian's Virtual Home Desktop enables users to control a computer desktop system using gestures and speech with a 12% error rate [86].

MOWGLI (Multi-modal Oral With Gesture Large display Interface) is a speech and gesture multi-modal computer interface [19]. Carbini *et al.* describe two applications of MOWGLI. The first application is a Wizard of Oz cooperative story telling experiment named Virstory, where user speech-gesture actions are interpreted in order to cooperatively build a story with another person. The second application is a chess game where moves are completed through speech, gesture, or a combination of the two. MOWGLI is an untethered system that tracks head and both hand positions. The head position is used to estimate the pointing direction. MOWGLI behaves as a "speech-gesture mouse" similar to Bolt's "Put that there" system. Using both hands, a user can navigate within a large panoramic image. The pointing hand is used for displacement.

Demirdjian *et al.* describe a virtual studio application that allows a user to edit a virtual world and navigate around it [25]. The application uses speech, one-handed, and two-handed gestures to identify ambiguous references in speech. The error rate was 8% with vision only, 5% with speech only, and 0% with both vision and speech.

Flatscape is a collaborative situation map application for planning military missions [33]. It uses a mouse, eye tracker, gesture, and dialog history, to resolve ambiguity in spoken commands.

Robbins researched seven speech and gesture based multi-modal interfaces [69]:

1. Bolt's "Put-That-There" system
2. CUBRICON system – Maintains dynamic user and dialog models to improve interpretation of gesture-based and natural language speech multi-modal input.
3. XTRA system – Includes the use of user and dialog models while exploring the use of variable granularity in deictic gestures involved in a point-and-speak interface model.
4. QuickSet, – A reusable map-based speech and pen multi-modal interface framework that allows more complex symbol gestures for creating objects as well as spatial and pointing gestures.
5. IBM's Human-Centric Word Processor – A word processor that explores the benefits of using gesture and speech to edit dictated text.
6. Portable Voice Assistant – A modular architecture for developing web-based multi-modal applications.
7. Field Medic Information System – A portable multi-modal system consisting of a wearable hands-free speech interface augmented by a speech and gesture tablet computer interface.

Sharma *et al.* used gestures to disambiguate the speech in two crisis management systems: Crisis Management (XISM) and Dialog Assisted Visual Environment for Geoinformation (DAVE_G) [75].

An application developed by Englert and Glass, where a wireless phone user with a MDA utilizes pen and speech input to register and administer customer care campaigns, uses context and multi-modal input accuracy to determine what the user

means [32]. The system develops several hypotheses as to what the user meant based on the multi-modal input. Inconsistent hypotheses are discarded. The remaining consistent hypotheses are ranked with regard to the most probable meaning. The ranking reflects the input accuracy of a modality. Speech is more difficult to interpret than keyboard input. Gestures are more complex to interpret than speech input.

In an application where two avatars are solving a problem in an interactive gaming environment, Gorniak and Roy used knowledge of the environment and the situation to resolve ambiguities in a very efficient dialog – efficient in the sense that few words are spoken because both participants in conversation are aware of the environment and the situation [35].

Metze *et al.* describe an Augmented Table which allows several users at the same time to perform multi-modal, cross-lingual document retrieval of audio-visual documents [59]. The Augmented Table enhances multi-lingual speech recognition with context and a visual gesture recognition system, using tokens. The English word error rate was reported to be 17.2%; the sentence error rate was 20.7%; and the finalized goal rate was 71.5%.

Several researchers have studied the linguistic aspects of gestures. This research helps to temporally align gestures and speech.

Kosmopoulos and Maglogiannis propose a method for extracting mid-level semantics from sign language videos, by employing high-level domain knowledge [49]. The approach is applied to sign-language videos, but it can be generalized to any case, where semantically rich information can be derived from gesture.

By studying how weathermen point at weather maps and users answer questions like "where is the nearest parking lot", Kettebekov and Sharma discovered parallels between gestures and speech [43]. They identified gesture phonemes that form into morphemes (morphology), which in turn are used to form phrases (syntax), and finally those yield meaning (semantics). A study of 332 gesture utterances revealed that 93.7% of time the adopted gesture primitives were temporally aligned with the semantically associated nouns, pronouns, spatial adverbials, and spatial prepositions.

In another study, Eisenstein and Davis examined using speech to classify gestures into a taxonomy of gesticulation created by linguists [31]. The automatic gesture classification system is based solely on an n-gram model of linguistic context and is intended to supplement a visual classifier. The system achieved 66% accuracy on a three-class classification problem, which represents higher accuracy than human raters achieve when presented with the same information.

Max is a conversational agent that collaborates in cooperative tasks taking place in virtual reality. Max is the application in a cooperative construction task [50] and is a guide in a public computer museum [48].

Corradini *et al.* described a full system design similar to Max, except that input is real speech instead of keyboard, like Max [24]. Neither paper cited the other work.

2.2 Human Robot Interaction (HRI)

Human Robot Interaction (HRI) is a newer offshoot of HCI, and less research has been done on HRI. For example, a search of Google Scholar produced 1,300,000 hits

for “Human Computer Interaction (HCI)”, while “Human Robot Interaction (HRI)” only produced 24,500 hits [96].

Kiesler and Hinds described the difference between HRI and HCI as [45]:

1. People perceive robots differently than they do most other computer technologies. People’s mental models of autonomous robots are often more anthropomorphic than are their models of other systems.
2. Robots are mobile, bringing them into physical proximity with other robots, people, and objects. This means robots have to negotiate their interactions in a dynamic, sometimes physically challenging, environment.
3. Robots make decisions. They learn about themselves and their world, and exert some control over the information they process and actions they take. Because a robot operates in the physical world, it must adjust its decisions sensibly and safely to account for its abilities and the challenges in a given environment.

In 2004, Breazeal listed the following major laboratories and researchers working on HRI [6]:

- Brigham Young University (Mike Goodrich and Dan Olsen)
- Carnegie Mellon (Reid Simmons, Sara Kiesler, and Illah Nourbakhsh)
- Georgia Tech (Ron Arkin and Tucker Balch)
- Massachusetts Institute of Technology (Cynthia Breazeal, Rod Brooks, and Brian Scassellati)
- NASA Johnson Space Center (Rob Ambrose and Bob Savely)
- NASA Ames (Illah Nourbakhsh, Terry Fong, and Bill Clancey)

- Navy Research Lab, Washington DC (Alan Schultz, Dennis Perzanowski, and Greg Trafton)
- National Institute of Standards and Technology (Jean Scholtz)
- University of Massachusetts – Lowell (Holly Yanco)
- Mitre Corporation (Jill Drury)
- Stanford University (Pam Hinds, Hank Jones, and Ousamma Khatib)
- TRACLabs (David Kortenkamp)
- University of South Florida (Robin Murphy and Christine Lisetti)
- University of Southern California (Maja Mataric, Gaurav Sukhatme, and Stephan Schaal)
- Vanderbilt (Kaz Kawamura, Alan Peters, Julie Adams, and Mitch Wilkes)

Work is also being done at the University of Kansas, where Brown *et al.* designed a cognitive robot to use in four research projects studying the resolution of ambiguity in the following human and robotic systems [11, 10]: natural language understanding systems, active vision systems, memory retrieval systems, and robot reasoning and actuation.

Yanco and Drury proposed taxonomy for classifying HRI in 2002 and updated it in 2004. The categories of the taxonomy are [92]:

> **Task Type** – E.g., urban search and rescue, walking aid for the blind, toy, or delivery robot.

Task Criticality – High, medium, or low; where high means a failure affects the life of a human.

Robot Morphology – Anthropomorphic (having a human-like appearance), zoomorphic (having an animal-like appearance), or functional (having an appearance that is neither humanlike nor animal-like, but is related to the robot's function).

Ratio of People to Robots – Denoted as a non-reduced fraction, with number of humans over the number of robots.

Composition of Robot Teams – Homogeneous (robots are all of the same type) or heterogeneous.

Level of Shared Interaction Among Teams – One human, one robot; one human, robot team; one human, multiple robots; human team, one robot; multiple humans, one robot; human team, robot team; human team, multiple robots; or multiple humans, robot team.

Interaction Role (of Human) – Supervisor, operator, teammate, mechanic/programmer, or bystander. One or more of these values can be assigned.

Type of Human-Robot Physical Proximity – Avoiding, passing, following, approaching, touching, or none (human and robot are not collocated).

Decision Support for Operators – Four subcategories of available sensor information, sensor information provided, type of sensor fusion, and pre-processing.

Time/Space Taxonomy – Divides human-robot interaction into four categories based on whether the humans and robots are using computing systems at the same time (synchronous) or different times (asynchronous) and while in the same place (collocated) or in different places (non-collocated).

Autonomy Level / Amount of Intervention – The Autonomy Level measures the percentage of time that the robot is carrying out its task on its own; the Amount of Intervention measures the percentage of time that a human operator must be controlling the robot. These two measures sum to 100%.

Recently HRI has been focused on the following areas:

Robot Hosting – Robots that conduct demonstrations or answer questions for humans. For instance, Sidner *et al.* developed a penguin robot that demonstrates itself and a device called an IGlass cup [77, 76]; and Den Os and Boves developed a sales avatar that uses eye contact, gaze, body posture, drawing with a pen, and speaking to assist shoppers in designing bathrooms [26].

Teleoperation – Robots that are remotely operated by humans and have varying degrees of autonomy. Examples are the analysis of HRI for Urban

Search and Rescue (USAR) robots by Yanco *et al.* [91] and the research on polymorphic robots (shape changer) used in USAR by Drury *et al.* [30, 29].

Service Robots – Robots that perform autonomous tasks for humans. An example is COGNIRON. COGNIRON is a European Union project to develop "a social robot companion" [3]. COGNIRON's mission statement is: "Developing cognitive robots whose "purpose in life" would be to serve humans as assistants or "companions". Such robots would be able to learn new skills and tasks in an active open-ended way and to grow in constant interaction and co-operation with humans" [3]. Another example of a service robot is a speech-controlled wheelchair, which was developed by Tellex and Roy [80]. The speech-controlled wheelchair understands high-level natural language commands such as "take a left".

pHRI – Physical Human-Robot Interaction, which as defined by Khatib *et al.*, involves haptic, force, neural, and other physical interactions between humans and robots [44].

2.3 Robot Learning

We studied robots that need to learn skills from humans for a variety of reasons:

- The skills are too complex to program (e.g., auto mechanic robot).
- The robot needs to learn the particular needs of the human it serves (e.g., domestic robot).

- The robot needs to learn the particularities of its environment (e.g., exploration robot).
- The subject matter expert is not a computer scientist or robot engineer, but is a human instructor adept at teaching the skills to humans.

Using Yanco and Drury's taxonomy, the key attributes of these robots are [92]:

Interaction Role (of Human) = Supervisor, as the human tells the robot what to do, but the robot plans and carries out the task.

Time/space taxonomy = Asynchronous (different times from human) and non-collocated (from human)

Autonomy Level / Amount of Intervention = Autonomy near 100% and Intervention near 0%.

In these cases, the human teaches the robot a task to be performed later.

We will define "Robot Learning" as the process of humans teaching skills to these types of robots.

There are various types of Robot Learning and many ways to classify them. Bruemmer proposed the following classifications [12]:

1. Artificial Neural Networks – A supervised, learning-with-a-trainer approach where knowledge is learned by adjusting weights between nodes of a neural network.

2. Reinforcement Learning – An unsupervised, learning-with-a-critic approach where mappings from percepts to actions are learned inductively through trial and error.
3. Evolutionary Learning – An unsupervised, learning-with-a-critic approach where controllers are derived deductively by alterations to an initial population of program code.
4. Learning by Imitation – A design methodology that uses a biologically inspired developmental paradigm to enable learning by emulation. Imitative Learning is more an approach than a specific computational means. Theoretically, it might involve any of the means described above.

Other researchers have grouped the first three together and called it Learning by Reinforcement. Bruemmer also did not discuss Instruction Based Learning.

For our work, we will classify the types of Robot Learning as:

RbD – Robot Programming by Demonstration, also referred to as Learning by Imitation. A non-verbal technique usually used to teach new motor skills to a robot. This approach works best for learning multidimensional and non-linear tasks.

Learning by Reinforcement – Trial and error method with the teacher, environment, or both providing reward and punishment when the goal is achieved, or not achieved. Learning by reinforcement is guided more by the robot than the teacher.

IBL – Instruction Based Learning. A technique using conventional teaching methods, where demonstration is difficult, e.g., driving directions, house cleaning directions (not to be confused with individual sub-tasks, like dusting a table with a dust cloth, which are very amenable to RbD and probably the most straightforward way to convey that skill to a robot). This technique goes beyond the other techniques by generating knowledge representations that the user can interrogate.

IBL is the subject of this dissertation; however, before we examine the work of other researchers with IBL, we will review the other two types of Robot Learning.

2.4 Robot Programming by Demonstration (RbD)

Taycher *et al.* proposed a new method for tracking 3D human motion [79]. They used a database of rendered images to evaluate their approach.

Klingspor *et al.* reviewed the state of the art in 1997 and identified two requirements for RbD [46]:

1. The human teacher must be competent in the skill being demonstrated.
2. The robot acquiring the skill must have adequate sensors and degrees of freedom to perform the skill.

Nicolescu and Mataric taught a Pioneer 2DX robot to reach a set of colored columns in a specific sequence as demonstrated by a human [63, 64].

Calinon *et al.* demonstrated another example of learning by imitation [17]. The experiment starts with the human and the robot facing each other across a table. There is a red and a green dot on the table. The human reaches for each dot alternatively

with the left or right arm. The robot then imitates the human. In another experiment, they taught a robot to move chess pieces by the user moving the robot's arm through the task [18].

Alissandrakis *et al.* addressed the RbD problem where the robot and the human may not share the same degrees of freedom, morphology, constraints, etc. [1]. In another work, Calinon and Billard explored the issue of recognizing, generalizing and reproducing arbitrary gestures [16]. Breazeal *et al.* used biologically inspired mechanisms to design a robot capable of learning how to imitate facial expressions from imitative games played with a human [8].

2.5 Learning by Reinforcement

Thomaz *et al.* looked at what they called, Socially Guided Machine Learning, in Sophie's world [81]. In their experiment, eighteen participants were asked to play a video game, in which the participant's goal was to get the robot, Sophie, to learn how to bake a cake on her own. The robot achieved this goal through a series of actions (e.g. pick-up eggs, turn left, use-eggs on bowl, etc.). The participants decided when they were finished with training Sophie. At this point, the experimenter tested the robot; and their game score was the degree to which Sophie finished baking the cake. They found teachers favored reward over punishment and had a tendency to use guidance in addition to reward and punishment.

Rosenstein and Barto examined what they called, Supervised Actor-Critic Reinforcement Learning [70]. They defined Supervised Actor-Critic Reinforcement Learning as reinforcement learning where positive and negative feedback from the

environment are augmented with teacher feedback and guidance. They demonstrated their methods on a seven degree-of-freedom manipulator arm. Lopes and Wang used reinforcement learning to teach CARL (Communication, Action, Reasoning and Learning in robotics) to identify images of a person, trashcan, or triangle by asking the teacher [54].

2.6 Instruction Based Learning (IBL)

Toptsis *et al.* proposed a multi-modal dialog system for a mobile robot called BIRON (Bielefeld Robot Companion) developed under the auspices of the COGNIRON project [82]. However, it was limited to the so-called home-tour, where a user shows a robot companion around his or her private home and teaches the robot important locations and objects, using speech and gestures.

In another COGNIRON project, Green *et al.* analyzed miscommunication in a multi-modal interface with a service robot [37]. The application was the home-tour. Their observations from twelve user sessions revealed two major areas of miscommunication:

1. Users misunderstand the robot's functionality
2. Ill-timed feedback from the robot with respect to the situation

Their solution to functionality mismatch is to train the user on the capabilities of the robot. We argue that another solution is to just allow the user to become familiar with the robot's capabilities through interaction in the same way two humans familiarize themselves with each other's capabilities. This of course means that the robot must be capable of engaging in a rich dialog. A lot of their ill-timed feedback

resulted from the slowness of the system. Part of our research studied the effect of various modes of feedback on learning.

Dominey *et al.* demonstrated the use of speech in three applications, namely, command, interrogate, and teach, on two robot platforms: Event Perceiver and AIBO ERS7 [28]. Although they looked at IBL, the interaction did not include gestures and was limited to the capabilities of the robots. These capabilities were simple. Event Perceiver was taught the names for various objects (e.g., "This is a stack") and AIBO ERS7 was taught to associate perceptual events with behaviors (e.g., "Head-touch to Bark"). Additionally, the dialog was verbose and robot controlled. AIBO's behaviors could not be triggered by temporal events (e.g., on Wednesdays) nor could they be combined into higher-level tasks (e.g., "sing and dance" meaning "bark" and "spin around").

Wilske and Kruijff developed a robot that could handle indirect speech acts [89]. Indirect speech acts are best illustrated in Asimov's classical story "Little Lost Robot" (1947), where a robot was told "Get lost!" and then tried to lose itself, instead of just backing-off. Their system was limited to fulfilling direct or indirect requests for bringing objects from locations where the robot knew such objects could be found.

Breazeal *et al.* looked at IBL as collaboration between a human and a robot, instead of the classical teacher-student relationship [7, 52]. Their research platform is Leonardo ("Leo"), a humanoid robot with 65 degrees of freedom. Leo uses gestures without speech to communicate. Their work draws from Joint Intention and Situated

Learning Theory, which is the basis for many HCI dialog managers. According to Cohen and Levesque's Joint Intention Theory, two or more partners doing something together have "joint intention", if they share the same goal and plan of execution [22, 23]. Breazeal *et al.* use Vygotsky's Zone of Proximal Development from Situated Learning Theory [84], which theorizes that a child learns a new skill from a teacher through the process of scaffolding. In HRI scaffolding, the human provides structure and assists with the new activity, so that the robot can achieve something it cannot do on its own. In their application, Leo learned to associate names with objects and then to perform actions on the labeled objects. They successfully taught Leo both primitive tasks (e.g., turning a button on) and compound tasks (e.g., turning a set of buttons on then off, and turning a button on as a single action). Although effective in teaching skills to the robot, the close collaboration between the robot and human will not work as well with more autonomous robot applications such as driving instructions or house cleaning.

In other research, Breazeal *et al.* used Leo again to examine two types of non-verbal communication, explicit (e.g., robot nods its head or points to an object) and implicit (i.e., gaze direction) [9]. They found that people infer task-relevant "mental" states of Leo not only from explicit communication, but also from implicit communication. One limitation of their research is the limited grammar to parse incoming phrases. These include simple greetings, labeling the buttons in the workspace, requesting or commanding the robot to press or point to the labeled buttons, and acknowledging that the task is complete. Other limitations are the lack of

speech synthesis, and the fact that they used gestures to communicate to the human, but did not interpret similar gestures from the human.

Like Breazeal *et al.* [7, 52], Lopes and Teixeira used guided instruction to teach CARL (Communication, Action, Reasoning and Learning in robotics) to turn a corner [53]. They explained guided instruction as teaching a task, in which the human explains one or two steps of the task, the robot tries them out, and then the human explains a little more, and so on. Their implementation used speech only and did not include teaching CARL compound tasks.

Zhang and Knoll defined two types of IBL [93]:

- Front-End Approach – The robot system receives instructions that completely specify a task the human wants to be performed. The input is analyzed and the necessary actions are taken in a subsequent separate step. Upon completion of the task, the system is ready for accepting new input. This approach is ideal for systems that have to deal only with a limited set and scope of tasks, which do not vary much over time either. Inadvertent changes of the environment resulting from the robot's actions, which would require a reformulation of the problem, cannot be considered because neither the programmer nor the instructor can foresee all of these states.

- Communicator Approach – The robot and human engage in a collaborative dialog in order for the robot to learn a task, much like that proposed by Breazeal *et al.* and Lopes and Teixeira.

The Communicator Approach was used to teach a two-arm multi-sensor robot system to assemble parts using human tools (e.g., screw driver, wrench) [93]. Like the other Communicator Approach examples above, this approach does not lend itself to imparting temporal conditions. Although complex, assembling parts is not as complex as cleaning a house. Using this method to teach a robot to clean a house, drive a car to the store, or search for life on a remote planet would be very tedious. Users are going to expect robots of this type to have some high-level primitives built-in.

Bugmann *et al.* implemented a system for teaching a robot to navigate through a city with a natural language interface [13, 15, 51]. The corpus contained 144 routes, which were produced by 24 subjects instructing six routes each [14]. From this corpus, they identified fifteen primitive actions. They found the primitive actions to be rather complex procedures for a robot, requiring the visual localization of landmarks, the identification of navigable areas, and the planning of a path to reach a landmark. Their HRI was limited to speech and did not include vision. Basing their natural language interface on a corpus (as opposed to designing the interface so it would be easy to program) has both good and bad points. It is good because the user is allowed to use unconstrained language. It is bad because new primitives have to be added as the number of teachers grows. Experimental data showed that one new function would be added to the functional lexicon for every 38 additional route instructions collected. Although a Front-End Approach (as defined by Zhang and Knoll), Bugmann *et al.* used high-level primitives to avoid the shortcoming of not being able to predict all the idiosyncrasies of the environment.

MacMahon [55] proposes MARCO, a modular architecture for reasoning about and following route instructions, which can be extended to other IBL applications. Like Bugmann, MacMahon's primitives are fairly complex and high level and leave a lot to be interpreted during execution. MARCO is a design. MacMahon never indicated that it had been implemented. MARCO deals with the robot interpreting speech from the human. MARCO does not include gesture recognition or generation by the robot, speech generation by the robot, or dialog management.

Green and Severinson-Eklundh, examined robot generated gestures in IBL by placing a life-like character, CERO, on top of a mobile robot [36]. CERO provided a visible direction for the robot and low-level visual feedback to supplement the spoken feedback issued by the dialog system. The dialog system was fairly advanced, using dialog acts in different situations (e.g., question, answer, require, repair). CERO was able to perform a small set of tasks:

- GOTO: Navigation from one location to another
- DELIVER: Carry an object while navigating
- FETCH: Navigate to a location and address another user in order to get an object and carry it back

Green and Severinson-Eklundh noted that their system did not handle out of sequence commands well. Others have referred to the ability to handle out of sequence commands as "accommodation" [83]. CERO did not interpret gestures from

the human and was only able to handle simple commands. CERO did not have the capability of learning compound tasks.

Suomela and Halme's WorkPartner has similarities with the work in this dissertation [78]. WorkPartner is a test bed for service robots. It is targeted for light outdoor applications like property maintenance, gardening, and light forestry tasks. It understands human speech and gestures and uses speech to carry on a dialog with humans. It can be taught compound tasks based on previously taught subtasks. Limitations of WorkPartner are:

- WorkPartner's language understanding is very limited and does not allow free flowing dialog between the human and robot.
- Learning is limited to a large number of very small subtasks, which the human then combines and teaches to the robot as a new task.
- WorkPartner does not use gestures to communicate with the human.

Mavridis and Roy's work also has similarities with our work [58, 71]. They have implemented a manipulator robot arm with seven degrees of freedom, equipped with force feedback actuators, a gripper with force-sensitive touch sensors integrated into each finger tip, joint angle encoders, and dual cameras mounted around the gripper. The robot's world consists of a tabletop on which various objects are placed and manipulated. They use what they call an "amodal" interface that integrates both language and sensor-derived information about the situation. For example, the robot can acquire parts of situations either by seeing them or by "imagining" them through

descriptions given by the user: “There is a red ball at the left”. These situations can later be used to create mental imagery, thus enabling bi-directional translation between perception and language. The robot is able to pass the first two parts of the Token Test [27], a standard test used to assess early-situated language skills in children. The robot is also able to answer questions about the present and past, act on objects and locations, and integrate verbal with sensory information about the world.

Blythe and Reilly simulated a household robot agent, Mr. Fixit, who could, along with other tasks, vacuum and clean up broken cups [4]. Although Blythe and Reilly’s research did not involve IBL, it is included because it is a good description of the application we propose to research.

According to Blythe and Reilly, the typical household is a challenging environment for a robot, partly because of the presence of other agents. The robot must sense and respond to the other agents as it completes its task. In addition, other agents may move furniture and create new cleaning tasks. This leads to a less structured environment, for example, than a nuclear power plant or a road-following task. Thus, it will be impossible to pre-program a vacuuming and cleaning robot to perform “blind”, without sensing the environment and monitoring the progress of tasks.

Some of the changes in the environment will demand reactive responses from the robot, such as interrupting a less important task to pick up a piece of trash. Thus, a vacuuming robot must be flexible enough in its design to respond intelligently to a changing set of goals, as well as a changing environment. Such a robot needs to

display a range of capabilities not typically found in a single system. Deliberative systems that embody powerful techniques for reasoning about actions and their consequences often fail to guarantee a timely response in time-critical situations. In addition, reactive systems that respond well in time-critical situations typically do not provide a reasonable response in situations unforeseen by the designers.

Reactive systems have traditionally been more successful than deliberative ones in controlling agents in dynamic domains, like a typical household. Building a reactive system, however, can be a complex and time consuming endeavor because of the need to pre-code all of the behaviors of the system for all foreseeable circumstances. An efficient reactive system is also likely to have a narrow area of applicability, for instance a specific house and task set. Deliberative systems are best suited to long-term, off-line planning for static environments, but not for controlling an autonomous agent operating in a dynamic environment. However, deliberative systems are often more robust to varying domains and sets of interacting goals because they employ some form of forward projection.

2.7 Assessment of IBL Research

The previous section identified a number of research projects on IBL. Each had its limitations. The limitations that we address in this dissertation are:

1. Use of only one modality, or the use of multiple modalities in only one direction (i.e., robot to human, or human to robot).
2. Limited dialog.
3. Robot controlled dialog.

4. Too many clarifying questions.
5. Unable to accommodate out of sequence utterances from the human.
6. Limited to simple tasks such as identifying objects, fetching objects, or turning lights on and off.
7. Only looked at what robots can do today.
8. Unable to learn behaviors triggered by temporal events (e.g., every Wednesday).
9. Unable to combine learned tasks into higher-level compound tasks.
10. Unable to teach autonomous behaviors, such as driving a car or cleaning a house.

A great deal of research has been done on using multi-modal interfaces to computer systems (HCI). However, the best results are achieved when context, situation, dialog state, and environment are also taken into account. Using multiple modalities, dialog states, context, and semantics improves recognition over speech recognition alone in these applications (Table 1.1). We believe that using these HCI techniques will improve IBL in the same manner.

2.8 TRINDIKIT

The TRINDI project proposed an "information state theory of dialog modeling" [83]. The TRINDI approach combines the following two approaches, using the advantages of each:

1. Finite state machine – Dialog states are usually viewed as viable for simple scripted dialogs.
2. Plan-based approach – Though more complex and difficult to implement, this approach is seen as more amenable to flexible dialog behavior. Plan-based

approaches are also criticized as being more obscure because of the large amount of procedural processing and lack of well founded semantics for plan related operations.

The TRINDI information state theory of dialog modeling consists of the following components:

1. A description of the informational components of the information state, e.g., participants, beliefs, common ground, and intentions.
2. Formal representations of the above components, e.g., as typed feature structures, lists, sets, propositions, or modal operators within a logic.
3. A set of dialog moves that triggers the update of the information state. These are generally also correlated with externally performed actions, such as particular natural language utterances.
4. A set of update rules that govern updating of the information state, given various conditions of the current information state and performed dialog moves. Some of these rules also select particular dialog moves for the system to perform in the case of a system participating in a dialog, rather than just monitoring one.
5. A control strategy for deciding which rules to select at a given point from the set of applicable ones. This strategy can range from something as simple as "pick the first rule that applies" to more sophisticated arbitration mechanisms based on game theory, utility theory, or statistical methods.

The TRINDI project has developed a toolkit, called TRINDIKIT, that is for building and experimenting with dialog move engines and information states [83].

The TRINDIKIT package also specifies formats for defining information states, update rules, dialog moves, and associated algorithms. It further provides a set of tools for experimenting with different formalizations of implementations of information states rules and algorithms.

The TRINDIKIT is implemented in SICStus Prolog. It works best with SICStus Prolog 3.8 or later versions. It is available free from the web [102].

2.9 Multi-Modal Interface Tools

Flippo *et al.* propose a multi-modal framework using a variety of modalities and methods for ambiguity resolution, and a novel approach to multi-modal fusion, where 92% of the application is reusable code [33].

The Extended Multi-Modal Annotation (EMMA) language annotates XML code with multi-modal features [32].

2.10 HRI Metrics

Olsen and Goodrich proposed the following metrics for evaluating HRI in a USAR application [65]:

1. Task Effectiveness (TE) – A measure of how well a human-robot team accomplishes some task.
2. Neglect Tolerance (NT) – A measure of how the robot's current task effectiveness declines over time when the user neglects the robot.
3. Interaction Effort (IE) – The amount of time required to interact with the robot.

4. Robot Attention Demand (RAD) – A unit-less quantity that represents the fraction of a human's time that is consumed by interacting with a robot. RAD = IE / (IE + NT).
5. Free Time (FT) – The fraction of the task time that the human does not need to pay attention to the robot. FT = 1.0 – RAD.
6. Fan-Out (FO) – An estimate of the number of robots that a user can effectively operate at once. FO = 1 / RAD = (IE + NT) / IE.

Except for the first metric, TE, these metrics are appropriate for measuring remote operation HRI, like USAR, but do not provide any insight into IBL.

Because IBL deals with autonomous behavior, NT and IE should be zero by definition. RAD becomes undefined when NT and IE are zero. If we assume NT and IE are very small numbers instead, RAD becomes almost zero and FT and FO become almost one, which makes NT, IE, RAD, FT, and FO meaningless.

TE, which measures how well a human-robot team accomplishes some task, is the only relevant one for IBL.

2.11 Task Corpus

Developing a multi-modal corpus was a key activity in this research. Potential sources for a corpus are household task training manuals, articles on "how to clean a house", domestic servant training video, and Home & Garden TV shows on house cleaning.

The list of primitives was developed from the corpus instead of the other way around. This approach to the definition of the robot's functionality and natural-language interface has been described as "corpus-based robotics" [14].

According to Wolf and Bugmann [90], speech transcription can be done using tools such as Transcriber. This produces a time stamped XML text corresponding to a recorded sound file. They are currently investigating if there are similar tools for the transcription and annotation of signs or gestures done in a card game on a touch screen. Otherwise, they will develop dedicated software. The task of such software is to annotate the raw recording of trajectories on the screen with high-level "sign tags", such as pointat(AceClubs) or turnover(AceHearts).

Chapter 3 Research Methodology

3.1 Approach

The research approach in this dissertation is:

1. Develop a computer system to translate speech and gestures into tasks for a robot to execute.
2. Develop the system by integrating off-the-shelf components with custom written software.
3. Generate lesson plans to teach the system tasks.
4. Using speech and gestures, input tasks into the system.
5. Measure how well the system learned the tasks.
6. Vary the parameters in the system to measure the impact of clarifying questions on the accuracy of learning tasks.
7. Introduce vision system recognition errors to simulate a more real environment.
8. Use the measurements to either prove or disprove the research hypothesis.

This research will not include simulating the robot.

In order to mitigate the effects of not being able to tell whether the robot's learned capabilities are due to the quality of the human instructor versus the quality of the system, only one human instructor is used in the experiments.

3.2 System Architecture

Figure 3.1 is a diagram of the system architecture.

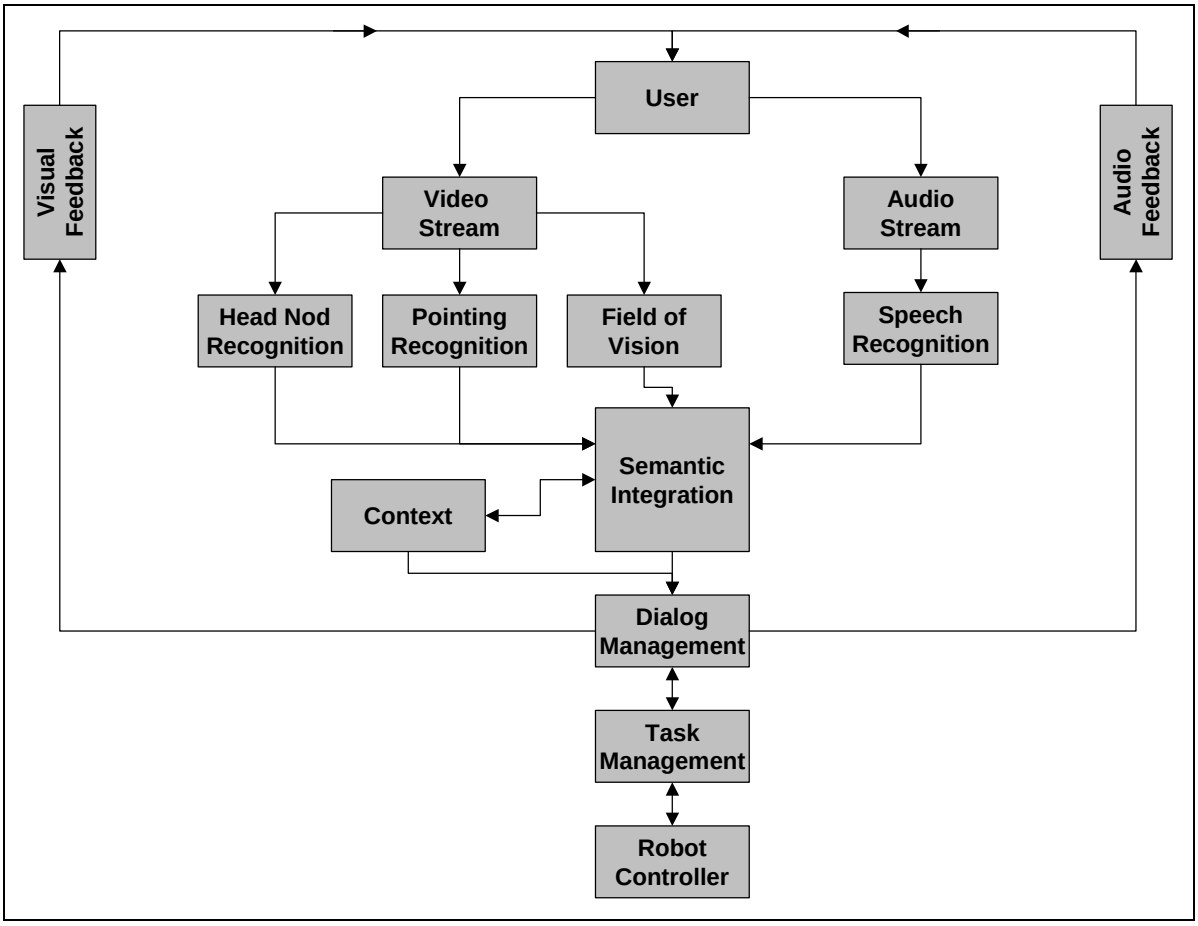

Figure 3.1: System Architecture.

The system architecture is a blend of MIND [21, 20], the XISM Framework [75], and MARCO [55]. Each component is briefly described below. The implementation is described in detail in a later chapter.

- **User** – Teaches the robot how to clean the house.
- **Audio Stream** – Utterances from the user.
- **Speech Recognition** – Translates the utterances into words.
- **Video Stream** – Robot's vision system.
- **Head Nod Recognition** – Recognizes yes and no head nods.

- **Pointing Recognition** – Captures objects selected by pointing. Temporal order of selection is maintained.
- **Field of Vision** – Captures objects that are visible to the robot. Temporal order of selection is NOT maintained.
- **Context –** Provides real world context and dialog history.
- **Semantic Integration** – Fuses nodding, pointing, field of vision, speech, real world context, and dialog history to resolve ambiguities in what the user says.
- **Dialog Management** – Manages dialog and resolves any remaining ambiguities through clarifying questions.
- **Visual Feedback** – Visual feedback from the robot to the user, such as head nodding and pointing.
- **Audio Feedback** – Generates speech from the robot.
- **Task Management** – Determines when tasks should be executed and breaks tasks into simple subtasks for the various subsystems of the robot to execute.
- **Robot Controller** – Mechanical and sensory systems of the robot used to execute tasks requested by the user.

The Dialog Management module works with all the modules above it in Figure 3.1 to translate the user's speech and gestures into tasks. We audited what the Dialog Management module sent to the Task Management module. This let us measure how well the Dialog Management module and all the modules above it did at correctly

interpreting the user's speech and gestures. By adjusting parameters in those modules, we tested permutations of speech, visual cues, and grounding strategies. Those modules encompass all the HCI techniques, which we believe have not been applied to HRI.

The execution of the tasks by the Task Manager and Robot Controller has already been covered in other research done by Blythe and Reilly [4].

3.3 System Implementation

The system was implemented by integrating software available from other researchers, commercial software, and software developed as part of this dissertation. The mapping of the software packages to the components of the system is shown in Table 3.1.

Component	Description	Utilized Software			Developed Software
		e-Speaking	Phoenix	Framework	
User	Teaches the robot how to clean the house				
Audio Stream	Utterances from the user				
Speech Recognition	Translates the utterances into words				
Video Stream	Robot's vision system				
Head Nod Recognition	Recognizes yes and no head nods				
Pointing Recognition	Captures objects selected by pointing and temporal order of selection is maintained				
Field of Vision	Captures objects that are visible to the robot and temporal order of selection is NOT maintained				
Context	Provides real world context and dialog history				
Semantic Integration	Fuses nodding, pointing, field of vision, speech, real world context, and dialog history to resolve ambiguities in what the user says				
Dialog Management	Manages dialog and resolves any remaining ambiguities through clarifying questions				
Visual Feedback	Visual feedback from the robot to the user, such as head nodding and pointing				
Audio Feedback	Generates speech from the robot				
Task Management	Determines when tasks should be executed and breaks tasks into simple subtasks for the various subsystems of the robot to execute				
Robot Controller	Mechanical and sensory systems of the robot used to execute tasks requested by the user				

Table 3.1: Mapping of Software Packages to System Components.

Each of the software packages is described below.

3.3.1 e-Speaking

The Speech Recognition component translates the user's utterances into a parsed list of words and passes it to the Semantic Integration component to be combined with other modalities and context. It is composed of two parts: a standard Automatic Speech Recognizer (ASR) and a parser.

The ASR selected for the system was e-Speaking, which is a shareware product published by e-Speaking.com [99]. e-Speaking was installed and trained using the speech recognition tools of Windows XP [98] and Microsoft SAPI 5.1 [97].

3.3.2 Phoenix Parser

The system parses the speech from the ASR with the Colorado University Phoenix parser [95]:

> "The Phoenix parser is designed for development of simple, robust Natural Language interfaces to applications, especially spoken language applications. Because spontaneous speech is often ill-formed and because the recognizer will make recognition errors, it is necessary that the parser be robust to errors in recognition, grammar and fluency."

3.3.3 Flippo Framework

The Flippo Framework is a multi-modal framework enabling rapid development of applications using a variety of modalities and methods for ambiguity resolution [33]. The framework uses an application-independent fusion technique that can be easily augmented to support application-specific demands as well as new modalities. The fusion algorithm separates the three parts of fusion: obtaining data from modalities, fusing that data to yield an unambiguous meaning, and calling application code to take an action based on that meaning. Separation of these three tasks makes the framework applicable to a wide range of applications and modalities.

The Framework is written in Java 1.4, and was provided for this work [33].

3.3.4 Developed Code – Dialog Management

It was originally proposed to use TRINDIKIT from the TRINDI project to implement the Dialog Management component [83]. However, we decided to develop the Dialog Management in Java for the following reasons:

- TRINDIKIT is implemented in Prolog, which would have complicated the integration with the Java-based Flippo Framework, which the other components are implemented in.
- The Flippo Framework included a dialog management component that was easily replaced with the more complicated dialog management software needed.

3.3.5 Developed Code – HRI Environment Simulator

The HRI Environment Simulator is a Java application developed to simulate various components of the system and to act as a console for conducting the research experiments. The various buttons, pull-down menus, and text fields will be described throughout the rest of the dissertation.

The video components of the system (Video Stream, Head Nod Recognition, Pointing Recognition, and Field of Vision) were not implemented because of difficulties with video processing, and instead they were simulated with the HRI Environment Simulator.

Similarly, the Visual Feedback, Audio Feedback, Task Management, and Robot Controller components were not implemented because they are not germane to the research problem. The output of the Dialog Management module to the Visual

Feedback, Audio Feedback, and Task Management modules are displayed as text on the HRI Environment Simulator screen, so the results of the experiments can be observed.

3.4 Robot Simulation

As described earlier, avatars have been used to simulate robots [50, 48, 24]. Simulation could be used to verify how well the robot learned a task. However, a more objective method of verification might be to examine the robot's memory. One advantage of verifying with a simulation is that the simulation might reveal a unique and unexpected implementation of a task by the robot. Examining the robot's memory for expected implementations might miss such an unexpected implementation.

3.5 Task Corpus

A task is the basic unit of work performed by the robot. A Primitive Task describes an **action** (e.g., clean, wash, dust) taken on an **object** (e.g., sink, dish, bed, couch) using none, one, or more **tools** (e.g., wash rag, vacuum cleaner) under a specified **condition** (e.g., daily, in the spring).

Compound Tasks are composed of multiple Primitive Tasks and are referred to in a conversation as a single task. For example, the Compound Task "clean the kitchen daily" might be composed of the following Primitive Tasks:

1. Load the dishwasher
2. Wipe down the sink with a sponge
3. Wipe down the stove top with a sponge

4. Wipe down the counters with a sponge
5. Vacuum the floor with a sweeper

Compound Tasks can also include other Compound Tasks.

The corpus of tasks to teach the robot was developed by performing an on-line search (using Google) on "how to clean a house", which yielded numerous results with the following characteristics:

- Multiple authors.
- Covering daily and seasonal cleaning tasks.
- Instructions are high-level complex tasks explained in only a few words that a human easily understands, like "sweep the floors".
- The robot's vision system will need to be able to recognize a large variety of household supplies, equipment, appliances, furniture, etc.

A total of 166 tasks were transcribed from these Web sites and grouped into 12 Compound Tasks:

1. Clean the kitchen daily (5 Primitive Tasks and 1 Compound Task)
2. Clean the bathroom daily (9 Primitive Tasks and 1 Compound Task)
3. Clean the bedroom daily (4 Primitive Tasks and 1 Compound Task)
4. Clean the family room daily (9 Primitive Tasks and 1 Compound Task)
5. Clean the living room daily (9 Primitive Tasks and 1 Compound Task)
6. Clean the foyer daily (9 Primitive Tasks and 1 Compound Task)
7. Clean the kitchen in the spring (25 Primitive Tasks and 2 Compound Tasks)
8. Clean the bathroom in the spring (27 Primitive Tasks and 2 Compound Tasks)

9. Clean the living room in the spring (26 Primitive Tasks and 2 Compound Tasks)
10. Clean the bedroom in the spring (23 Primitive Tasks and 2 Compound Tasks)
11. Clean the dining room in the spring (11 Primitive Tasks and 4 Compound Tasks)
12. Clean the house every day (8 Primitive Tasks and 1 Compound Task)

The 9 Primitive Tasks in Compound Tasks 4, 5, and 6 are identical and are counted only once in the 166 total.

Compound Tasks can be thought of as high-level tasks (such as, "clean the dining room in the spring"), which are composed of steps, or sub-tasks, which can be more Compound Tasks, or Primitive Tasks, which have no sub-tasks.

As an example, Compound Task 11 (clean the dining room in the spring) is composed of 11 Primitive Tasks and 4 Compound Tasks in the following steps:

1. Dust down the ceiling with a feather duster (Primitive Task 1)
2. Dust down the corners with a feather duster (Primitive Task 2)
3. Dust and clean all wall art (Primitive Task 3)
4. Dust and clean the Ceiling Fan (Primitive Task 4)
5. Take down draperies (Primitive Task 5)
6. Take down curtains (Primitive Task 6)
7. Wash down the dining table (Compound Task 2, "clean the dining room in the spring" is Compound Task 1)
8. Wash down the chairs (Compound Task 3)
9. Wash down any other furniture (Compound Task 4)
10. Clean the carpets (Primitive Task 7)

11. Clean the rugs (Primitive Task 8)

Compound Task 2 (wash down the dining table) is composed of the following sub-tasks:

1. Clean wood with damp cloth (Primitive Task 9)
2. Oil wood with furniture oil (Primitive Task 10)

Compound Task 3 (wash down the chairs) is composed of the following sub-tasks:

1. Clean wood with damp cloth (Primitive Task 9 – same as in Compound Task 2)
2. Oil wood with furniture oil (Primitive Task 10 – same as in Compound Task 2)
3. Spot clean upholstery with spot cleaner (Primitive Task 11)

Compound Task 4 (wash down any other furniture) is composed of the following sub-tasks:

1. Clean wood with damp cloth (Primitive Task 9 – same as in Compound Task 2)
2. Oil wood with furniture oil (Primitive Task 10 – same as in Compound Task 2)

3.6 Lesson Plans

For the experiments, the 12 Compound Tasks were combined into 10 Lesson Plans to teach the robot. Compound Tasks 4, 5, and 6 (i.e., the ones with identical Primitive Tasks) were combined into one Lesson Plan, and the rest were assigned to their own Lesson Plan.

As explained later, at the end of the teaching process, after the robot has asked all the clarifying questions and understood all the parts of the Task (i.e., action, object, tools, and condition), it asks, “How do I perform this task?” The user responds in one of three ways:

- "Watch me" or "I'll show you" – At this point it is assumed the robot and the user engage in **RbD**.
- "This task is like Task A, which you already know how to do" – At this point the robot copies any sub-tasks (i.e., Primitive Tasks) from the known Task and creates a new Task.
- Starts giving the robot a list of sub-tasks (i.e., Primitive Tasks) for this Compound Task.

Table 3.2 shows how many times each of the three teaching methods is employed in the ten Lesson Plans:

		Teaching Methods		
Lesson Plan	**Compound Tasks**	**RbD**	**Is Like**	**Nested Sub-Tasks**
1	1	3	2	1
2	2	4	5	1
3	3	2	2	1
4	4, 5, & 6	6	5	1
5	7	17	8	2
6	8	19	8	2
7	9	16	10	2
8	10	18	5	2
9	11	8	4	3
10	12	7	1	1

Table 3.2: Teaching Methods Used in Lesson Plans.

The words used to teach each task were transcribed directly from the Web sites, usually without any tools or conditions. The tools and conditions were provided in gestures or utterances in response to the robot's clarifying questions, or assumed by the robot from the Real World Context or Dialog History.

The teaching words included both simple sentences (e.g., "vacuum the floor") and compound sentences with two or more tasks (e.g., "wipe down the sink after loading the dishwasher", "take down the draperies, curtains, and blinds"), which made the robot's parser more sophisticated, but made the dialog more natural. Table 3.3 shows the number of simple and compound sentences used in each Lesson Plan:

		Teaching Methods			Teaching Sentences	
Lesson Plan	Compound Tasks	RbD	Is Like	Nested Sub-Tasks	Simple	Compound
1	1	3	2	1	5	1
2	2	4	5	1	4	3
3	3	2	2	1	3	1
4	4, 5, & 6	6	5	1	6	3
5	7	17	8	2	15	5
6	8	19	8	2	12	8
7	9	16	10	2	9	9
8	10	18	5	2	10	7
9	11	8	4	3	6	4
10	12	7	1	1	5	1

Table 3.3: Simple and Compound Sentences Used in Lesson Plans.

All ten Lesson Plans are included in the Appendix.

3.7 Procedure

The experiments were performed as follows for each of the seven scenarios:

1. Set the grounding mode to Optimistic.
2. Load the Real World Context with tasks from Lesson Plans 2 through 10, which are not also in Lesson Plan 1.
3. Teach the robot Lesson Plan 1 (the teaching methodology is explained later).
4. Set the grounding mode to Cautious.

5. Reload the Real World Context with tasks from Lesson Plans 2 through 10, which are not also in Lesson Plan 1. (This ensures the robot does not retain in any Real World Context learned in Step 3).
6. Teach the robot Lesson Plan 1.
7. Set the grounding mode to Pessimistic.
8. Reload the Real World Context with tasks from Lesson Plans 2 through 10, which are not also in Lesson Plan 1. (This ensures the robot does not retain in any Real World Context learned in Step 3 or 6).
9. Teach the robot Lesson Plan 1.
10. Repeat Steps 1 through 9 for all other 9 Lesson Plans, loading the Real World Context appropriately (e.g., load the Real World Context with tasks from Lesson Plans 1 and 3 through 10, which are not also in Lesson Plan 2, and teach Lesson Plan 2).

The Real World Context is loaded using ten-fold cross-validation. Kohavi reviewed accuracy estimation methods and compared the two most common methods: cross-validation and bootstrap. His results show that for selecting a good classifier, ten-fold cross-validation may be better than the more expensive leave-one-out cross-validation [47].

The exact method of teaching the robot a task is difficult to formalize because of all the variables involved. However to ensure some consistency in the experiments, the following methodology was used for teaching each Lesson Plan:

- Use the exact wording in the “Steps” section of the Lesson Plan the first time through (Figure 4.4). For example in Lesson Plan 1, say “Wipe down the sink after loading the dishwasher” to teach the first two tasks.
- After the first try, then use whatever works. For example for the first task of Lesson Plan 1, use “Wipe down the sink with a sponge daily”.
- It is allowed to answer a slot-filling question, with just the value for the slot. For example in Lesson Plan 1, “a sponge” is an acceptable response to the question, “What tools do you want me to use to wipe down the sink?”
- Try to teach the robot a task 3 times, then give up by saying, “forget it” or “quit”.
- Use the exact wording in the “How To” section of the Lesson Plan, if it is specified; otherwise, “I’ll show you” can be used (Figure 4.5).
- If the user is unable to teach the robot one of the Primitive Tasks in a Compound Task, then go ahead and teach it the rest of the Primitive Tasks.
- When Grounding is Optimistic, answer “Are there more tasks ...” with the next task, if there are more instead of “yes”. This will reduce the number of Clarifying Questions. With Cautious and Pessimistic Grounding, the robot will verify the next task is part of the Compound Task, if the user does not answer “yes”, so the number of Clarifying Questions is the same whether the user answers with “yes” or the next Task.

Chapter 4 System Implementation

The hypothesis was tested by simulating a domestic robot that can be taught to clean a house using a multi-modal interface. Figure 3.1 is a block diagram of the system, which simulates the robot. The system was implemented by integrating software from other researchers, commercial software, and software developed as part of this dissertation. This chapter describes each of the blocks of the system in detail.

4.1 Speech Recognition

This component translates the user's utterances into a parsed list of words and passes it to the Semantic Integration component to be combined with other modalities and context. It is composed of two parts: a standard Automatic Speech Recognizer (ASR) and a parser.

4.1.1 Speech Recognition – ASR

These ASRs were considered and studied:

- IBM ViaVoice [101] – An internet search revealed that IBM no longer supports JSAPI (Java Speech API) for ViaVoice. Because the rest of the code is written in Java, the lack of JSAPI support would have made the development effort more difficult (and costly).
- Sphinx 4.0 from Carnegie-Mellon University (CMU) [85] – A great deal of effort was expended trying to integrate Sphinx 4.0 into the system because it was free, supported by a large on-line community, and was written in Java. Sphinx 4.0 is a highly modular and tunable system. Working with Nickolay V. Shmyrev, a

SourceForge [100] developer, we tried many configurations and improved the word error rate (WER) on a small subset of our corpus to 16%. According to their Web site, SourceForge is the world's largest Open Source software development Web site. SourceForge provides free hosting to Open Source software development projects, including Sphinx 4.0, with a centralized resource for managing projects, issues, communications, and code. However, when Sphinx 4.0 was integrated into the rest of the software and tested, the performance was unacceptable. The primary reason for not working was that we used an acoustic model built from the words in the Wall Street Journal. Although applicable to a wide variety of applications, the Wall Street Journal had few, if any, occurrences of the typical phrases in our corpus. CMU has another tool, SphinxTrain, for building new acoustic models from scratch. The problem is that SphinxTrain only runs under UNIX and it would be a non-trivial effort to move it to Windows.

- Dragon NaturallySpeaking from Nuance [94] – We attempted to install Dragon NaturallySpeaking, but it requires a minimum of 512MB of memory and our system only had 368MB and could not be upgraded.

The ASR finally selected for the system was e-Speaking, which is a shareware product published by e-Speaking.com [99]. e-Speaking was simple to install and was trained using the speech recognition tools that come with Windows XP [98] and Microsoft SAPI 5.1 [97].

The integration into the rest of the system was not as elegant as with Sphinx 4.0, but it provides keyboard-free input of speech into the system, which was what was

desired. e-Speaking is a stand-alone software package that lets the user speak words into any standard Windows application. A simple Java Windows program that is provided with the Flippo Framework allowed us to use e-Speaking to speak words into the system. The user speaks to the robot as follows:

1. The user positions the cursor in the e-Speaking application window and says (through the computer microphone), “start dictation” (Figure 4.1).
2. e-Speaking says (through the computer speakers), “You may begin dictating.”
3. The user positions the cursor in the Java “Speech input” window and says something to the robot like, “clean the kitchen daily” (Figure 4.2).
4. To send the speech to the robot, the user says, “return”.
5. The Java application sends the speech to the parser, and highlights the user’s words in blue to indicate they have been sent to the parser (Figure 4.3).
6. The user continues speaking to the robot by repeating Steps 3-5.

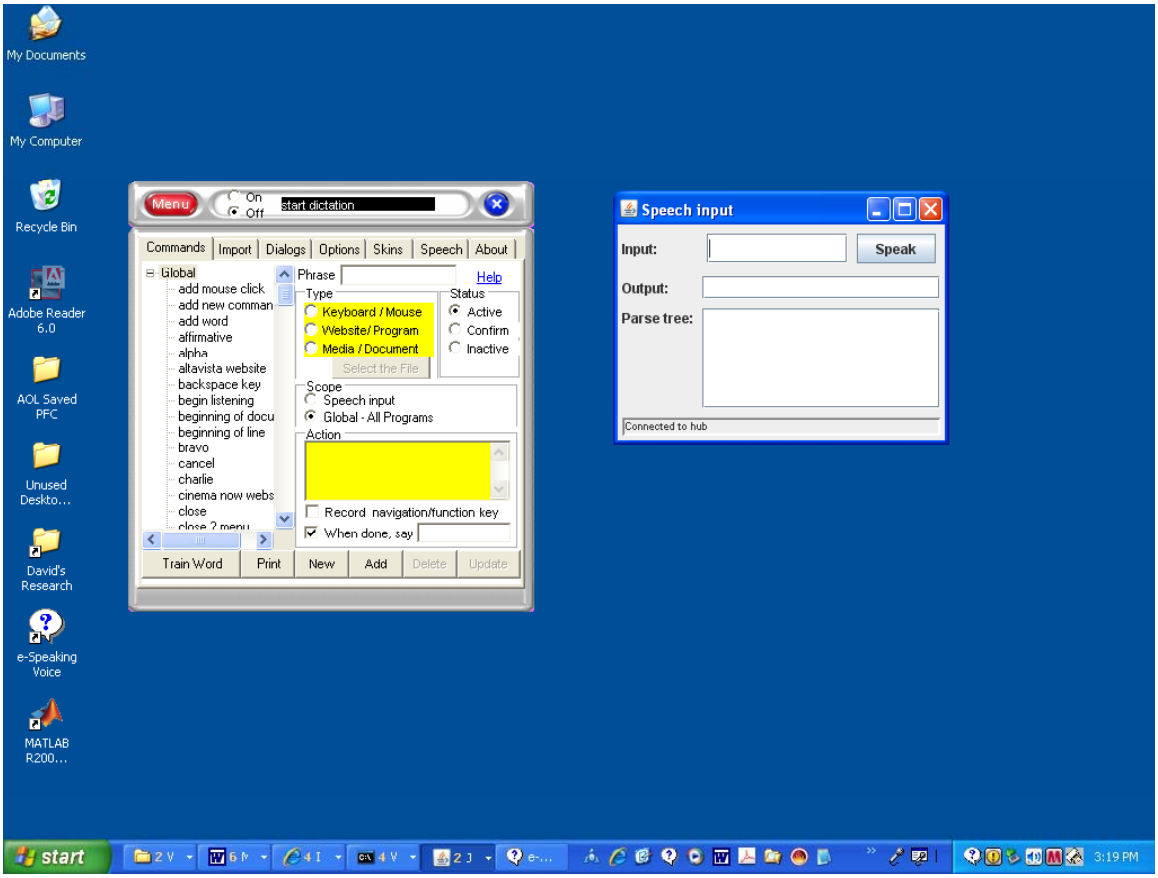

Figure 4.1: User Says “start dictation” to e-Speaking.

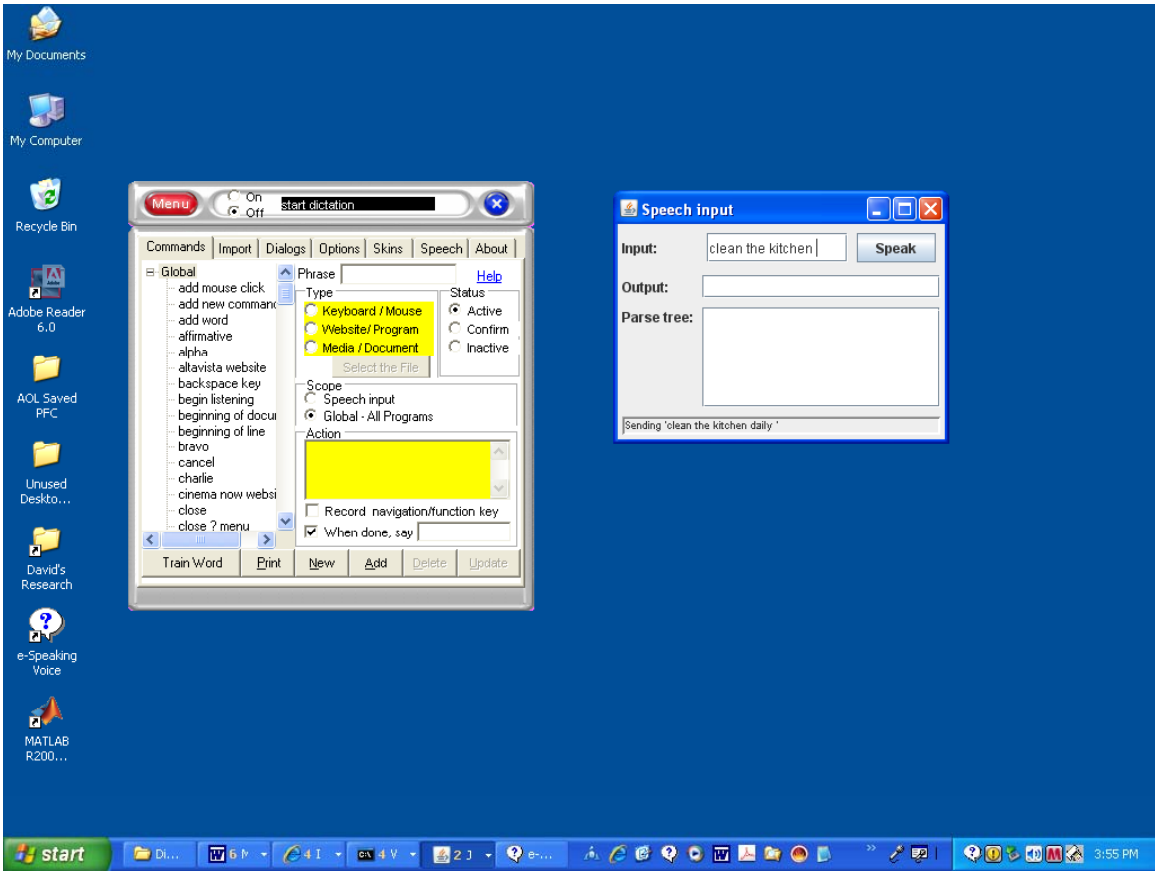

Figure 4.2: User Says “clean the kitchen” to Speech Input Window.

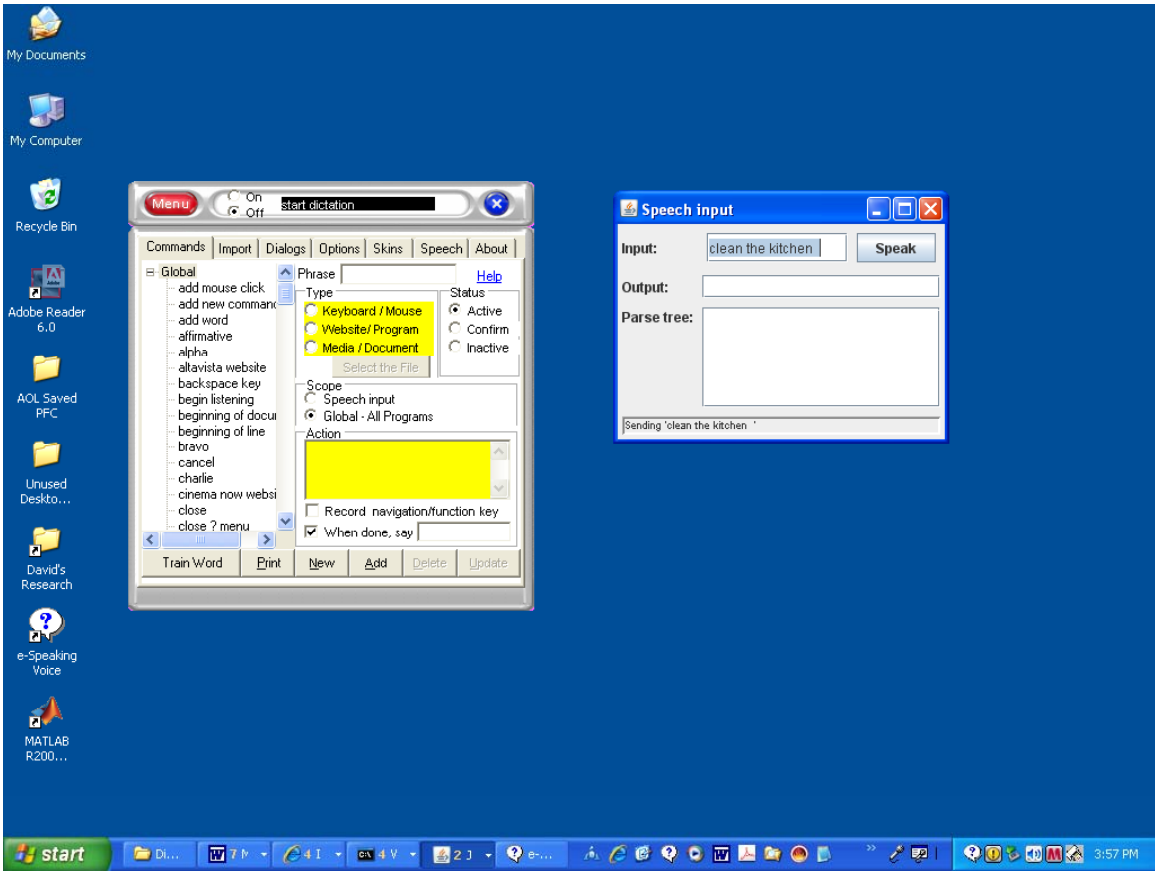

Figure 4.3: System Acknowledges "clean the kitchen" Sent to Parser.

The Word Error Rate (WER) for e-Speaking was measured experimentally as 6.7% using the same subset of the corpus as was used to measure it for Sphinx 4.0.

4.1.2 Speech Recognition – Parser

The system parses the speech from the ASR using the Colorado University Phoenix parser [95]:

> "Phoenix parses each input utterance into a sequence of one or more semantic frames. The developer must define a set of frames and provide grammar rules that specify the word strings that can fill each slot in a frame".

We designed eight frames that correspond to the dialog between the user and the robot:

1. Single-task – The primary unit of the dialog, which describes an action (e.g., clean, wash, dust) taken on an object (e.g., sink, dish, bed, couch) using none, one, or more tools (e.g., wash rag, vacuum cleaner) under a specified condition (e.g., daily, in the spring).
2. Connector – Connects two simple sentences in single-task frames together to form a compound sentence with words like after, next, or then (e.g., wipe down the sink after loading the dishwasher).
3. Learning – Tells the robot that two tasks are similar with words like, “is like” or “is similar to” (e.g., wipe down the stove top is similar to wipe down the sink).
4. RbD – Robot Programming by Demonstration (RbD), which tells the robot that the user is going to show it how to perform a task (e.g., I’ll show you, watch me)
5. Answer – Reply of yes, no, or “I don't know” to a question posed by the robot.
6. Quit – Tells the robot that the user is going to start over (e.g., quit, forget it).
7. RequestRepeat – Tells the robot to repeat what it just said (e.g., “what?”, “huh?”, “I don't understand”).
8. Greet – A greeting (e.g., welcome, hello, how are you?).

The frames, their slots, and an example are shown in Table 4.1 through Table 4.8.

Frame	single-task	
Slots	[action]	An action (e.g., clean, wash, dust) ...
	[object]	... taken on an object (e.g., sink, dish, bed, couch) ...
	[tools]	... using none, one, or more tools (e.g., wash rag, vacuum cleaner)
	[condition]	... under a specified condition (e.g., daily, in the spring).
	[anaph]	This is a special slot to hold anaphora (e.g., it, that), which might refer to any of the other slots. The anaphora is never resolved explicitly, but its presence causes Semantic Integration to fill all slots, so a simple command like "do that" will cause Semantic Integration to fill all the other slots with information from all the modalities and contexts to create a valid single-task frame.
Example		
Utterance	clean the kitchen daily with a sponge	
Phoenix Parse	Single-task: [clean] [kitchen] [sponge] [daily]	

Table 4.1: Single-task Frame.

Frame	Connector	
Slots	[connector]	Connects two simple sentences in single-task frames together to form a compound sentence with words like after, next, or then.
Example		
Utterance	wipe down the sink after loading the dishwasher	
Phoenix Parse	Single-task: [wipe down] [sink] [] [] Connector: [after] Single-task: [loading] [dishwasher] [] []	

Table 4.2: Connector Frame.

Frame	Learning	
Slots	[operator]	Tells the robot that two tasks are similar with words like, “is like” or “is similar to”.
Example		
Utterance	wipe down the stove top is similar to wipe down the sink	
Phoenix Parse	Single-task: [wipe down] [stove top] [] [] Learning: [is similar to] Single-task: [wipe down] [sink] [] []	

Table 4.3: Learning Frame.

Frame	RbD	
Slots	[show]	Robot Programming by Demonstration (RbD), which tells the robot that the user is going to show it how to perform a task.
Example		
Utterance	watch me	
Phoenix Parse	RbD: [watch me]	

Table 4.4: RbD Frame.

Frame	Answer	
Slots	[answer]	Reply of yes, no, or I don't know to a question posed by the robot.
Example		
Utterance	nope	
Phoenix Parse	Answer: [no]	

Table 4.5: Answer Frame.

Frame	Quit	
Slots	[quit]	Tells the robot that the user is going to start over (e.g., quit, forget it).
Example		
Utterance	forget it	
Phoenix Parse	Quit: [forget it]	

Table 4.6: Quit Frame.

Frame	RequestRepeat	
Slots	[requestRepeat]	Tells the robot to repeat what it just said (e.g., what?, huh?, I don't understand).
Example		
Utterance	huh	
Phoenix Parse	RequestRepeat: [huh]	

Table 4.7: RequestRepeat Frame.

Frame	Greet	
Slots	[greeting]	A greeting (e.g., welcome, hello, how are you?).
Example		
Utterance	hello	
Phoenix Parse	Greet: [hello]	

Table 4.8: Greet Frame.

The grammar rules that specify the word strings that can fill each slot in a frame are divided into two sets, namely, base grammar rules and task grammar rules.

The base grammar rules define the word strings that can fill each slot of all the frames, except the single-task frame. The base grammar rules are defined in the Appendix.

The task grammar rules are automatically extracted from the XML-based Lesson Plan files by a Java tool we developed called the GrammarParser. The Lesson Plan files are used to guide the user in teaching the robot the tasks used in the experiments to evaluate the research hypothesis. They contain the following information:

- Task: This is the main task being taught (e.g., clean the kitchen daily).
- Steps: These are the sub-tasks of the main task (e.g., wipe down the sink after loading the dishwasher).
- Answers: These are the answers to the questions that the robot might ask about the missing slots in the Steps (e.g., "a sponge" is the answer to the question, "What tools do you want me to use to wipe down the sink?").
- How To: These are the Teaching Methods the user uses to teach the robot how to perform a specific task or sub-task (e.g., RbD, Is Like, or Nested Sub-Tasks).
- The task is in the following format that we developed:

- action,object,[tools],condition,[sub-tasks];

The “Task”, “Steps”, “Answers”, and “How To” information is displayed in the Lesson Plan field of the HRI Environment Simulator as shown in Figure 4.4 and Figure 4.5. (The other details of the HRI Environment Simulator are provided later in this chapter).

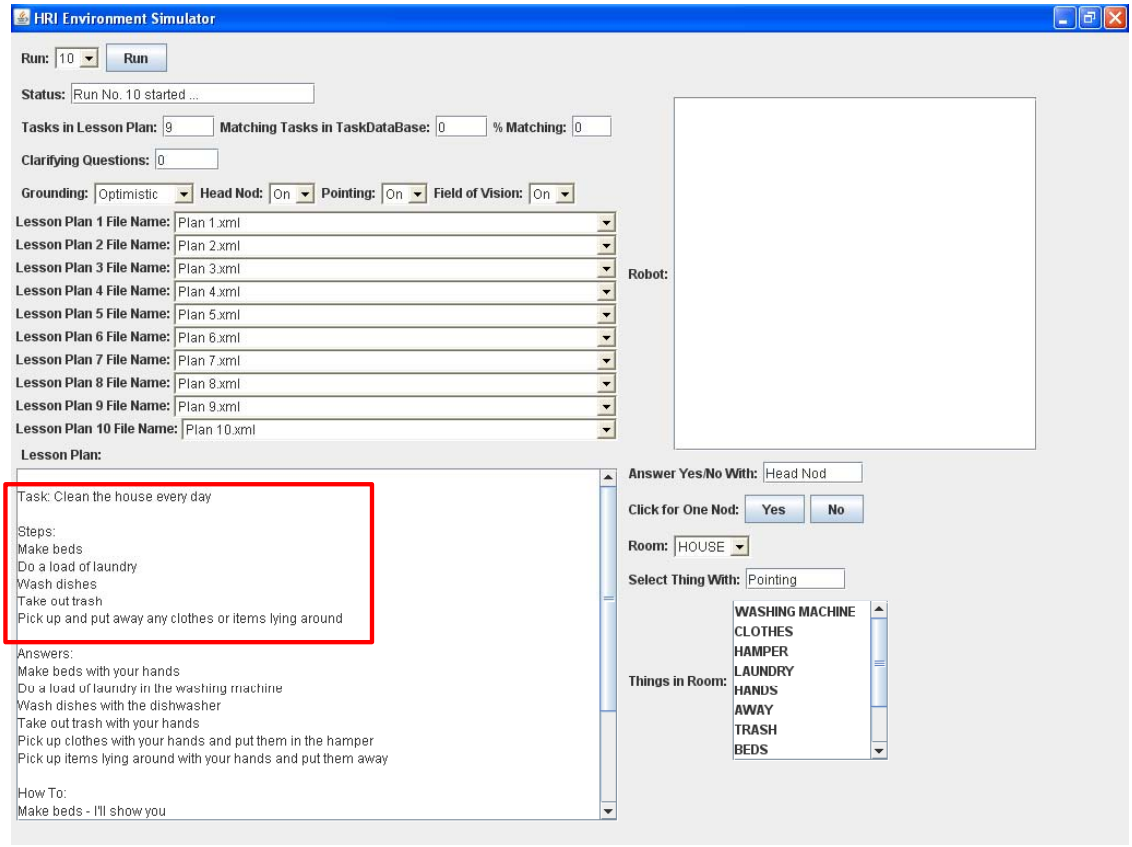

Figure 4.4: Task and Steps Information in HRI Environment Simulator.

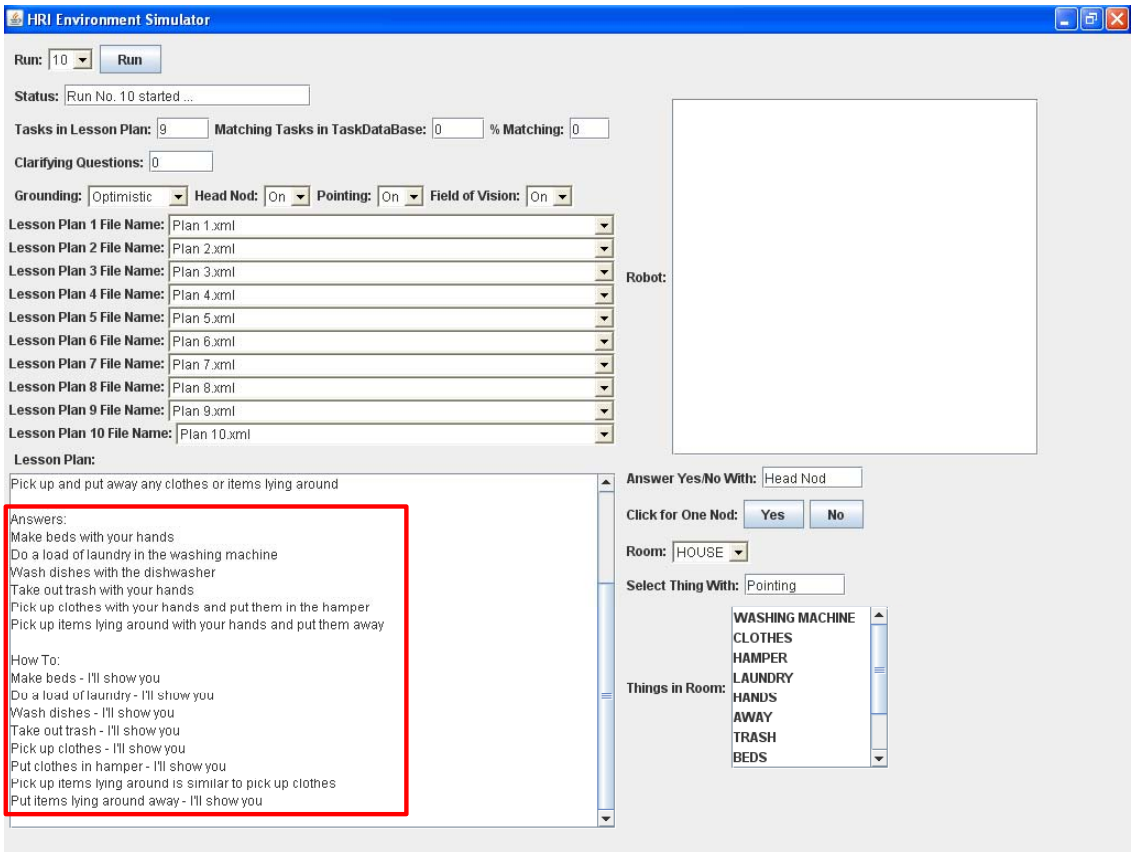

Figure 4.5: Answers and How To Information in HRI Environment Simulator.

The Appendix includes the Lesson Plans we used in the experiments and more details on the format of the Lesson Plan files. The task grammar rules are also included in the Appendix.

Because we are using a very syntactically relaxed dialog in our experiments, many word strings in the task grammar can be used to fill more than one slot. For example, "vacuum" can be either an "action" to perform, or a "tool" to use in performing a task as illustrated in these two example utterances:

- **Vacuum** the floor.
- Clean the carpets and rugs with a **vacuum** and rug shampoo machine.

The Phoenix parser is robust enough to identify both parses, and both parses are provided to the Semantic Integration module where the other modalities and contexts are used to resolve the ambiguity.

4.2 Head Nod Recognition

When people interact naturally with each other, head nods are frequently used to answer yes or no.

This component recognizes yes and no head nods and passes that information to the Semantic Integration component to be combined with other modalities.

This component (and other modality and context modules) were developed by us and added to the Flippo Framework.

The Flippo Framework defines two abstract Java classes:

- AbstractContextProvider – Collects data from one modality connected to the system.
- AbstractResolver – Provides a list of possible values for a slot, along with confidence scores (i.e., probability that the value is correct), for one modality.

Our HeadNodContextProvider simulates the Head Nodding recognizer described by Morency and Darrell [60]. In their experiments, a head nod was recognized correctly 85.3% of the time.

The user in our experiments simulates a head nod by clicking on either the Yes or No button on the HRI Environment Simulator window shown in Figure 4.6.

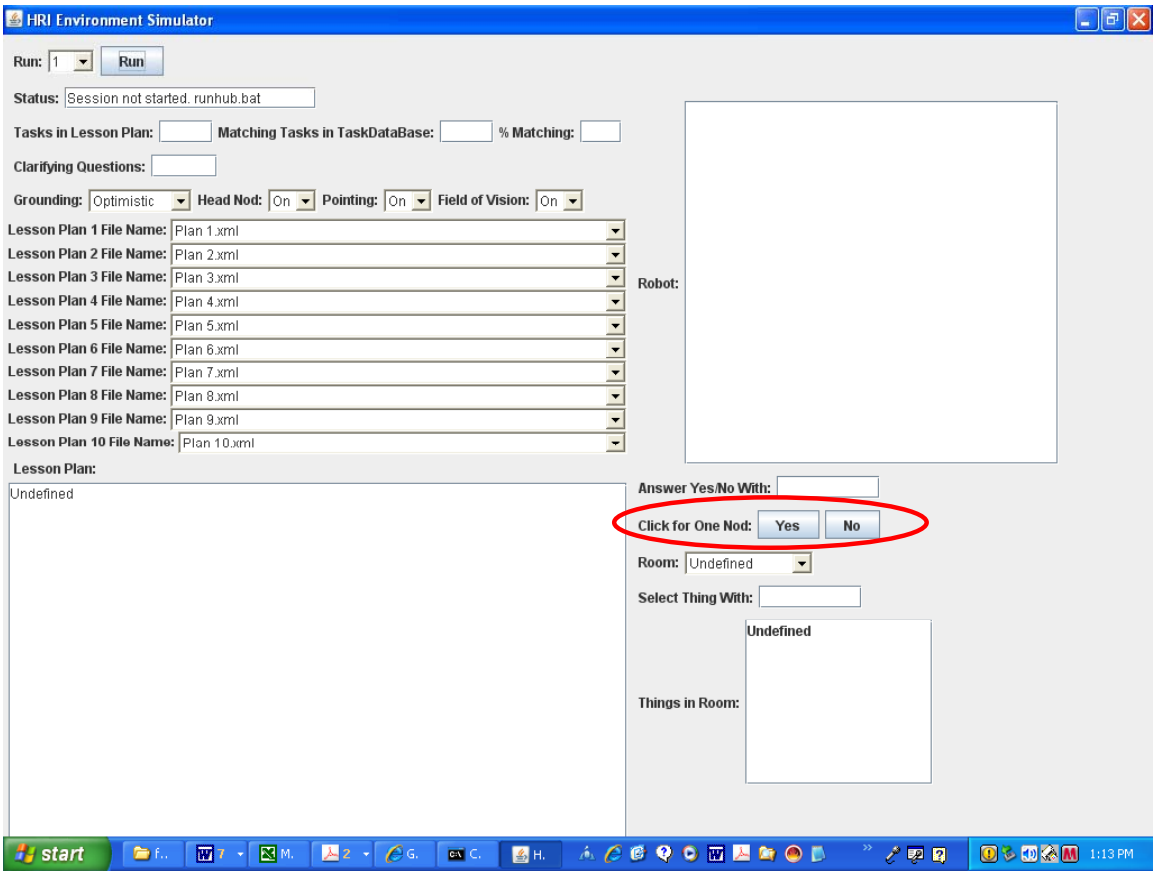

Figure 4.6: Head Nod Buttons in HRI Environment Simulator.

The HeadNodResolver provides either a "yes" or "no" value with a confidence score of 100% to Semantic Integration. Although the HeadNodContextProvider is simulating errors consistent with the Head Nodding recognizer described by Morency and Darrell [60], as far as the HeadNodResolver is concerned the value provided by the HeadNodContextProvider is 100% accurate. Therefore the confidence score is 100%. The fact that the nod might be incorrect is taken into consideration by the Semantic Integration component, which assigns a weight to the inputs it receives from the various Resolvers as part of the fusion process, which is described in more detail later.

4.3 Pointing Recognition

Pointing is also a frequently used gesture when people interact naturally with each other.

This component identifies which objects the user is pointing at and passes that information to the Semantic Integration component to be combined with other modalities and context.

Our PointingContextProvider simulates the Pointing recognizer described by Nickel and Stiefelhagen [62]. They reported a true-positive rate (number of detected gestures divided by the total number of ground truth gestures) of 78.3% and a false-positive rate (number of falsely detected frames divided by the total number of non-gesture frames) of 11.6%.

The user in our experiments simulates pointing by selecting a Thing in the Room on the HRI Environment Simulator window, as SWEEPER is shown in Figure 4.7.

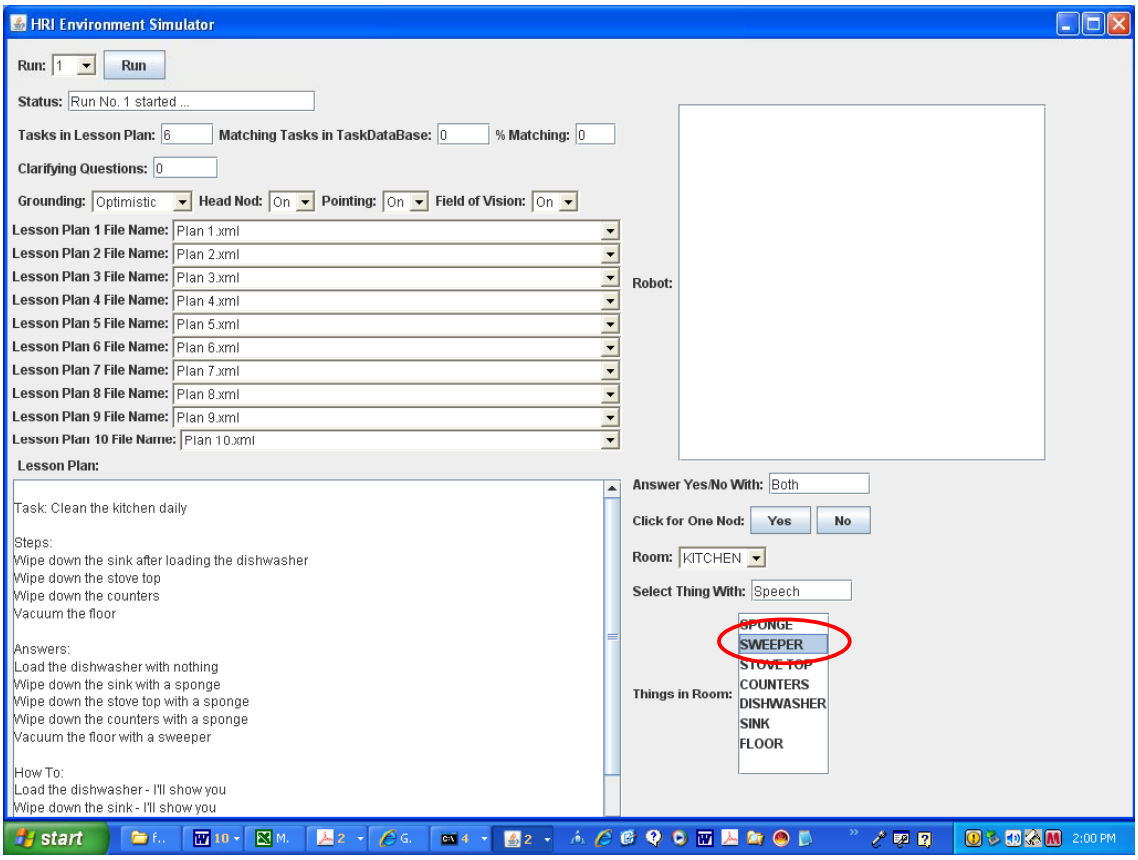

Figure 4.7: Pointing at Things In Room in HRI Environment Simulator.

When Semantic Integration calls a Resolver, it tells the Resolver which slot it is trying to fill. This is important to the PointingResolver because a Thing in the Room can be either an "object" or a "tool". After the PointingResolver gets the Thing that the user is pointing at from PointingContextProvider, it first searches the Real World Context database to see if the Thing can be an "object", if the object slot is being filled, or if the Thing can be a "tool", if the tools slot is being filled. The tools slot may contain none, one, or many "tool" strings.

The Real World Context database is a set of all the tasks the robot knows how to perform. The Real World Context database is described in more detail later.

If the Thing can fill the slot, the PointingResolver provides the Thing with a confidence score of 100% to Semantic Integration. If the Thing cannot fill the slot, the PointingResolver does not pass it on to Semantic Integration.

4.4 Field of Vision

Chai *et al.* propose an algorithm for disambiguating speech using pointing, dialog history, and all visible objects (i.e., field of vision) [21]. Field of vision is used as a final test when pointing and dialog history fail to disambiguate the user's utterance. An example of where Field of Vision is applicable to our research, is:

> The robot and the user are standing in the kitchen. The user is explaining what to clean in the kitchen and says, "Clean the table after every meal." "Table" is an ambiguous reference because a house contains several tables. However, the Field of Vision narrows the reference down to the "table in the kitchen".

This component identifies the objects in the robot's field of vision and passes that information to the Semantic Integration component to be combined with other modalities and context.

The FieldOfVisionContextProvider simulates the Field of Vision recognizer described by Jensfelt *et al.* [39]. They reported a true-positive rate (number of detected Things divided by the total number of Things in the Room) of 95.0% and a false-positive rate (number of falsely detected Things divided by the total number of Things detected) of 2.3%.

In our experiments, all the Things in the Room on the HRI Environment Simulator window simulate the Field of Vision, as SPONGE, SWEEPER, STOVE TOP, COUNTERS, DISHWASHER, SINK, and FLOOR are shown in Figure 4.8.

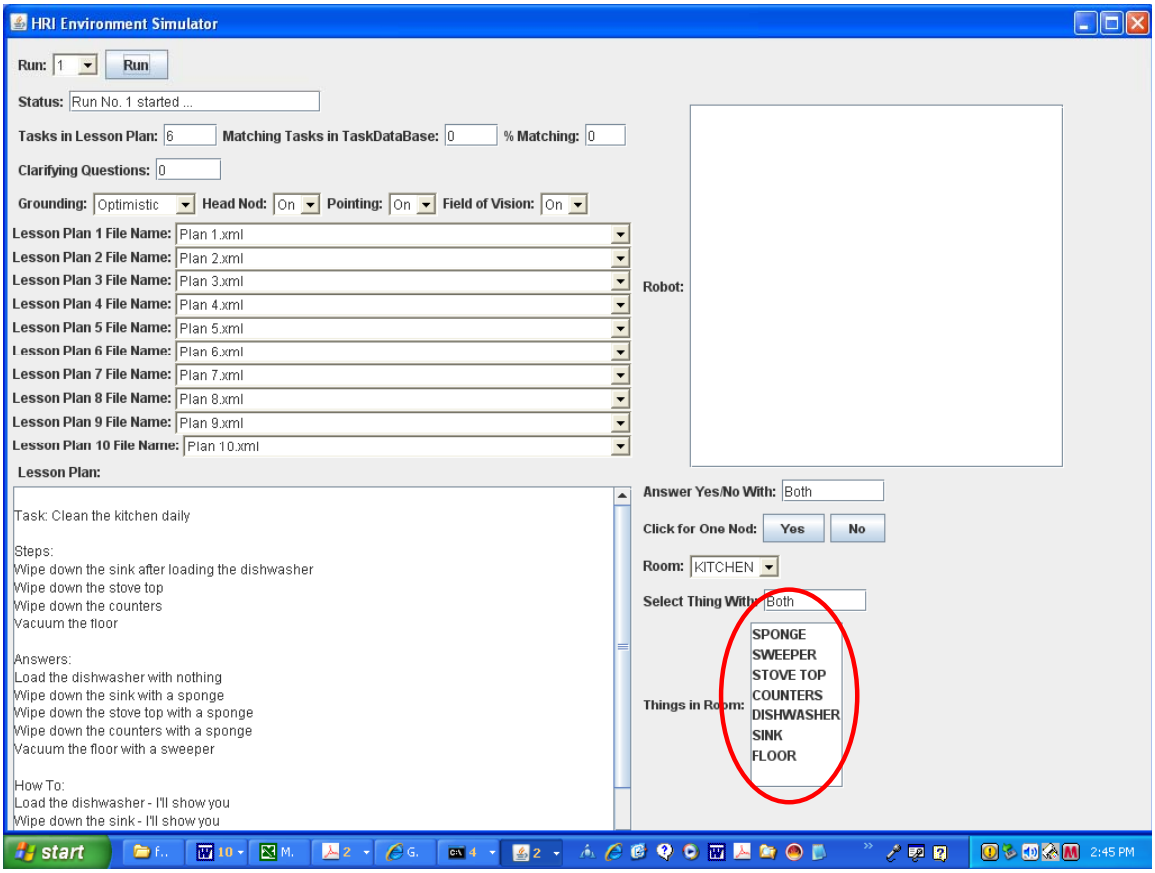

Figure 4.8: Field of Vision in HRI Environment Simulator.

Similar to the PointingResolver, after the FieldOfVisionResolver gets the Things in the field of vision from the FieldOfVisionContextProvider, it first searches the Real World Context database to see if the Things can be "objects", if the object slot is being filled, or if the Things can be "tools", if the tools slot is being filled.

After it has determined the valid Things in the field of vision, what the FieldOfVisionResolver does next depends on the slot being filled.

If the object slot is being filled, the FieldOfVisionResolver provides a list of Things each with a confidence score of 1/*n* to Semantic Integration, where *n* is the number of valid Things.

For example, if the FieldOfVisionResolver found these Things filling object slots in the Real World Context database:

1. STOVE TOP
2. COUNTERS
3. DISHWASHER
4. SINK
5. FLOOR

It would then provide this list of five Things, each with a confidence score of 20%, to Semantic Integration as possible values for the object slot.

If the tools slot is being filled, the FieldOfVisionResolver first creates sets of tools by taking permutations of the Things. This is because the tools slot may contain none, one, or many "tool" strings. Then, it provides a list of tool sets, each with a confidence score of 1/*n* to Semantic Integration, where *n* is the number of tool sets.

For example, if the FieldOfVisionResolver found these Things filling tool slots in the Real World Context database:

1. SPONGE
2. SWEEPER
3. DISHWASHER

FieldOfVision would create the following seven tool sets:

1. [SPONGE]
2. [SWEEPER]
3. [DISHWASHER]
4. [SPONGE, SWEEPER]
5. [SPONGE, DISHWASHER]
6. [SWEEPER, DISHWASHER]
7. [SPONGE, SWEEPER, DISHWASHER]

It would then provide this list of seven tool sets, each with a confidence score of 14.3%, to Semantic Integration as possible values for the tools slot.

Obviously, a large number of tools in the field of vision could generate a rather large number of tool sets. To prevent this, FieldOfVisionResolver trims the list of tool sets in two ways:

1. The number of tools in a set cannot exceed a maximum number of tools per set. For our experiments, this value was set to three under the assumption that most household tasks are performed with three or fewer tools.
2. The number of tool sets cannot exceed a maximum number of tool sets. For our experiments, this value was set to 25 under the assumption that with a larger number of tool sets, the confidence level (e.g., 4%) would be so low that Semantic Integration would ignore them as possibilities to fill the tools slot.

However, at a minimum, the FieldOfVisionResolver will always provide a list of tool sets containing each of the tools in the field of vision in a separate tool set, even if the number of sets exceeds the maximum number of tool sets allowed. For

example, if there are *n* tools in the field of vision, FieldOfVisionResolver will always provide at least *n* sets of tools.

Also, the FieldOfVisionResolver never provides an incomplete list of permutations for any number *m*, where *m* is the number of tools in a set. For example, if there are six tools in the field of vision:

- The number of tool sets with one tool in the set is *nt*/1!, where *nt* is the number of tools, or 6.
- The number of tool sets with two tools in the set is *nt*(*nt*-1)/2!, or 15.
- The number of tool sets with three tools in the set is *nt*(*nt*-1)(*nt*-2)/3!, or 20.
- 6 + 15 + 20 > 25
- So, FieldOfVisionResolver would trim the list of tool sets to the 21 sets containing one or two tools.

Because DISHWASHER appears both as an object and a tool in the Real World Context database example above, the FieldOfVisionResolver (and the PointingResolver) will provide it to Semantic Integration to fill both the object and tools slot. Semantic Integration will try to resolve this ambiguity as described later.

4.5 Context

Humans also use contextual information to resolve missing or ambiguous references in a conversation. Our robot simulation uses Real World Context and Dialog History.

4.5.1 Real World Context

Real World Context is the information that a human knows through experience. For example, if Mary tells John to clean the sink, John may know that sinks are usually cleaned with a sponge. So, Mary does not have to tell him to clean the sink with a sponge, and John does not have to ask Mary what to clean the sink with.

We provide the robot with Real World Context by pre-loading it with a set of already defined tasks.

This component searches through the pre-loaded tasks looking for values to fill empty slots and passes that information to the Semantic Integration component to be combined with other modalities and context.

The Lesson Plans are used to pre-load the Real World Context for the experiments. For each Lesson Plan, the tasks in the other Lesson Plans are pre-loaded into the Real World Context. For example, when teaching the robot the tasks in Lesson Plan 1, the tasks from Lesson Plans 2 through 10 are pre-loaded into the Real World Context.

The only exception is that no tasks, which are in the Lesson Plan being taught, are pre-loaded into the Real World Context. As an example, the task “Take down the draperies” occurs in both Lesson Plans 5 and 6. So, when Lesson Plan 5 is being taught, “Take down the draperies” from Lesson Plan 6 will not be included in the Real World Context. This is done to ensure that the robot has to learn every task in a Lesson Plan, and cannot fill unspoken slots of the task (e.g., “tools” or “condition”) by merely looking in the Real World Context.

The file names for each Lesson Plan are identified in the HRI Environment Simulator window as shown in Figure 4.9.

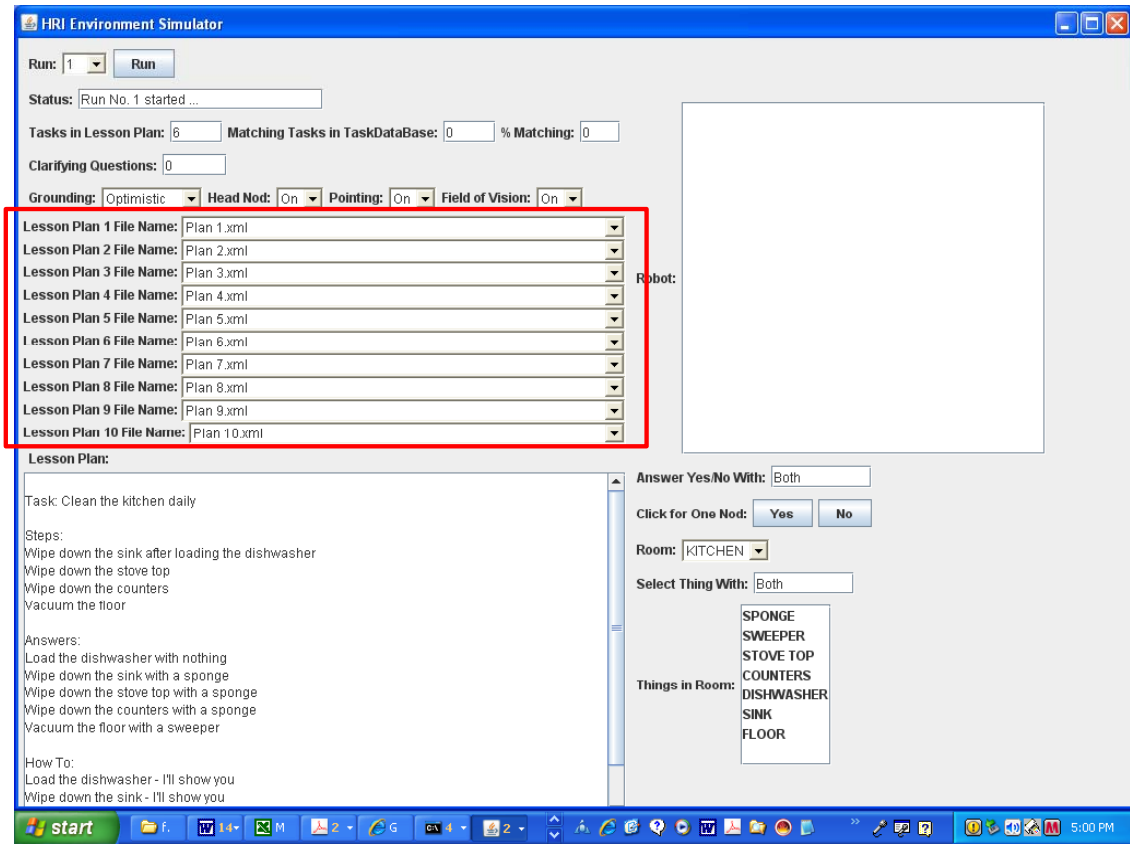

Figure 4.9: Lesson Plan File Names in HRI Environment Simulator.

The file names are selected with drop-down menus in the HRI Environment Simulator window as shown for Lesson Plan 5 in Figure 4.10.

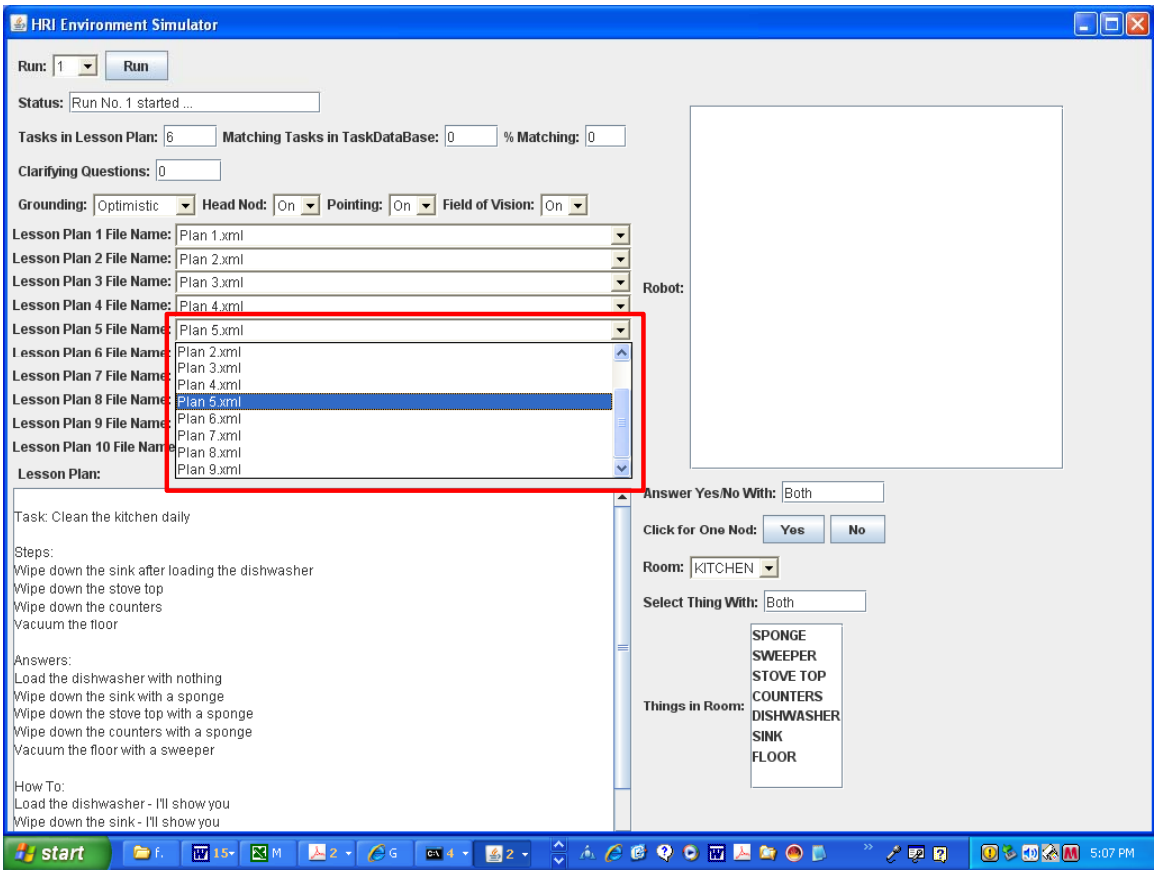

Figure 4.10: Selecting Lesson Plan File Name.

The Lesson Plan being taught is identified in the Run box of the HRI Environment Simulator window as displayed in Figure 4.11.

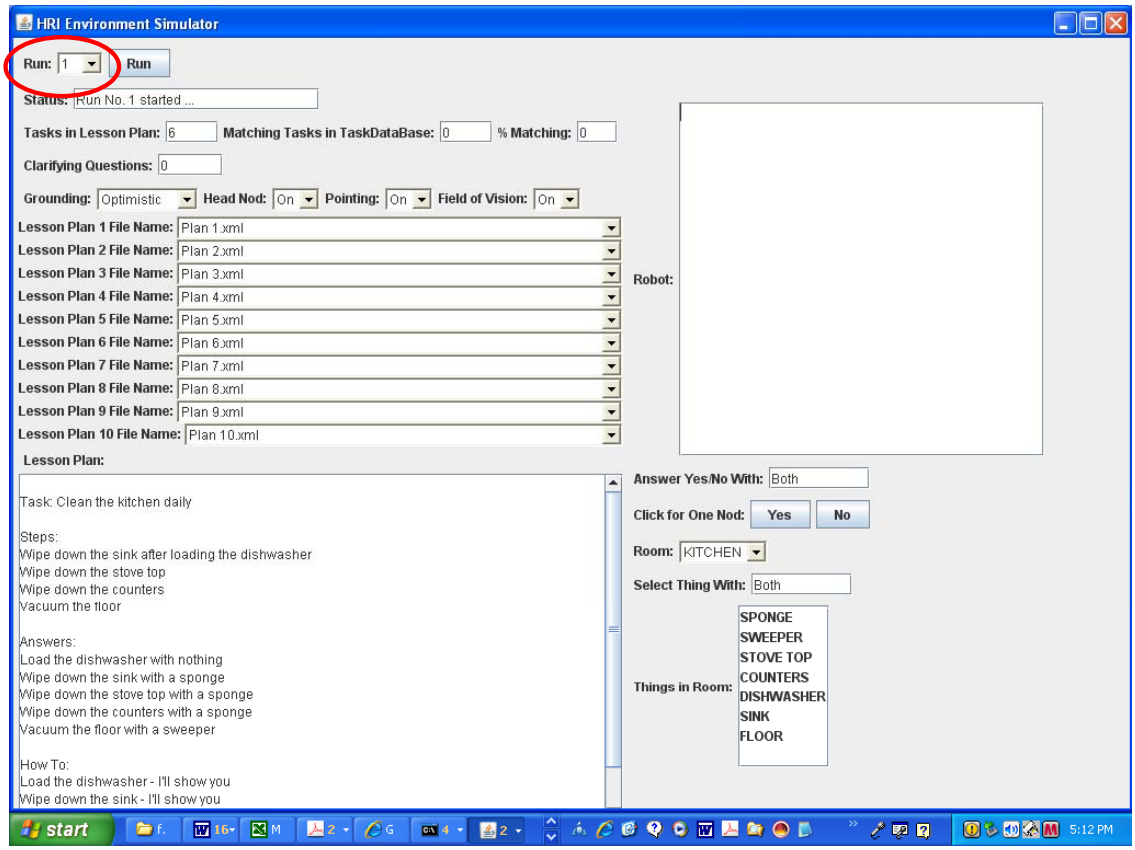

Figure 4.11: Lesson Plan Being Taught in HRI Environment Simulator.

The Lesson Plan to teach is selected with the drop-down menu in the HRI Environment Simulator window as illustrated in Figure 4.12.

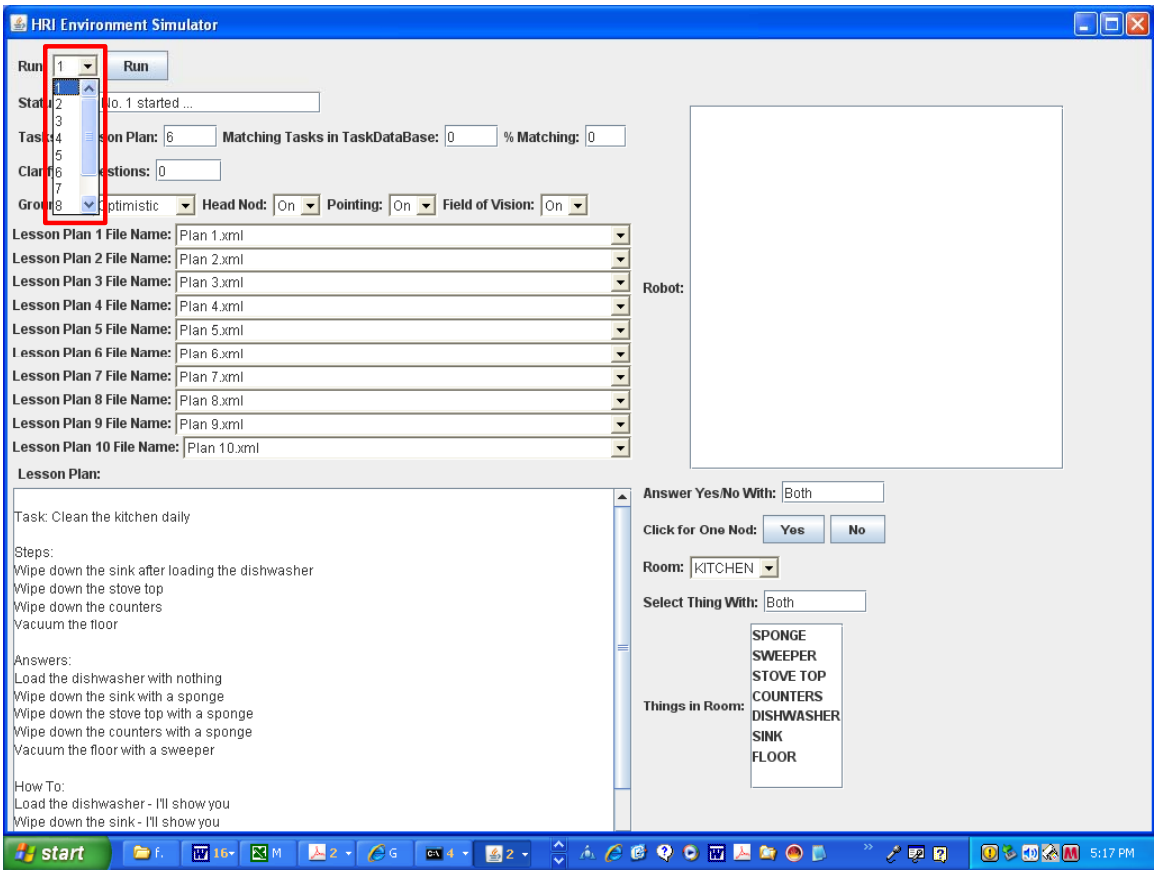

Figure 4.12: Selecting Lesson Plan To Teach in HRI Environment Simulator.

When the user clicks on the Run button on the HRI Environment Simulator window in Figure 4.13, the system pre-loads the Real World Context with the tasks from the other Lesson Plans, excluding any that are also in the Lesson Plan being taught.

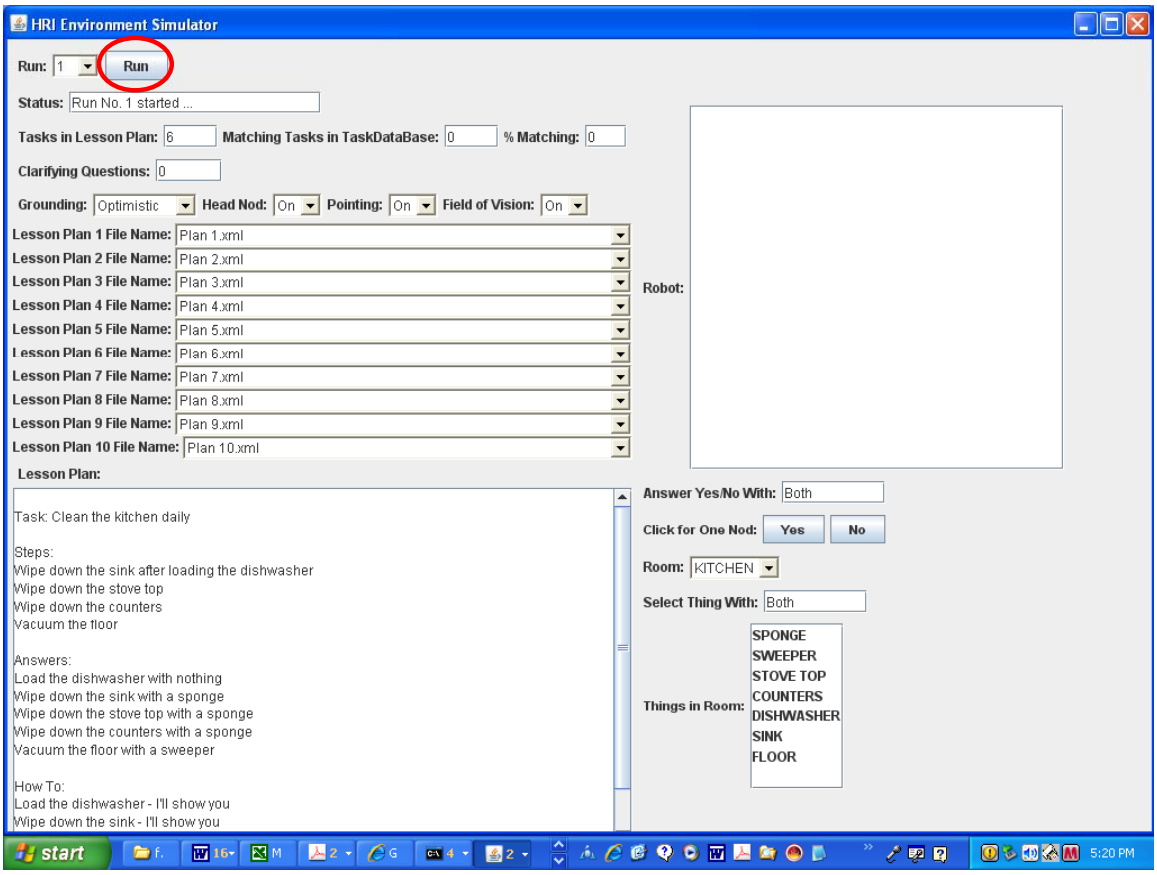

Figure 4.13: Pre-Load Real World Context in HRI Environment Simulator.

As with the other components that feed into Semantic Integration, we implemented this component as a Flippo Framework AbstractResolver. The RealWorldResolver searches the Real World Context for tasks that match the partially filled single-task frame and gets a list of values from the matching tasks for the specified slot. Then, it calculates a confidence score for each unique value in the list as m/n, where m is the number of occurrences of the unique value and n is the number of all values found.

For example, if the partially filled single-task frame is:

action: clean

object: kitchen

tools: not filled

condition: not filled

RealWorldResolver will search the Real World Context for all the tasks with action = clean and object = kitchen. Since the tools slot and condition slot are not filled, RealWorldResolver will treat these as wildcards in its search, i.e., matching with anything.

For the example, Semantic Integration is trying to fill the condition slot and it finds the following values for the condition slot in its search:

27 occurrences of DAILY

48 occurrences of IN THE SPRING

9 occurrences of EVERY DAY

2 occurrences of AFTER EVERY USE

The confidence scores for each unique value would be:

DAILY = 27/(27 + 48 + 9 +2) = 31%

IN THE SPRING = 48/(27 + 48 + 9 +2) = 56%

EVERY DAY = 9/(27 + 48 + 9 +2) = 10%

AFTER EVERY USE = 2/(27 + 48 + 9 +2) = 3%

4.5.2 Dialog History

During a dialog, humans tend to only specify the new or changed aspects of the discussion without repeating what has been mentioned earlier in the conversation. In

this way, humans use the Dialog History to resolve missing or ambiguous references in a conversation. For example, if Mary is explaining how to "clean the kitchen **daily**" to John, she does not have to add "**daily**" to the end of each step. It is sufficient to tell John to "wipe the counters with a sponge", and not say, "wipe the counters with a sponge **daily**". Dialog history is also used to resolve anaphora (e.g., it, that, those). The Flippo Framework provides a DialogHistoryResolver, which was used in this work.

When Semantic Integration has resolved the slots of a frame as much as it can without further dialog (e.g., asking the user to clarify something), its slots are added to the Dialog History. There is currently room for seven slots. Flippo states the reason for seven was that since the human mind can only maintain approximately seven entities at a time, we do not need to store more than that number in the Dialog History [34]. Although modifying a constant in the Java code easily changes the number, we left it at seven due to the lack of a better number.

During fusion, the DialogHistoryResolver provides Semantic Integration with the first slot value in the dialog history for a specified slot with a confidence score of 100%. An example dialog is as follows:

User: Clean the kitchen daily.

Robot: How do I clean the kitchen daily?

User: First, wipe down the sink.

Robot: What do I wipe down the sink with?

User: A sponge.

After the first utterance from the user, the Dialog History would be:

action: clean (from 1st utterance)

object: kitchen (from 1st utterance)

condition: daily (from 1st utterance)

For the second user utterance, the Speech Recognition component would provide the following to Semantic Integration:

action: wipe down

object: sink

tools: not filled

condition: not filled

During fusion, the DialogHistoryResolver would provide Semantic Integration with a condition slot value of "daily". After fusion of the second utterance, the Dialog History would be:

action: wipe down (from 2nd utterance)

object: sink (from 2nd utterance)

condition: daily (from 2nd utterance)

action: clean (from 1st utterance)

object: kitchen (from 1st utterance)

condition: daily (from 1st utterance)

For the third user utterance, Speech Recognition would provide the following:

tools: sponge

During fusion, the DialogHistoryResolver would provide Semantic Integration with the following slot values:

action: wipe down (slot 1 in the Dialog History)

object: sink (slot 2 in the Dialog History)

condition: daily (slot 3 in the Dialog History

After fusion of the third utterance, the Dialog History would become:

action: wipe down (from 3rd utterance)

object: sink (from 3rd utterance)

tools: sponge (from 3rd utterance)

condition: daily (from 3rd utterance)

action: wipe down (from 2nd utterance)

object: sink (from 2nd utterance)

condition: daily (from 2nd utterance)

4.6 Semantic Integration

The Semantic Integration component performs two functions:

1. Fusion – Combining input from multiple modalities (e.g., Speech, Pointing, Field of Vision) and contexts (e.g., Real World Context, Dialog History) into a frame.

2. Integration – Combining the fused frames into a meaningful Fused Utterance (e.g., task, answer to a question, explanation of how to perform a task)

4.6.1 Fusion

The fusion software for Semantic Integration was provided as part of the Flippo Framework.

A flexible XML configuration file controlled the fusion software. The configuration file defined the following for each frame and slot received from the Phoenix parser:

- A list of Resolvers used to resolve the slot.
- The weight that is applied to the confidence scores provided by the Resolvers.

For example, the configuration file used for the experiments with all the modalities and contexts active (Speech + All) contained the following configuration in XML format:

```
frame: single-task
        slot: action
                resolver: SpeechResolver (weight = 1.0)
                resolver: RealWorldResolver (weight = 0.2)
                resolver: DialogHistoryResolver (weight = 0.6)
        slot: object
                resolver: SpeechResolver (weight = 0.92)
                resolver: PointingResolver (weight = -0.06)
                resolver: FieldOfVisionResolver (weight = -0.63)
                resolver: RealWorldResolver (weight = 0.12)
                resolver: DialogHistoryResolver (weight = 0.84)
        slot: tools
                resolver: SpeechResolver (weight = 0.55)
                resolver: PointingResolver (weight = 0.52)
                resolver: FieldOfVisionResolver (weight = -0.92)
                resolver: RealWorldResolver (weight = 1.08)
                resolver: DialogHistoryResolver (weight = 0.54)
```

slot: condition
resolver: SpeechResolver (weight = 0.12)
resolver: RealWorldResolver (weight = 0.84)
resolver: DialogHistoryResolver (weight = 0.12)

frame: connector
slot: connector
resolver: SpeechResolver (weight = 1.6)

frame: learning
slot: learning
resolver: SpeechResolver (weight = 1.6)

frame: rbd
slot: show
resolver: SpeechResolver (weight = 1.6)

frame: answer
slot: answer
resolver: SpeechResolver (weight = 0.933)
resolver: HeadNodResolver (weight = 0.853)

frame: quit
slot: quit
resolver: SpeechResolver (weight = 1.6)

frame: requestRepeat
slot: requestRepeat
resolver: SpeechResolver (weight = 1.6)

frame: greet
slot: greet
resolver: SpeechResolver (weight = 1.6)

Except for the single-task and answer frames, fusion is trivial because there is only one modality – Speech.

4.6.2 Single-Task Fusion

For the single-task frame, Semantic Integration fuses input from multiple modalities (e.g., Speech, Pointing, Field of Vision) and contexts (e.g., Real World

Context, Dialog History) into a task for the robot to perform (e.g., sweep the floor). Each task is composed of four parts:

- action – The action to be performed (e.g., sweep)
- object – The object on which the action is to be performed (e.g., floor)
- tools – The tools the robot is to use to perform the action on the object (e.g., broom)
- condition – How often the robot is to perform the task (e.g., every day)

Specifically, the following modality and context input is provided to Semantic Integration for fusion:

- Speech – E.g., "sweep the floor"
- Pointing – What the user is pointing at, e.g., "broom"
- Field of Vision – What the robot sees in the room, e.g., "broom", "floor", "mop"
- Real World Context – Tasks the robot already knows how to perform, e.g., "sweep the floor with a vacuum once a week"
- Dialog History – What the robot and user have been talking about recently, e.g., "clean the counters with a dry rag"

The input format is a list of phrases and a confidence score that each phrase is the correct phrase. Each part, or slot, of the task (i.e., action, object, tools, and condition) is fused separately. In the given example, Semantic Integration would receive the following inputs for the "tools" slot:

- Speech: null, 100% (there is nothing in "sweep the floor" about what tools to use)
- Pointing: broom, 100%

- Field of Vision: broom, 50%; mop, 50% ("floor" is an object and thus is not supplied as a possible "tool")
- Real World Context: vacuum, 100%
- Dialog History: dry rag, 100%

Semantic Integration takes the confidence score of each phrase and multiplies it by a weight assigned to each input and uses the one with the highest number as the phrase for that slot. In the example, if the input weights were as follows:

- Speech = 1.00
- Pointing = 0.80
- Field of Vision = 0.60
- Real World Context = 0.40
- Dialog History = 0.20

The fusion calculations would be as follows:

- broom: (100% x 0.80) + (50% x 0.60) = 1.10
- vacuum: (100% x 0.40) = 0.40
- mop: (50% x 0.60) = 0.30
- dry rag: (100% x 0.20) = 0.20

Semantic Integration would pick "broom" for the "tools" slot, which turns out to be what the user meant, even though, he or she never said it.

We did make one change to the Flippo Framework fusion algorithm. It was modified so that it returns no results if there is a tie for what goes in a slot. The assumption is that, with all the additional modalities, a tie is unlikely. But, if there is

still a tie for a slot, it is better to ask the user what was meant, than to arbitrarily select one of the ones with highest vote, which is what the Flippo Framework does.

The fusion weights were determined experimentally and are different for each slot as explained later. The weights are influenced by the error rates of the modalities (e.g., Speech, Pointing, Field of Vision), which indirectly influence the error rates of the contexts (e.g., Real World Context, Dialog History). Negative weights are possible. A negative weight does not indicate that the information from a modality should be ignored, as indicated by the experimental results discussed later.

4.6.3 Answer Fusion

For the answer frame, fusion is similar to that of tasks. There is only one slot, called "answer", which can have only one of two values: "yes" or "no". Each modality, Speech and Head Nodding, provides either a "yes" or "no" with a confidence score of 100% as the input to fusion. As an example, the fusion component might receive the following inputs for the "answer" slot:

- Speech: yes, 100%
- Head Nodding: no, 100%

As with tasks, the fusion algorithm takes the confidence score of each phrase and multiplies it by a weight assigned to each input and uses the one with the highest number as the "answer". In the example, if the input weights were as follows:

- Speech = 1.00
- Head Nodding = 0.80

The fusion calculations would be as follows:

- yes: (100% x 1.00) = 1.00
- no: (100% x 0.80) = 0.80

The fusion component would pick “yes” as the correct “answer”.

The fusion weights for “answer” slot are determined based on the ability of the modality to correctly recognize a “yes” or “no”. As discussed, the overall error rate of the Speech Recognizer was measured experimentally as 6.7%. In other words, the Speech Recognizer recognizes a phrase correctly 93.3% of the time. Similarly, the Head Nodding errors are simulated based on the results of the Head Nodding recognizer described by Morency and Darrell [60]. In their experiments, a head nod was recognized correctly 85.3% of the time.

Thus, the weights used for the “answer” slot for the two modalities are:

- Speech = 0.933
- Head Nodding = 0.853

4.6.4 Integration

After the frames are fused, Semantic Integration combines them into meaningful dialog units, which we call Fused Utterances, for Dialog Management to process.

Table 4.9 shows how fused frames are combined into Fused Utterances.

Fused Utterance	Frames	Description	Examples
SINGLE-TASK	1 Single-task	An **action** taken on an **object** using none, one, or more **tools** when a specified **condition** occurs.	Clean the counter with a sponge daily.
MULTIPLE-TASK	1 or more Single-task + 0 or more Connector	An ordered list of tasks to perform.	Wipe down the sink after loading the dishwasher. Wipe the mirror and faucet.
RBD	RbD	Robot Programming by Demonstration (RbD), which tells the robot that the user is going to show it how to perform a task.	Watch me. I'll show you.
LEARNING	Single-task + Learning + Single-task	Tells the robot that two tasks are similar.	Wipe down the stove top is like wipe down the sink.
ANSWER	Answer	Reply to a question posed by the robot.	Yes. No. I don't know.
REQUEST-REPEAT	RequestRepeat	Tells the robot to repeat what it just said.	What? Huh? I don't understand.
GREET	Greet	A greeting.	Welcome. Hello. How are you?
QUIT	Quit	Tells the robot that the user is going to start over.	Quit. Forget it.
NO-PARSE	None	Indicates the Speech Recognizer heard something, but did not recognize it.	The rain in Spain mainly falls in the plain.

Table 4.9: Fused Utterances.

4.7 Dialog Management

4.7.1 Overview

The Dialog Management component manages the dialog between the user and robot and resolves any ambiguities remaining after Semantic Integration, through clarifying questions.

Although we originally proposed to use TRINDIKIT from the TRINDI project to implement the Dialog Management component [83], it was decided to implement the Dialog Management in Java as discussed earlier.

The Dialog Management component still uses dialog plans similar to the TrindiKit design. The goal of a Dialog Plan is to translate a Fused Utterance from Semantic Integration into a list of valid tasks. In pursuing its goal, a Dialog Plan can ask questions and receive additional Fused Utterances from the user and launch other Dialog Plans as sub-plans in achieving its goal.

Dialog Plans are implemented as Java Threads. Each Dialog Plan resolves exactly one Fused Utterance. The Dialog Plan suspends itself after it asks a clarifying question. The Dialog Management module resumes the Dialog Plan when it receives a Fused Utterance from Semantic Integration.

Tasks are represented in the Task Database as an action, object, set of tools, condition, and a list of sub-tasks. A Primitive Task contains an empty sub-task list. A Compound Task contains a sub-task list of other Primitive and Compound Tasks. For example, the “Load the dishwasher” sub-task might actually be a Compound Task composed of two Primitive Tasks:

1. Pick up the dishes with your hands.
2. Put the dishes in the dishwasher with your hands.

4.7.2 Initial Dialog Plan

The Dialog Management component begins by executing the Initial Dialog Plan. The Initial Dialog Plan handles each Fused Utterance as follows:

SINGLE-TASK

1. Execute the Single Task Dialog Plan (later section) to process the Fused Utterance.
2. The Single Task Dialog Plan returns a Task; and Dialog Management adds it to the Task Database. The Task Database is a set of Tasks that the robot executes when the specified conditions occur. It is also provides the Real World Context.
3. Acknowledge the Task by saying, "OK. Anything else?"
4. Wait for next Fused Utterance.

MULTIPLE-TASK

1. Execute the Multiple Task Dialog Plan (later section) to process the Fused Utterance.
2. The Multiple Task Dialog Plan returns a list of Tasks; and Dialog Management adds them to the Task Database.
3. Acknowledge Tasks by saying, "OK. Anything else?"
4. Wait for next Fused Utterance.

ANSWER

1. If the user said Yes, say, "What can I do for you?"

2. If the user said No or “I don’t know”, presumably in response to the question in Step 1, say, “OK. Just let me know when you want me to do something.”
3. Wait for the next Fused Utterance.

REQUEST-REPEAT or GREET

1. Say, “Hello my name is Robot.”
2. Wait for the next Fused Utterance.

QUIT

1. Assume the user is responding to the question, “What can I do for you?” and say, “OK. Just let me know when you want me to do something.”
2. Wait for the next Fused Utterance.

RBD or LEARNING

1. These Fused Utterances do not make sense at this point in the dialog; therefore let the user know that by saying, “I'm sorry I don't understand what you mean. Please start over.”
2. Wait for next Fused Utterance.

NO-PARSE

1. Let the user know that the robot did not understand what the user said, by saying, “I'm sorry I don't understand what you mean. Please start over.”
2. Wait for next Fused Utterance

4.7.3 Single Task Dialog Plan

The Single Task Plan processes a SINGLE-TASK Fused Utterance as follows:

1. Try to fill the empty slots by asking the clarifying questions, as shown in Table 4.10.

Slot	Clarifying Question	Example
action	What do you want me to do to the <object> with the <tools> <condition>?	What do you want me to do to the ACCENT TABLES with the DUST RAG and PLEDGE DAILY?
object	What do you want me to <action> with the <tools> <condition>?	What do you want me to DUST with the DUST RAG and PLEDGE DAILY?
tools	What tools do you want me to use to <action> the <object> <condition>?	What tools do you want me to use to DUST the ACCENT TABLES DAILY?
condition	How often do you want me to <action> the <object> with the <tools>?	How often do you want me to DUST the ACCENT TABLES with the DUST RAG and PLEDGE?

Table 4.10: Clarifying Questions to Fill Empty Slots.

The clarifying question is adjusted if not all the slots are filled. For example, if only the object (e.g., ACCENT TABLE) and condition (e.g., DAILY) are known, the robot would try to fill the action slot with the question, "What do you want me to do to the ACCENT TABLE DAILY?"

2. Patiently try to fill the slots by repeating the clarifying questions until a QUIT Fused Utterance is received from Semantic Integration.
3. If the QUIT Fused Utterance is received, acknowledge the user wants to start over by saying, "OK. Just let me know when you want me to do something.", and then start over by executing the Initial Dialog Plan.
4. If any of the other Fused Utterances are received while trying to fill empty slots, say, "I'm sorry. I don't understand what you mean." and repeat the clarifying question again.

5. Once all the slots are filled, learn how to perform the task by executing the Robot Learning Dialog Plan.

4.7.4 Robot Learning Dialog Plan

The purpose of this Dialog Plan is to learn a new Task, which is currently not in the Task Database. The robot learns the new Task in one of the following ways:

1. The user shows the robot how to perform the Task (Robot Learning by Demonstration – RbD). The user saying, "I'll show you", simulates this. Then, the Task is added to the Task Data Base as new Primitive Task.
2. The user teaches the robot how to perform the Task by saying it is like another Task the robot already knows how to do (LEARNING Fused Utterance).
3. The user teaches the robot how to perform the Task by breaking it down into sub-tasks that the robot does know how to do (Compound Task). Another Dialog Plan is executed to manage this Instruction Based Learning (IBL) dialog. Then, the Task is added to the Task Data Base as a new Compound Task.

The plan begins by the robot asking the user how to perform the task. What happens next depends on what the user says, as shown below for each possible Fused Utterance:

RBD (Case 1)

1. Assume the user shows the robot how to perform the Task.
2. Return the task being learned as a Primitive Task to the Dialog Plan that called this one. For example:

User: Sweep the floors daily. (Initial Dialog Plan executes Single Task Dialog Plan)

Robot: What tools do you want me to use to sweep the floors daily?

User: A broom. (Single Task Dialog Plan executes Robot Learning Dialog Plan)

Robot: How do I sweep the floors with a broom daily?

User: I'll show you. (Robot Learning Dialog Plan returns "sweep the floors with a broom daily" task to Single Task Dialog Plan; Single Task Dialog Plan returns it to Initial Dialog Plan; and Initial Dialog Plan saves it in the Task Database and waits for the next utterance from the user)

LEARNING (Case 2)

1. Extract the two Tasks from the Fused Utterance.
2. If two Tasks were not found, say, "I'm sorry I don't understand what you mean. Please start over." Then start the Robot Learning Dialog Plan over by asking the user how to perform the task again.
3. If two Tasks were found, determine which one is the known Task and which one is the unknown task by assuming the found task, which matches the task we are trying to learn the closest, is the unknown task and the other one is the known task. For example, given the following dialog:

   Robot: How do I clean the foyer daily?

   User: Clean the family room is like clean the foyer.

then,

Task Robot is Trying to Learn =

action: clean

object: foyer

tools: none

condition: daily

Task 1 from Fused Utterance =

action: clean

object: family room

tools: unknown

condition: unknown

Task 2 from Fused Utterance =

action: clean

object: foyer

tools: unknown

condition: unknown

Now, Task 2 is more like the Task the Robot is Trying to Learn, so

Unknown Task = Task 2 (clean the foyer)

Known Task = Task 1 (clean the family room)

This algorithm allows the user to speak more freely with the robot by not requiring a very strict grammar to be followed. For example, if the user had said,

"Clean the foyer is like clean the family room." then it would have ended up with the same unknown and known task:

Unknown Task = Task 1 (clean the foyer)

Known Task = Task 2 (clean the family room)

4. Find the Known Task in the Task Database.
5. If a perfect match is found (i.e., action, object, tools, and condition slots match), then use it.
6. If a perfect match is not found, then find the first partial match where three out of the four slots match and use it.
7. If neither a perfect or partial match is found, say:

   "I'm sorry I don't understand what you mean."

   "I don't know how to do the first task or how to do the second task."

   "Please start over."

   And, then start the Robot Learning Dialog Plan over by asking the user how to perform the Task again.
8. If either a perfect or partial match is found, copy the subtasks from the known task to the one being learned.
9. Return the Task being learned (with the subtasks from the known task) to the Dialog Plan that called this one. For example, given that the known task above (clean the family room) is defined as:

   action: clean

   object: family room

tools: none

condition: daily

subtasks:

fluff the cushions with your hands daily

wipe the tabletops with pledge and a dry rag daily

Now the newly created Compound Task that is returned is defined as:

action: clean

object: foyer

tools: none

condition: daily

subtasks:

fluff the cushions with your hands daily

wipe the tabletops with pledge and a dry rag daily

SINGLE-TASK or MULTIPLE-TASK (Case 3)

1. Assume this is the beginning of a Compound Task definition and execute the Compound Task Dialog Plan. The Compound Task Dialog Plan handles both simple sentences (e.g., vacuum the floors) in SINGLE-TASK Fused Utterances or compound sentences (e.g., wipe down the sink after loading the dishwasher) in MULTIPLE-TASK Fused Utterances.
2. Return the task being learned as a Compound Task to the Dialog Plan that called this one. For example:

a. User: Clean the kitchen daily. (Initial Dialog Plan executes Single Task Dialog Plan)
b. Robot: What tools do you want me to use to clean the kitchen daily?
c. User: None. (Single Task Dialog Plan executes Robot Learning Dialog Plan)
d. Robot: How do I clean the kitchen with no tools daily?
e. User: Wipe down the stovetop. (Robot Learning Dialog Plan executes Compound Task Dialog Plan)
f. User and robot engage in dialog to define additional subtasks (e.g., "wipe down the counters" and "vacuum the floors")
g. Compound Task Dialog Plan returns the Compound Task to this Dialog Plan, which returns it to the Single Task Dialog Plan; Single Task Dialog Plan returns it to Initial Dialog Plan; and Initial Dialog Plan saves it in the Task Database and waits for the next utterance from the user)

ANSWER

1. If the answer is Yes, assume the user did not understand the question and repeat the question, "How do I perform this task?"
2. If the answer is not Yes, say "I'm sorry I don't understand what you mean. Please start over." and restart the dialog by returning to the Dialog Plan that called this one with an empty list of Tasks.

REQUEST-REPEAT

1. Repeat the question, “How do I perform this task?”

QUIT

1. Say “OK. Just let me know when you want me to do something.” and restart the dialog by returning to the Dialog Plan that called this one with an empty list of Tasks.

NO-PARSE or GREET

1. Say, “I'm sorry. I don't understand what you mean.”
2. Repeat the question, “How do I perform this task?”

4.7.5 Compound Task Dialog Plan

The Compound Task Dialog Plan handles both simple sentences (e.g., vacuum the floors) in SINGLE-TASK Fused Utterances or compound sentences (e.g., wipe down the sink after loading the dishwasher) in MULTIPLE-TASK Fused Utterances. This allows the user to express a Compound Task in a variety of ways, for example:

- Wipe down the sink (No connector and only one task).
- Sweep and mop the floors (No connector, but more than one task).
- Then, dust the cabinets (A connector, but only one task).
- Wipe down the sink after loading the dishwasher (A connector and more than one task).

Normally, in a Compound Task the order of executing the sub-tasks is assumed to be in the order that the user expressed them to the robot. For example, the order of execution for “sweep and mop the floors” is assumed to be:

1. Sweep the floors.

2. Mop the floors.

But, the connectors, “before” and “after”, take precedence over the order of expression. For example, if the user says, “Wipe down the sink after loading the dishwasher. The order of expression is:

1. Wipe down the sink.
2. Load the dishwasher.

However, the connector, “after”, pre-empts the order of expression, which means the user really wants the order of execution to be:

1. Load the dishwasher.
2. Wipe down the sink.

The primary goal of the Compound Task Dialog Plan is to combine a sequence of utterances from the user, which may be spoken all at the same time, or over a period of time into a Compound Task with the sub-tasks in the correct order.

The Compound Task Dialog Plan is invoked by another Dialog Plan to process a SINGLE-TASK or MULTIPLE-TASK Fused Utterance. After the Fused Utterance is processed, the robot asks the user if there are more subtasks in the Compound Task by saying, “Are there more steps in the process to <action> the <object> with the <tools> <condition>?” (e.g., Are there more steps in the process to clean the kitchen with no tools daily?). If the user answers Yes, the Compound Task Dialog Plan processes the next SINGLE-TASK or MULTIPLE-TASK Fused Utterance. If the user answers No, the Compound Task Dialog Plan returns the finished Compound Task to the Dialog Plan that invoked it.

The Compound Task Dialog Plan stops and returns an empty Task list to the Dialog Plan that invoked it, whenever the user says a QUIT Fused Utterance.

The Compound Task Dialog Plan responds to all other Fused Utterances (e.g., GREET) by saying, “I'm sorry I don't understand what you mean. Are there more steps in the process to <action> the <object> with the <tools> <condition>?”

The Compound Task Dialog Plan uses a state table to keep track of how subtasks are added to the subtask list based on the use of NEXT, BEFORE, and AFTER, where these are connector frames in a MULTIPLE-TASK Fused Utterance. These can also be thought of as generic representations of these words in an English sentence (e.g., Wipe down the sink AFTER loading the dishwasher).

The state table allows the user to provide a list of sub-tasks using adverbs such as, “before”, “after”, and “next”. It can handle the adverb at the beginning of a sentence (e.g., “After loading the dishwasher, wipe down the sink.”) or in the middle (e.g., “Wipe down the sink after loading the dishwasher.”).

The state table is shown in Table 4.11.

		Input			
		Task	**NEXT**	**BEFORE**	**AFTER**
State	**Z: Initial State**	T	N	B	A
		Save current subtask as last subtask	No action	No action	No action
		False	False	False	False
	T: Task Received	T	TN	TB	TA
		Add last subtask to end of subtask list and save current subtask as last subtask	No action	No action	No action
		False	False	False	False
	N: NEXT Received	NT	N	B	A
		Save current subtask as last subtask	No action	No action	No action
		False	True	True	True
	B: BEFORE Received	BT	N	B	A
		Save current subtask as last subtask	No action	No action	No action
		False	True	True	True
	A: AFTER Received	AT	N	B	A
		Save current subtask as last subtask	No action	No action	No action
		False	True	True	True
	TN: Task NEXT Received	Z	TN	TB	TA
		Add last subtask before current subtask	No action	No action	No action
		True if insertion is ambiguous	True	False	False
	TB: Task BEFORE Received	Z	TB	TB	Z
		Add last subtask before current subtask	No action	No action	Add last subtask to end of subtask list
		True if insertion is ambiguous	True	True	True
	TA: Task AFTER Received	Z	TA	Z	TA
		Add current subtask before last subtask	No action	Add last subtask to end of subtask list	No action
		True if insertion is ambiguous	True	True	True
	NT: NEXT Task Received	Z	TN	TB	TA
		Add last subtask before current subtask	No action	No action	No action
		True if insertion is ambiguous	False	False	False
	BT: BEFORE Task Received	Z	BT	Z	Z
		Add current subtask before last subtask	No action	Add last subtask to end of subtask list	Add last subtask to end of subtask list
		True if insertion is ambiguous	True	True	True
	AT: AFTER Task Received	Z	AT	Z	Z
		Add last subtask before current subtask	No action	Add last subtask to end of subtask list	Add last subtask to end of subtask list
		True if insertion is ambiguous	True	True	True

Table 4.11: NEXT, BEFORE, and AFTER State Table.

The input to the state table is the current frame, which can be a single-task, or one of three connectors: NEXT, BEFORE, or AFTER. The current state represents what has been received before. The output of the state table is a tuple of the next state, the action the Compound Task Dialog Plan takes with regard to adding subtasks to the Compound Task, and the cautious setting. An example of a tuple in Table 4.11 is the case when the state is Z and the input is Task:

Next State: T

Action: Save current subtask as last subtask

Cautious Setting: False

The cautious setting determines how the Dialog Management component grounds the dialog and is explained in a later section.

An example of how the state table works is illustrated with the following dialog:

Robot: How do I clean the kitchen with no tools daily?

User: Wipe down the sink after loading the dishwasher.

Robot: Are there more steps in the process to clean the kitchen with no tools daily?

User: Wipe down the stovetop.

Robot: Are there more steps in the process to clean the kitchen with no tools daily?

User: Wipe down the counters.

Robot: Are there more steps in the process to clean the kitchen with no tools daily?

User: No.

Table 4.12 shows how the Compound Task Dialog Plan would process the user utterances. For the purposes of clarity, the dialog to fill the missing slots has been omitted.

User Utterance	Frames	Current State	Next State	Action	Current Subtask	Last Subtask	Subtask List
Wipe down the sink after loading the dishwasher.	Task: Wipe down the sink	Z	T	Save current subtask as last subtask	Wipe down the sink	Wipe down the sink	empty
	AFTER	T	TA	No action	Wipe down the sink	Wipe down the sink	empty
	Task: Load the dishwasher	TA	Z	Add current subtask before last subtask	Load the dishwasher	Wipe down the sink	1. Load the dishwasher 2. Wipe down the sink
Wipe down the stovetop.	Task: Wipe down the stovetop	Z	T	Save current subtask as last subtask	Wipe down the stovetop	Wipe down the stovetop	1. Load the dishwasher 2. Wipe down the sink
Wipe down the counters.	Task: Wipe down the counters	T	T	Add last subtask to end of subtask list & save current subtask as last subtask	Wipe down the counters	Wipe down the counters	1. Load the dishwasher 2. Wipe down the sink 3. Wipe down the stovetop
No.	State table not used; last subtask is added to end of subtask list and completed Compound Task is returned to the invoking Dialog Plan				Wipe down the counters	Wipe down the counters	1. Load the dishwasher 2. Wipe down the sink 3. Wipe down the stovetop 4. Wipe down the counters

Table 4.12: Compound Task Dialog Plan Example.

4.7.6 Multiple Task Dialog Plan

The Initial Dialog Plan only executes Multiple Task Dialog Plan when the robot is not learning a Compound Task and the user utters a MULTIPLE-TASK Fused Utterance (e.g., wipe down the sink after loading the dishwasher). No other Dialog Plans execute this plan.

The Multiple Task Plan processes a MULTIPLE-TASK Fused Utterance as follows:

1. Create the skeleton of a new Compound Task using the action, object, tools, and condition slots from the first frame in the Fused Utterance. For example, if the user said "Sweep and mop the floors daily", the Compound Task skeleton would be:

   action: sweep

   object: floors

   tools: none

   condition: daily

2. Execute the Compound Task Dialog Plan to fill in the subtasks of the Compound Task (see Section 4.7.5).

4.7.7 Grounding

As explained earlier, we used three different strategies for grounding in our experiments, Optimistic, Cautious, and Pessimistic. The Dialog Management component can operate using any of these three strategies, or modes. The user in the experiments selects the grounding mode with the pull-down window on the HRI Environment Simulator window, as shown in Figure 4.14.

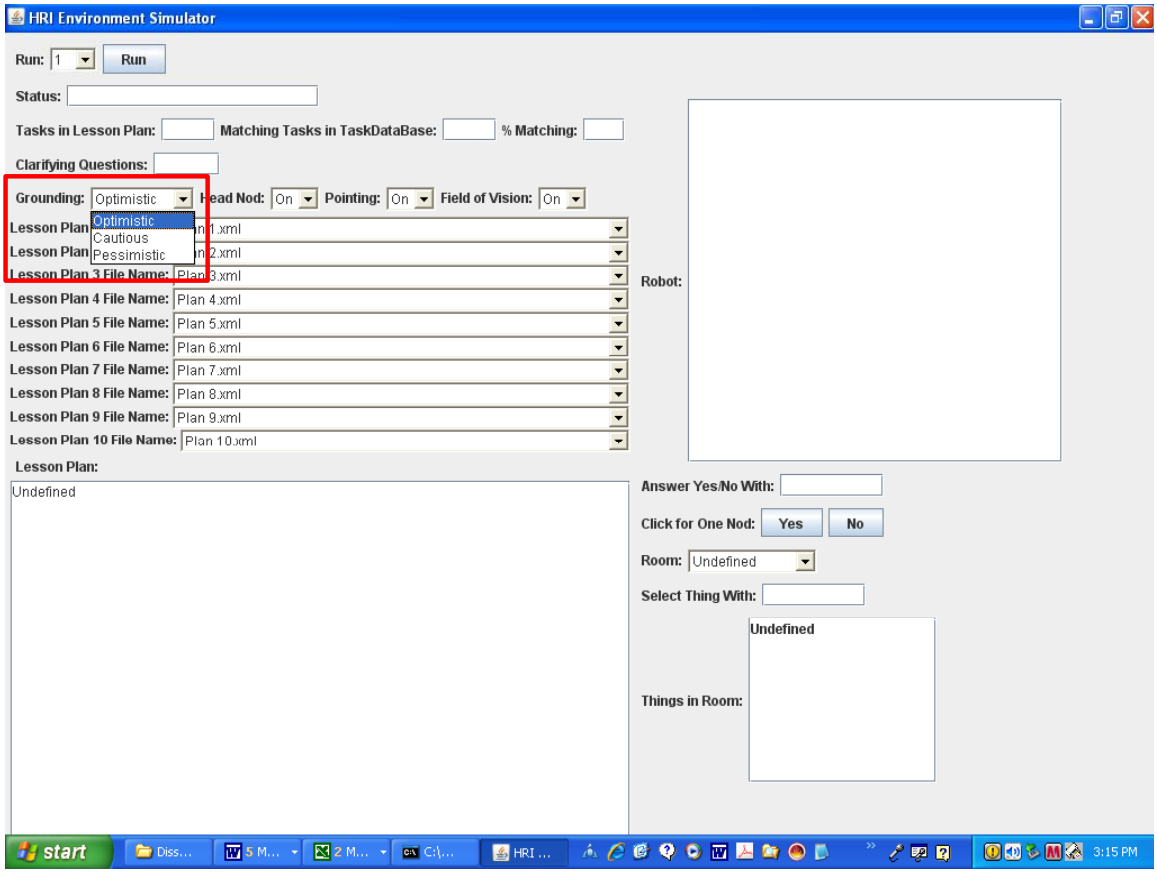

Figure 4.14: Grounding Mode Selection in the HRI Environment Simulator.

Whenever the Dialog Management makes an assumption about what the user means, it invokes the "check assumption" method. If the robot is in Optimistic mode, "check assumption" always returns true. If the robot is in Pessimistic mode, "check assumption" will ask the user to verify the assumption with a "yes" or "no". If the robot is in Cautious mode, it will ask the user to verify the assumption, if there is a possibility the assumption is not true. In Cautious mode, the invoking Dialog Plan is the one that decides whether an utterance is ambiguous and whether "check assumption" should verify the assumption or not.

During the dialog, the user's intentions are ambiguous in the following situations:

- Because of audio and visual errors from the modalities (both real and simulated) and incorrect assumptions by the contexts, all the slots may not be filled correctly during Fusion.
- When the user responds to "How do I perform this task?" with a Task, did the user misunderstand the robot, or is it giving the first step in a Compound Task?
- When the user responds to "Are there more steps in the process?" with a Task, did the user misunderstand the robot, or is the user giving the next step in a process?
- When a Fused Utterance contains two or more single-tasks and at least one BEFORE or AFTER, the intentions of the user are ambiguous. For example, if the user says:

  wipe down the sink before mopping the floors sweep the floors

  Does the user mean?

  Wipe down the sink.

  Before mopping the floors, sweep the floors.

  Or, does the user mean?

  Wipe down the sink before mopping the floors.

  Sweep the floors.

  The subtask order for the first interpretation is:

  1. Wipe down the sink.
  2. Sweep the floors.
  3. Mop the floors.

  The subtask order for the second interpretation is:

1. Wipe down the sink.
2. Mop the floors.
3. Sweep the floors.

- When the user teaches the robot how to perform a Task by saying it is like another Task that the robot already knows how to do, and the robot finds a match in the Task Database. Because of audio and visual errors from the modalities (both real and simulated) and incorrect assumptions by the contexts, all the slots may not be filled correctly during Fusion. Consequently, the known Task may be incorrect.

Table 4.13 illustrates what the Dialog Management component does in each of these ambiguous situations for each of the three grounding modes.

		Grounding		
		Optimistic	**Cautious**	**Pessimistic**
Ambiguous Situation	Slots may not be filled correctly during Fusion	Assumes slots are filled correctly	Only verifies tools slot and assumes all other slots are filled correctly	Verifies all slots
	User responds to "How do I perform this task?" with a Task	Assumes Task is first step	Verifies Task is first step	Verifies Task is first step
	User responds to "Are there more steps in the process?" with a Task	Assumes Task is next step	Verifies Task is next step	Verifies Task is next step
	Fused Utterance contains two or more single-tasks and at least one BEFORE or AFTER	Assumes a subtask order	Verifies a subtask order, only if this situation occurs	Always verifies a subtask order
	Match found in the Task Database for "is like" Task	Assumes match is the known Task	Verifies partial match is the known Task and assumes perfect match is the known Task	Verifies partial or perfect match is the known Task

Table 4.13: Grounding During Ambiguous Situations.

4.7.8 Accommodation

Traum *et al.* refer to the ability to handle out of sequence answers as "accommodation" [83]. Humans use accommodation all the time, but it is not easy to handle accommodation programmatically. The Dialog Management component supports two forms of accommodation:

Slot Filling – When Dialog Management is trying to fill slots, it asks the user for each slot individually, but will accept values for any slot. For example in the following dialog:

User: Wipe down the counters.

Robot: What tools do you want me to use to wipe down the counters?

User: Daily.

Robot: What tools do you want me to use to wipe down the counters daily?

User: A sponge.

In its first question, the robot did not ask for the condition, but that is what the user answered with. So, the robot accommodated the out of sequence answer by accepting it as the condition, and acknowledged the accommodation by including it in the question asking for the tools again.

Compound Task Process Definition – When using Optimistic grounding and the user responds to "Are there more steps in the process?" with a Task, Dialog Management accommodates the out of sequence answer as the next Task in the process.

4.8 Visual Feedback

Feedback from the robot to the user is conceptually the inverse of modality Fusion, and is frequently referred to as fission [33]. When the robot wants to respond to the user, the Dialog Management component will generate output to send to the Visual Feedback component, as well as sending corresponding text to the natural language generator, or Audio Feedback.

For the experiments, the HRI Environment Simulator provides limited Visual Feedback as follows:

- The Head Nod buttons darken when the user clicks down on them and returns to normal when the user clicks up as shown in Figure 4.15.
- The Thing pointed at darkens when the user selects it as shown in Figure 4.7 and returns to normal when the robot "sees" what the user is pointing at, as shown in Figure 4.16.

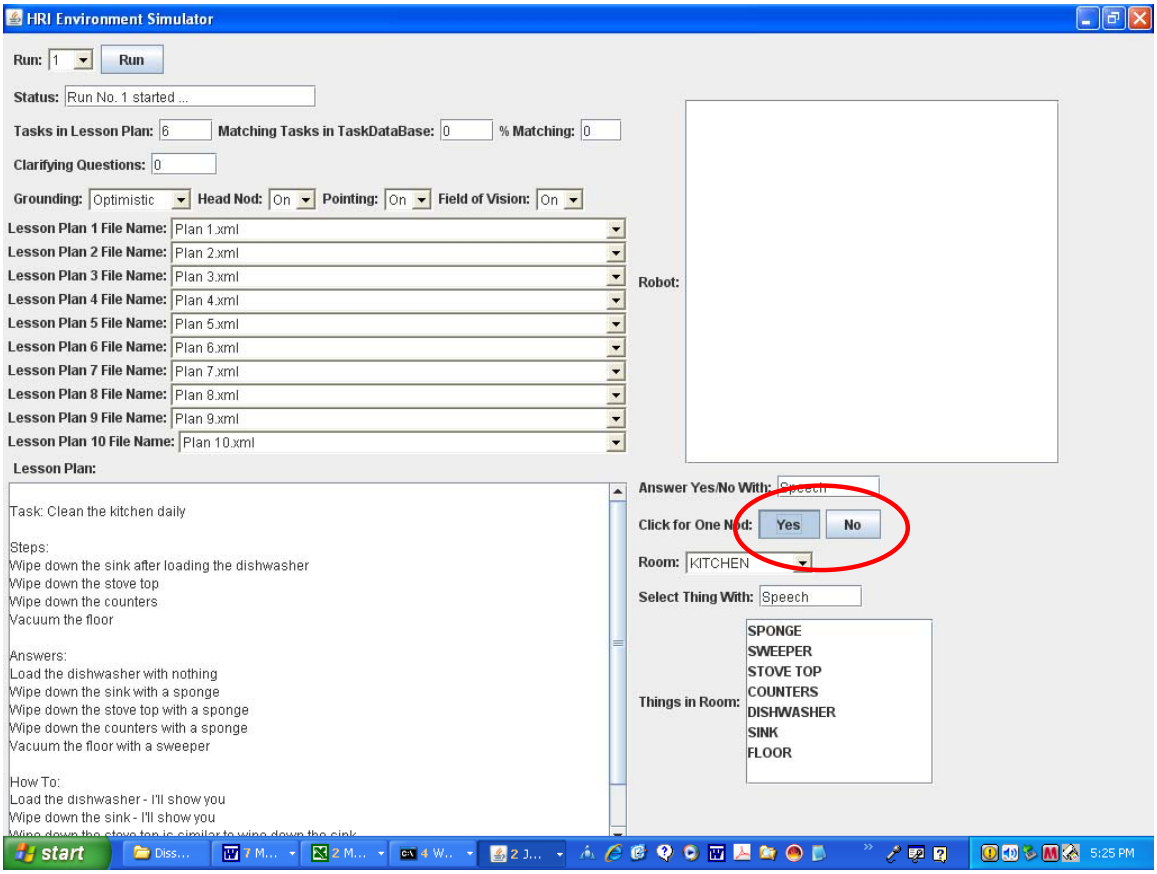

Figure 4.15: Visual Feedback of Head Nod.

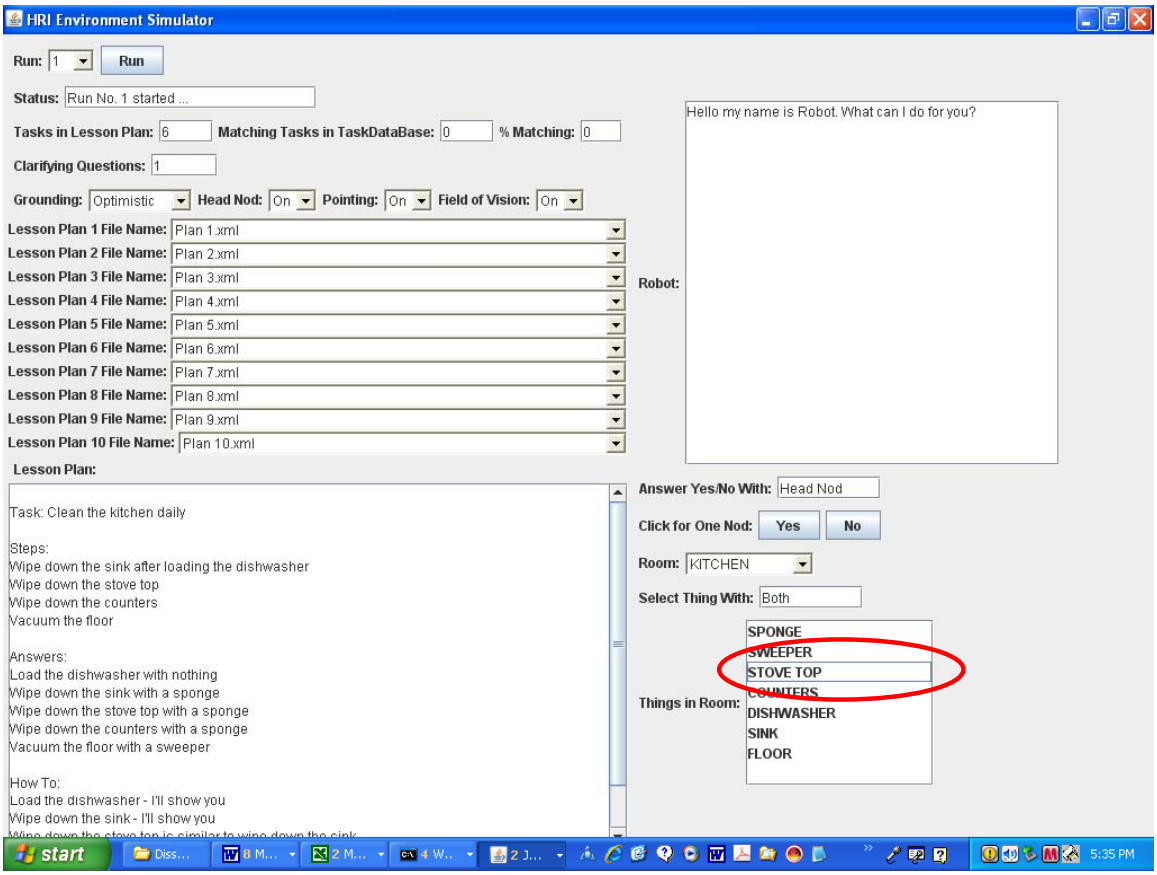

Figure 4.16: Visual Feedback of Pointing.

4.9 Audio Feedback

We did not implement a natural language generator for the experiments. Instead, what the robot says is displayed in the HRI Environment Simulator, as shown in Figure 4.17.

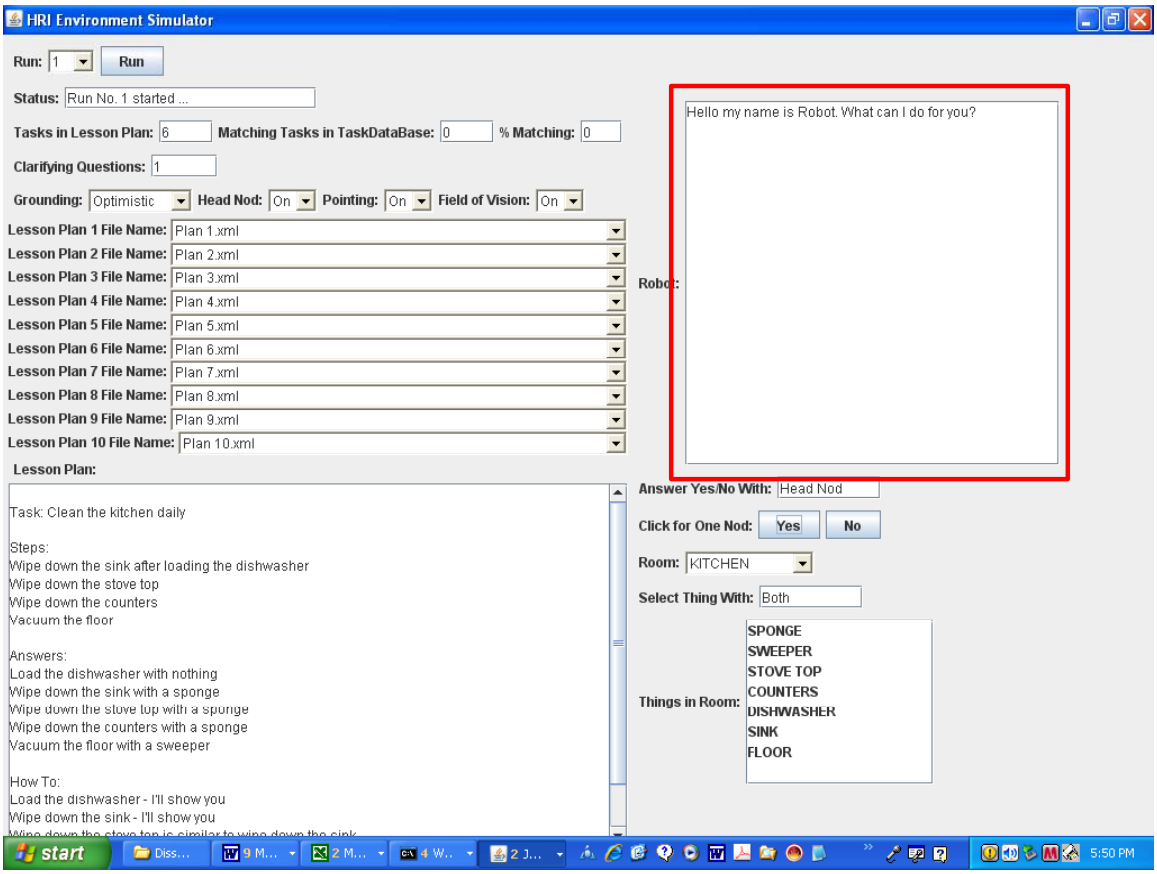

Figure 4.17: Audio Feedback Simulation in the HRI Environment Simulator.

Other components invoke the Audio Feedback component to generate strings by calling generic Java methods, such as "What do you want me to do to the <object> with the <tools> <condition>?" In this way, the responses are centralized and can be changed easily. Table 4.14 shows the generic methods and the strings they produce.

Java Method	Arguments	Description	String Produced
askForSlot	task, slot	Fill an empty Slot	Action – What do you want me to do to the <object> with the <tools> <condition>? Object – What do you want me to <action> with the <tools> <condition>? Tools – What tools do you want me to use to <action> the <object> <condition>? Condition – How often do you want me to <action> the <object> with the <tools>?

Java Method	Arguments	Description	String Produced
verifySlots	task	Verify all filled Slots are filled correctly	Are you asking me to <action> the <object> with the <tools> <condition>?
confirmTask	task	Confirm a task	I'm just confirming. Do you want me to <action> the <object> with the <tools> <condition>?
confirmKnownTask	task	Confirm a known Task	I'm just confirming. I already know how to <action> the <object> with the <tools> <condition>. Is that what you want me to do?
confirmNewTask	task	Confirm a new Task	I'm just confirming. Is the new task to <action> the <object> with the <tools> <condition>?
askIfMoreSubtasks	task	Ask if there more subtasks	Are there more steps in the process to <action> the <object> with the <tools> <condition>?
askForNextSubtask	task	Ask for the next subtask in a Compound Task	OK. What is the next step in the process to <action> the <object> with the <tools> <condition>?
confirmCompoundTaskNotDone	task	Confirm Compound Task is NOT completed	I'm just confirming. Is this another step to <action> the <object> with the <tools> <condition>?
confirmCompoundTaskDone	task	Confirm Compound Task is completed	I'm just confirming. Are you done telling me the steps to <action> the <object> with the <tools> <condition>?
verifySubtasks	task	Verify subtasks in a Compound Task	I'm just confirming. Are these the steps to <action> the <object> with the <tools> <condition>? <action> the <object> with the <tools> <condition> <action> the <object> with the <tools> <condition> … <action> the <object> with the <tools> <condition>
askForSubtasksAgain	none	Ask subtasks in a Compound Task again	OK, please give me the steps for this process again.
verifyUnknownIsLikeKnown	unknown task, known task	Verify unknown task is like known task	I know how to <action> the <object> with the <tools> <condition>. Is <action> the <object> with the <tools> <condition> done the same way as that?
verifyNewIsLikeTwoKnowns	unknown task, 2 known tasks	Verify an unknown task is like two known tasks	I know how to <action> the <object> with the <tools> <condition> and how to <action> the <object> with the <tools> <condition>. Is <action> the <object> with the <tools> <condition> done the same way as those?
unknownTasks	2 unknown tasks	Tell user neither Task in Learning frame was known	I don't know how to <action1> the <object1> with the <tools1> <condition1> or how to <action2> the <object2> with the <tools2> <condition2>.

Java Method	Arguments	Description	String Produced
speak	string	Say string	string
robotIntroduction	none		Hello my name is Robot.
robotRequest	none		What can I do for you?
pleaseStartOver	none		Please start over.
tryAgain	none		OK. Let's try again.
robotRequestRepeat	none		I'm sorry I don't understand what you mean. Please start over.
robotStandby	none		OK. Just let me know when you want me to do something.
robotAcksTask	none		OK. Anything else?
iDontUnderstand	none		I'm sorry I don't understand what you mean.

Table 4.14: Audio Feedback Methods.

The Audio Feedback component does more than just display character strings. The most significant is to form a "task clause" from the internal representation of a task, which only includes the action, object, tools, and condition. It automatically accounts for missing slots and no tools to make the speech more natural.

For example if fusion produces the following task:

action = clean

object = unknown

tools = none

condition = daily

The robot asks for the object slot with, "What do you want me to clean with no tools daily?"

If more than one slot is missing, as in the following example:

action = clean

object = unknown

tools = none

condition = unknown

The robot asks for the object slot with, “What do you want me to clean with no tools?”

Also, this component removes the “-ed” and “-ing” suffixes from action verbs. For example, if the task is:

action = loading

object = unknown

tools = none

condition = daily

The robot asks for the object slot with, “What do you want me to load with no tools?”

The Audio Feedback performs one other function of counting clarifying questions, which is one of the experimental metrics. The number of clarifying questions is displayed in the HRI Environment Simulator as shown in Figure 4.18. The number of clarifying questions is reset to zero every time the Run button is clicked.

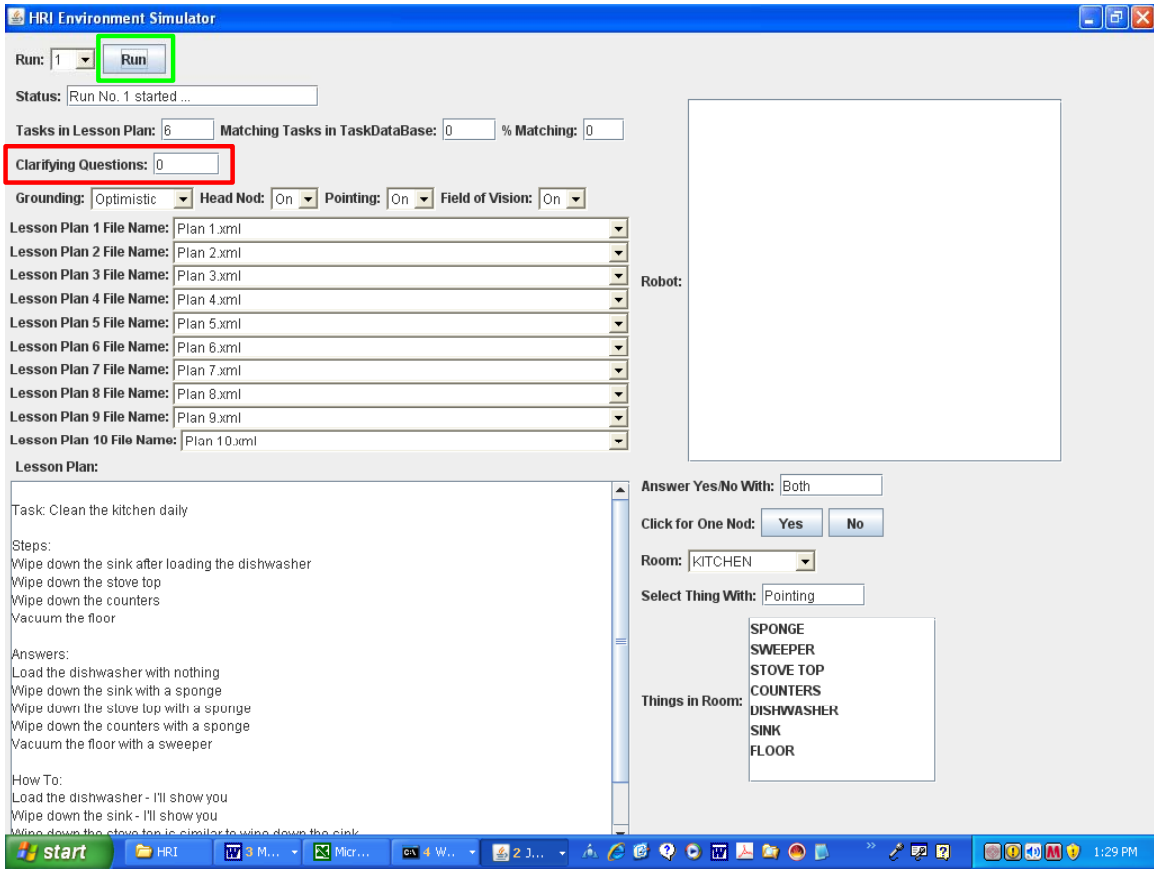

Figure 4.18: Clarifying Questions Counter.

4.10 Task Management and Robot Controller

In a fully functioning robot, the Task Management component would break down all tasks into action primitives. The Robot Controller would take these primitives and execute them using the robot's sensory and mechanical systems.

We did not simulate the Task Management component because it was done by Blythe and Reilly who simulated a household robot agent, Mr. Fixit, who could, along with other tasks, vacuum and clean up broken cups [4]. We did not simulate the Robot Controller because it was beyond the scope of this research.

Another important function of the HRI Environment Simulator, which has not been discussed yet, is determining whether the robot is learning the tasks correctly.

Determining whether the robot is learning the tasks correctly is one of two experimental metrics; the other being the number of clarifying questions.

When the user clicks on the Run button on the HRI Environment Simulator, the system loads a Golden Task Database with the tasks in the Lesson Plan being taught, and updates the Tasks in Lesson Plans field in the HRI Environment Simulator as shown in Figure 4.19. The Golden Task Database represents the correct set of Tasks the robot is supposed to learn from this Lesson Plan.

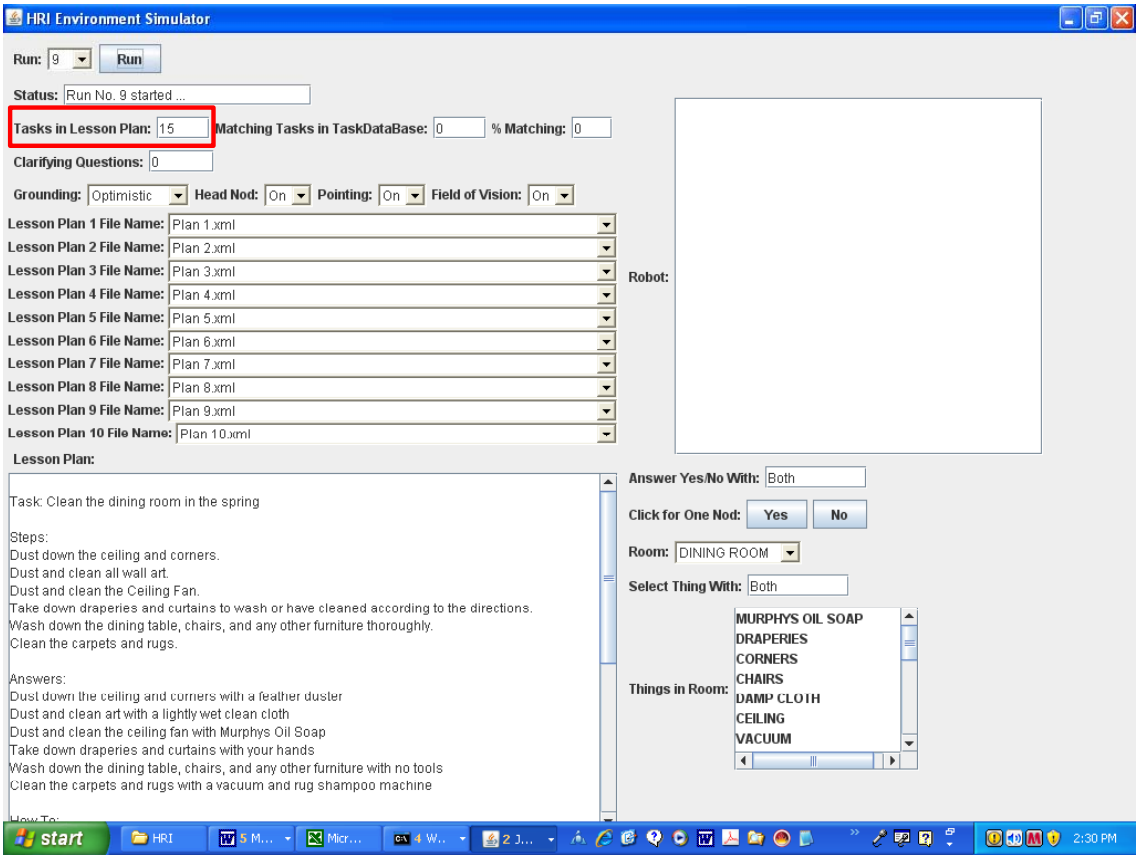

Figure 4.19: Tasks in Lesson Plan in HRI Environment Simulator.

The system counts each unique Compound Task as one task and each unique Primitive Task that make up a Compound Task as one task. For example, the Golden

Task Database for Lesson Plan 9 contains 15 tasks (shown in Appendix). Figure 4.20 illustrates how the system counts the number of tasks.

Golden Task No.	Lesson Plan 9 XML
	<task>
1	CLEAN,DINING ROOM,[],SPRING,[
2	DUST DOWN,CEILING,[FEATHER DUSTER],SPRING,[];
3	DUST DOWN,CORNERS,[FEATHER DUSTER],SPRING,[];
4	DUST AND CLEAN,ART,[LIGHTLY WET CLEAN CLOTH],SPRING,[];
5	DUST AND CLEAN,CEILING FAN,[MURPHYS OIL SOAP],SPRING,[];
6	TAKE DOWN,DRAPERIES,[HANDS],SPRING,[];
7	TAKE DOWN,CURTAINS,[HANDS],SPRING,[];
8	WASH DOWN,DINING TABLE,[],SPRING,[
9	CLEAN,WOOD,[DAMP CLOTH],SPRING,[];
10	OIL,WOOD,[FURNITURE OIL],SPRING,[];
	];
11	WASH DOWN,CHAIRS,[],SPRING,[
same as 9	CLEAN,WOOD,[DAMP CLOTH],SPRING,[];
same as 10	OIL,WOOD,[FURNITURE OIL],SPRING,[];
12	SPOT CLEAN,UPHOLSTERY,[SPOT CLEANER],SPRING,[];
	];
13	WASH DOWN,OTHER FURNITURE,[],SPRING,[
same as 9	CLEAN,WOOD,[DAMP CLOTH],SPRING,[];
same as 10	OIL,WOOD,[FURNITURE OIL],SPRING,[];
same as 12	SPOT CLEAN,UPHOLSTERY,[SPOT CLEANER],SPRING,[];
	];
14	CLEAN,CARPETS,[VACUUM,RUG SHAMPOO MACHINE],SPRING,[];
15	CLEAN,RUGS,[VACUUM,RUG SHAMPOO MACHINE],SPRING,[];
	];
	</task>

Figure 4.20: Golden Tasks for Lesson Plan 9.

Each time Dialog Management adds a new task to the Task Database, the system compares it with the Golden Task Database. If the new task is in the Golden Task Database, then the "Matching Tasks in TaskDataBase" and "% Matching" fields in the HRI Environment Simulator are updated as displayed in Figure 4.21.

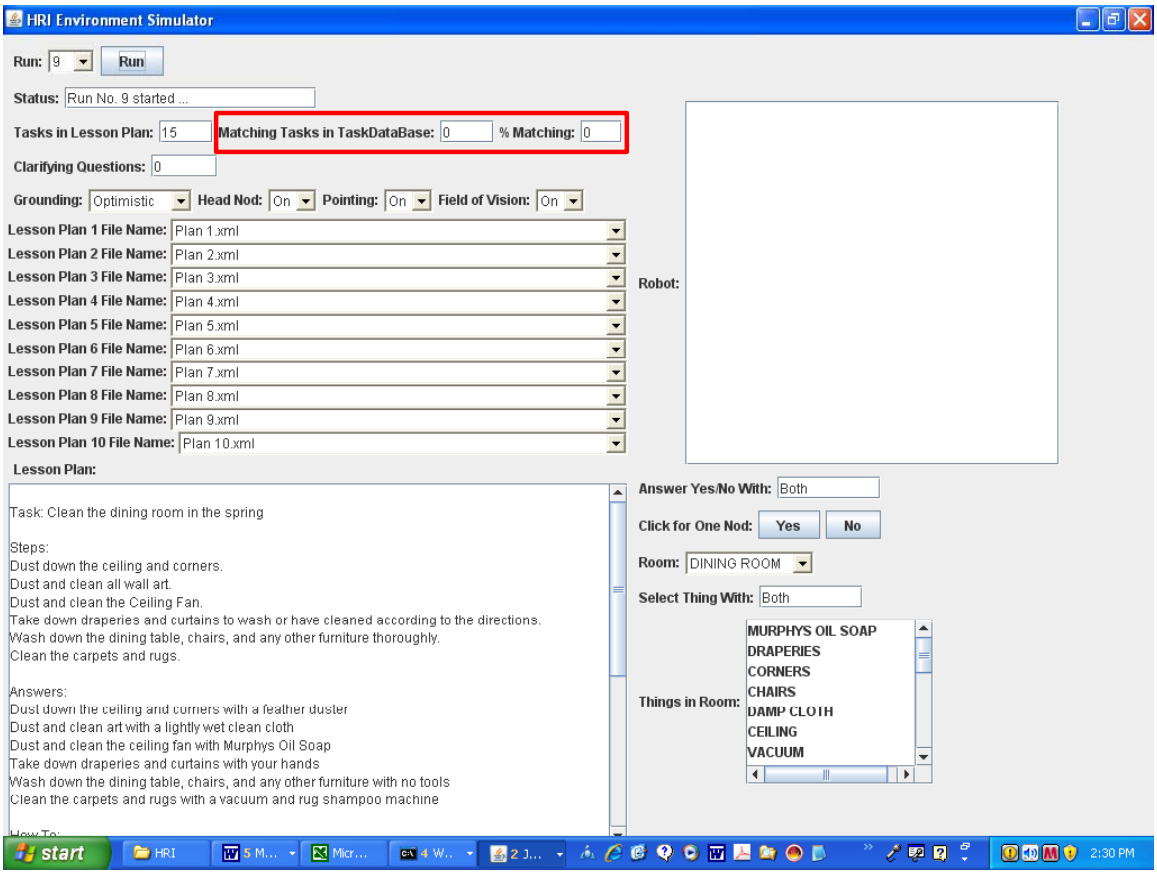

Figure 4.21: Matching Tasks in TaskDataBase.

Chapter 5 Semantic Integration

We have defined Semantic Integration as the component that fuses input from multiple modalities (e.g., Speech, Pointing, Field of Vision) and contexts (e.g., Real World Context, Dialog History) into a task for the robot to perform (e.g., sweep the floor).

The fusion weights were determined experimentally and are different for each slot. The weights are influenced by the error rates of the three modalities (e.g., Speech, Pointing, Field of Vision), which indirectly influence the error rates of the contexts (e.g., Real World Context, Dialog History). As stated earlier, the error rates are:

- Speech: 6.7% (overall)
- Pointing: true-positive rate of 78.3% and a false-positive rate of 11.6% [62]
- Field of Vision: true-positive rate of 95.0% and a false-positive rate of 2.3% [39]

The fusion weights used in these experiments were determined using the following approach [41]:

- Prepare 20 log files containing all 166 tasks in the corpus.
- Determine 20 sets of weights for each of the 20 logs with no-errors.
- Cluster the 20 sets of weights using k-means clustering [56]
- Determine the six sets of "best" weights for each slot using the following methods:
    - Reconstruction Error and Peakedness
    - Centroid
    - Closest to Centroid

    - Frequency

- Determine 20 sets of weights for each of the 20 logs with errors
- Cluster the 20 sets of weights
- Determine the six sets of "best" weights for each slot using the following methods:
    - Reconstruction Error and Peakedness
    - Centroid
    - Closest to Centroid
    - Frequency
- Calculate the Fusion Error Rate (FER) for the 20 logs with errors using the 12 sets of "best" weights determined above.
- Plot 240 FER to determine which of the 12 sets of "best" weights to use.

This approach is described in detail in this chapter.

5.1 Preparation of Log Files

We started with seven log files that covered all 166 tasks in the corpus. These log files were created during the debug phase of the software development. We processed the files to extract the Fusion sessions and used the file sizes to divide them into 20 approximately equal log files. Each log contains about 275 Fusion sessions with a total of 5,497 fusion sessions in all 20 logs. Each Fusion session represents one instance where the Semantic Integration module fused inputs from multiple modalities, Dialog History, and Real World Context to generate a potential Task for the Dialog Management module to validate. Each sample contains the actual

confidence scores provided by the modality and context inputs (i.e., Speech, Pointing, Field of Vision, Real World Context, and Dialog History) for a specific phrase and whether the phrase was the Best choice or Not. Table 5.1 summarizes the log files.

Log File	Original Log	Lesson Plans	Fusion Sessions
Log 1a	communicator before 9-20.log	LP9, LP10	252
Log 1b	communicator before 9-20.log	LP9, LP10	253
Log 1c	communicator before 9-20.log	LP9, LP10	252
Log 2a	communicator before 8-27-07 1300.log	LP5, LP6	205
Log 2b	communicator before 8-27-07 1300.log	LP5, LP6	205
Log 2c	communicator before 8-27-07 1300.log	LP5, LP6	205
Log 3	communicator before 8-9-07 1510.log	LP1	291
Log 4	communicator before 8-15-07 1115.log	LP2	421
Log 5	communicator before 8-17-07 1645.log	LP2, LP3, LP4	582
Log 6a	communicator before 9-10 1630.log	LP6, LP7	265
Log 6b	communicator before 9-10 1630.log	LP6, LP7	264
Log 6c	communicator before 9-10 1630.log	LP6, LP7	265
Log 6d	communicator before 9-10 1630.log	LP6, LP7	265
Log 6e	communicator before 9-10 1630.log	LP6, LP7	264
Log 6f	communicator before 9-10 1630.log	LP6, LP7	265
Log 7a	communicator log before 9-12.log	LP8, LP9	249
Log 7b	communicator log before 9-12.log	LP8, LP9	248
Log 7c	communicator log before 9-12.log	LP8, LP9	249
Log 7d	communicator log before 9-12.log	LP8, LP9	248
Log 7e	communicator log before 9-12.log	LP8, LP9	249
		Average	275
		Total	5497

Table 5.1: Log Files Used To Determine Fusion Weights.

5.2 Calculation of Fusion Weights with No Errors

Next, we determined the 20 sets of weights using the brute-force method, for each of the 20 logs with no-errors as shown in Table 5.2.

		No Errors																			
Slot	**Weight**	**Log 1a**	**Log 1b**	**Log 1c**	**Log 2a**	**Log 2b**	**Log 2c**	**Log 3**	**Log 4**	**Log 5**	**Log 6a**	**Log 6b**	**Log 6c**	**Log 6d**	**Log 6e**	**Log 6f**	**Log 7a**	**Log 7b**	**Log 7c**	**Log 7d**	**Log 7e**
action	realWorldW	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20	0.20	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20
	historyW	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20
	speechW	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60
	Fusion error rate	6%	33%	18%	6%	8%	9%	11%	6%	4%	9%	11%	5%	4%	3%	5%	16%	13%	11%	11%	8%
object	realWorldW	-0.20	-0.20	-0.26	-0.20	-0.20	-0.20	0.14	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20
	historyW	0.60	0.60	0.61	0.60	0.60	0.60	0.33	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60
	fieldOfVisionW	-0.20	-0.20	-0.26	-1.00	0.20	-0.20	0.04	-0.20	0.20	-0.60	-1.00	-1.00	-1.00	-1.00	-1.00	-0.60	-1.00	-0.20	-0.20	-0.20
	pointingW	0.20	-0.20	0.14	0.20	0.20	0.20	0.02	0.20	0.20	0.20	0.20	-1.00	0.20	0.20	0.20	0.20	0.20	0.20	0.20	0.20
	speechW	1.00	1.00	0.96	1.00	1.00	1.00	0.66	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	Fusion error rate	17%	20%	27%	31%	23%	21%	13%	17%	10%	22%	22%	23%	25%	23%	24%	20%	21%	23%	18%	16%
tools	realWorldW	1.00	1.00	0.66	-0.20	-0.20	-0.20	0.28	-0.20	-0.20	-0.20	1.00	0.60	1.00	1.00	1.00	-0.20	1.00	0.36	-0.20	1.00
	historyW	0.60	0.60	0.36	0.60	0.20	0.20	0.12	0.20	0.60	0.20	0.60	0.20	0.60	0.60	0.60	0.60	0.60	0.12	0.60	0.60
	fieldOfVisionW	-1.00	-1.00	0.57	-0.60	0.20	0.20	0.76	0.20	-1.00	0.20	1.00	-1.00	-0.60	-1.00	-0.60	-0.20	-0.60	0.60	-0.20	-1.00
	pointingW	1.00	1.00	0.60	0.60	0.20	0.20	0.20	0.20	0.60	0.20	1.00	0.60	0.20	0.60	1.00	0.60	1.00	0.20	1.00	1.00
	speechW	1.00	1.00	0.44	1.00	0.60	0.60	0.36	0.60	1.00	0.60	1.00	0.60	1.00	1.00	1.00	1.00	1.00	0.20	1.00	1.00
	Fusion error rate	41%	43%	38%	22%	31%	43%	33%	65%	29%	44%	31%	24%	25%	21%	32%	18%	15%	19%	20%	29%
condition	realWorldW	0.20	0.20	-0.60	0.20	0.20	-0.60	-1.00	-0.20	-0.20	0.20	0.20	-0.60	-0.60	0.20	0.20	-0.60	-0.60	0.20	0.20	0.20
	historyW	0.20	0.60	1.00	0.60	0.60	1.00	0.60	0.20	0.20	0.60	0.60	1.00	1.00	0.60	0.60	1.00	1.00	0.60	0.60	0.20
	speechW	0.60	0.20	0.20	0.20	0.20	-0.60	1.00	0.60	0.60	0.20	0.20	-0.20	-0.20	0.20	-0.20	0.20	0.20	-0.20	-0.20	-0.20
	Fusion error rate	0%	2%	0%	1%	1%	0%	13%	3%	1%	2%	1%	1%	0%	0%	0%	0%	0%	0%	0%	0%

Table 5.2: Fusion Weights (No Errors).

The brute-force algorithm tries all combination of weights, measuring the FER each time, and picking the set of weights with the lowest FER. For faster convergence, we start by using a large delta on the first pass and then using a smaller delta around the first pass weights on the second pass, and so forth.

5.3 Calculation of Fusion Weights with Errors

Then, we determined the 20 sets of weights for each of the 20 logs with errors as shown in Table 5.3.

		Errors																			
Slot	**Weight**	**Log 1a**	**Log 1b**	**Log 1c**	**Log 2a**	**Log 2b**	**Log 2c**	**Log 3**	**Log 4**	**Log 5**	**Log 6a**	**Log 6b**	**Log 6c**	**Log 6d**	**Log 6e**	**Log 6f**	**Log 7a**	**Log 7b**	**Log 7c**	**Log 7d**	**Log 7e**
action	realWorldW	0.20	-0.20	-0.20	-0.20	0.20	-0.20	0.79	0.20	0.52	-0.20	0.20	0.20	-0.50	-0.20	0.20	0.20	0.20	0.20	0.20	0.20
	historyW	0.60	0.20	0.20	0.20	0.60	0.20	0.02	0.60	0.28	0.20	0.60	0.20	0.61	0.20	0.60	0.60	0.60	0.60	0.60	0.60
	speechW	1.00	0.60	0.60	0.60	1.00	0.60	0.39	1.00	0.36	0.60	1.00	0.60	0.62	0.60	1.00	1.00	1.00	1.00	1.00	1.00
	Fusion error rate	35%	55%	47%	39%	37%	33%	23%	24%	22%	42%	38%	29%	29%	27%	34%	38%	37%	36%	40%	30%
object	realWorldW	0.04	-0.04	0.70	0.04	0.28	0.60	0.84	0.10	0.84	0.12	-0.04	0.02	0.92	0.44	0.20	0.60	-0.04	0.92	0.70	-0.01
	historyW	0.75	1.00	0.84	0.60	0.76	-0.12	0.68	0.09	0.12	0.84	1.00	0.84	0.12	1.03	0.12	0.04	0.90	0.12	0.73	1.07
	fieldOfVisionW	-0.40	-1.40	-0.50	-1.40	-0.82	-1.40	-0.12	-0.26	-0.60	-0.63	-0.60	-1.05	-0.60	-1.40	-1.40	-0.36	-0.44	-0.76	-0.49	-0.61
	pointingW	0.00	0.04	-0.07	-0.04	0.01	0.04	0.04	0.10	0.60	-0.06	0.04	-1.56	-0.12	0.01	-0.12	0.36	0.01	-0.12	-0.05	0.00
	speechW	0.81	1.08	0.92	0.68	0.86	0.60	0.76	1.56	0.84	0.92	1.08	0.92	0.84	1.11	1.08	0.60	0.98	1.08	0.83	1.18
	Fusion error rate	32%	36%	47%	43%	37%	39%	29%	25%	20%	39%	36%	46%	40%	37%	41%	40%	33%	42%	34%	28%
tools	realWorldW	1.24	0.76	1.21	0.92	0.92	0.76	1.05	0.52	1.34	0.92	0.92	1.08	0.50	0.76	1.24	1.24	1.24	1.40	0.82	0.76
	historyW	0.36	0.49	0.21	0.84	0.52	0.68	0.25	-0.12	0.36	0.84	0.68	0.54	0.42	0.52	0.60	0.28	0.60	0.68	0.44	0.68
	fieldOfVisionW	-1.56	-0.50	0.30	-1.40	0.76	-0.60	0.57	-0.52	-1.56	-0.20	0.60	-0.92	-0.55	-0.52	-1.08	0.04	0.12	-0.28	0.52	-1.40
	pointingW	0.02	-0.76	0.20	0.68	0.12	0.04	0.03	0.28	-0.01	0.04	0.52	0.52	0.09	0.28	0.52	0.52	0.28	0.04	0.23	0.04
	speechW	0.44	0.57	0.26	1.08	0.76	0.76	0.37	0.44	0.60	0.92	0.76	0.55	0.50	0.60	0.76	0.60	0.68	0.76	0.50	0.76
	Fusion error rate	52%	51%	50%	34%	48%	48%	39%	72%	35%	51%	41%	32%	34%	28%	48%	28%	27%	28%	35%	39%
condition	realWorldW	0.76	-0.60	-0.60	1.08	0.92	1.00	-1.40	0.20	0.20	0.66	0.68	0.60	0.84	1.00	0.60	1.00	1.00	1.00	1.00	1.00
	historyW	0.12	1.00	1.00	0.28	0.12	0.20	0.84	0.60	0.60	0.25	0.12	0.20	0.12	0.20	0.20	0.20	0.20	0.20	0.20	0.20
	speechW	0.28	1.00	1.00	0.12	0.12	0.20	1.08	1.00	1.00	0.02	0.12	0.20	0.12	0.20	0.20	0.20	0.20	0.20	0.20	0.20
	Fusion error rate	16%	39%	31%	18%	20%	25%	39%	25%	30%	23%	19%	20%	20%	19%	19%	24%	26%	26%	21%	15%

Table 5.3: Fusion Weights (Errors).

The experiments recorded in the log file were performed in an error free environment to test the software, but we processed the samples and added errors that are consistent with the Speech Recognition, Pointing, and Field of Vision errors.

5.4 K-Means Clustering of Fusion Weights

Next, we used the K-Means clustering algorithm to cluster both the no errors and errors weights. We varied *k* from 2 to 20 clusters.

The K-Means algorithm requires the determination of the initial centroids for each cluster. We used an evenly distributed grid between the minimum and maximum of each weight for these reasons:

- The results would be repeatable.
- A null cluster would indicate an area of empty space between clusters, which would mean the points were truly clustered and not just evenly distributed throughout the feature space.

Another popular way to do it is randomly [2].

Table 5.4 and Table 5.5 show the number of sets of weights in each cluster.

	Cluster No.																		
k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
2	19	1																		
3	19		1																	
4	19			1																
5	19				1															
6	19					1														
7	19						1													
8	19							1												
9	19								1											
10	19									1										
11	19										1									
12	19											1								
13	19												1							
14	19													1						
15	19														1					
16	19															1				
17	19																1			
18	19																	1		
19	19																		1	
20	19																			1

action slot (no errors)

	Cluster No.																		
k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
2	1	19																		
3	1	8	11																	
4	1		8	11																
5	1		6	3	10															
6	1			6	4	9														
7	1			6	3	7	3													
8	1				6	2	9	2												
9	1				6		2	9	2											
10	1					6	2	1	8	2										
11	1					6		2	1	8	2									
12	1						6		2	1	8	2								
13	1						6		2		9		2							
14	1							6		2	1	8		2						
15	1							6			2	1	8		2					
16	1								6		2		1	8		2				
17	1								6			2		1	8		2			
18	1									6			2	1	8			2		
19	1									6			2		1	8			2	
20	1										6			2		1	8			2

object slot (no errors)

	Cluster No.																		
k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
2	12	8																		
3	8	8	4																	
4	4	8	7	1																
5	3	6	7	3	1															
6	4	8	4		3	1														
7	2	1	2	7	6	1	1													
8		3	7	6		3		1												
9		1	4		7	6	1		1											
10		4	1	4	7		2	1		1										
11		2	1		2	7	6	1			1									
12		1	1	3	4	7		2	1			1								
13		1	1	3	7				6	1			1							
14			2	1		2	1	6	6	1				1						
15			1	1	3	4	7			2	1				1					
16			1	1	3	7				6		1				1				
17			1	1	3		4	7			2	1					1			
18			1	1	3		5			2	2	4	1					1		
19			1		5	3			1		2	6		1					1	
20				1	1	3		4	5		2	1	1		1					1

tools slot (no errors)

	Cluster No.																		
k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
2	7	13																		
3	3	4	13																	
4	1	5	10	4																
5	3	4	4	9																
6		1	5	3	11															
7		1	5	4	6	4														
8		1	5	4		6	4													
9		1		2	3	3	11													
10			1	5	4		6	4												
11			1		2	3	3	10	1											
12			1		5	4		3	7											
13			1		5	4			6	4										
14				1		5	4		3	6	1									
15				1		5	4			6		4								
16				1		5		4			6	4								
17				1			5	4			3	6	1							
18					1		5		4			6		4						
19					1			5	4			3	6	1						
20					1			5	4				3	6	1					

condition slot (no errors)

Table 5.4: K-Means Clustering of Fusion Weights (No Errors).

k	Cluster No. 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
2	8	12																		
3	7	3	10																	
4	7	3	10																	
5	1	6	3	10																
6	1	6	1	2	10															
7		7		1	10	2														
8		7		1	2	10														
9		1	6		1	2	10													
10		1	6		1		2	10												
11			7			1	2	10												
12			7			1		2	10											
13			1	6			1	2		10										
14			1	6			1		2	10										
15			1	6				1		2	10									
16				1	6			1		2		10								
17				1	6				1		1	10	1							
18				1	6				1			2	10							
19				1	6					1		2		10						
20				1		6				1			1	10	1					

action slot (errors)

k	Cluster No. 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
2	6	14																		
3	1	5	14																	
4	1	3	9	7																
5	1		5	14																
6	1		3	8	8															
7		1	2	3	9	5														
8		1		3	2	9	5													
9		1		3	2	6	8													
10		1			3	5	8	3												
11		1			3		2	9	5											
12			1			3	2	3	7	4										
13			1			3		3	6	7										
14			1			2	1	2	4	6	4									
15			1				3		2	8	3	3								
16			1				2	1	2	2	6	3	3							
17				1				3		2	3	7	4							
18				1				2	1	2	2	6	3	3						
19				1					3		2	2	6	3	3					
20				1					3			2	6	5	3					

object slot (errors)

k	Cluster No. 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
2	11	9																		
3	4	9	7																	
4	6	5	4	5																
5	2	5	4	5	4															
6		3	6	4	4	3														
7		3	3	5	3	4	2													
8		2	2	3	5	4	3	1												
9		2	2	5	2	2	5	1	1											
10			3	1	3	4	3	4	2											
11			2	1	4	4	2	2	3	2										
12			2	1	2	3	3	2	3	2	2									
13			2	2	1	2	4		3	4	1	1								
14				3	1	1	3	3	2	2	3	1	1							
15				2	2	1	2	4		2	3	2	1	1						
16				2	1	1	2	3	2		3	2	2	1	1					
17				2	1	1	2	1	4		2	2	2		3					
18					2	1	1	2	3	2		2	2	3		2				
19					2	1	1	2	1	4		2	2	2	1	1	1			
20					2	2	1		1	2	3		2	2	2	1	1	1		

tools slot (errors)

k	Cluster No. 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
2	3	17																		
3	1	4	15																	
4	1	2	2	15																
5	1		2	2	15															
6		1	4		6	9														
7		1		2	2	15														
8		1		2		2	15													
9		1			2	5	2	10												
10		1			2		2	6	9											
11			1			2		2	6	9										
12			1			2		5	2	10										
13			1				2		2	6	9									
14			1				2			2	6	9								
15			1					2		5	2	9	1							
16				1				2			2	6	8	1						
17				1				2				2	6	9						
18				1					2			5	2	2	8					
19				1					2				2	6	8	1				
20					1					2				2	6	8	1			

condition slot (errors)

Table 5.5: K-Means Clustering of Fusion Weights (Errors).

5.5 Best Fusion Weights (Reconstruction Error)

We used two different methods to determine which *k* to use. This section discusses using *reconstruction error* and the next section discusses using *peakedness*.

Alpaydin suggested plotting the *reconstruction error* as a function of *k* and looking for the "elbow" of the curve [2].

Alpaydin defined *reconstruction error* as the sum of the mean-squared-error between all 20 samples and the centroid of the cluster they were assigned to, or [2]:

$$E(\{\boldsymbol{m_i}\}^{k}{}_{i=1}|X) = \Sigma_t \Sigma_i \; b^t{}_i \left\| \boldsymbol{x^t} - \boldsymbol{m_i} \right\|^2$$

where

$$b^t_i = \begin{cases} 1, \text{ if } \left\| \boldsymbol{x}^t - \boldsymbol{m}_i \right\| = \min_j \left\| \boldsymbol{x}^t - \boldsymbol{m}_i \right\| \\ 0, \text{ otherwise} \end{cases}$$

As shown in Figure 5.1, the reconstruction error for the "action" slot with no errors was zero for all k, so we used $k = 2$. Cluster No. 0 in Table 5.4 is the largest cluster for $k = 2$, which was used.

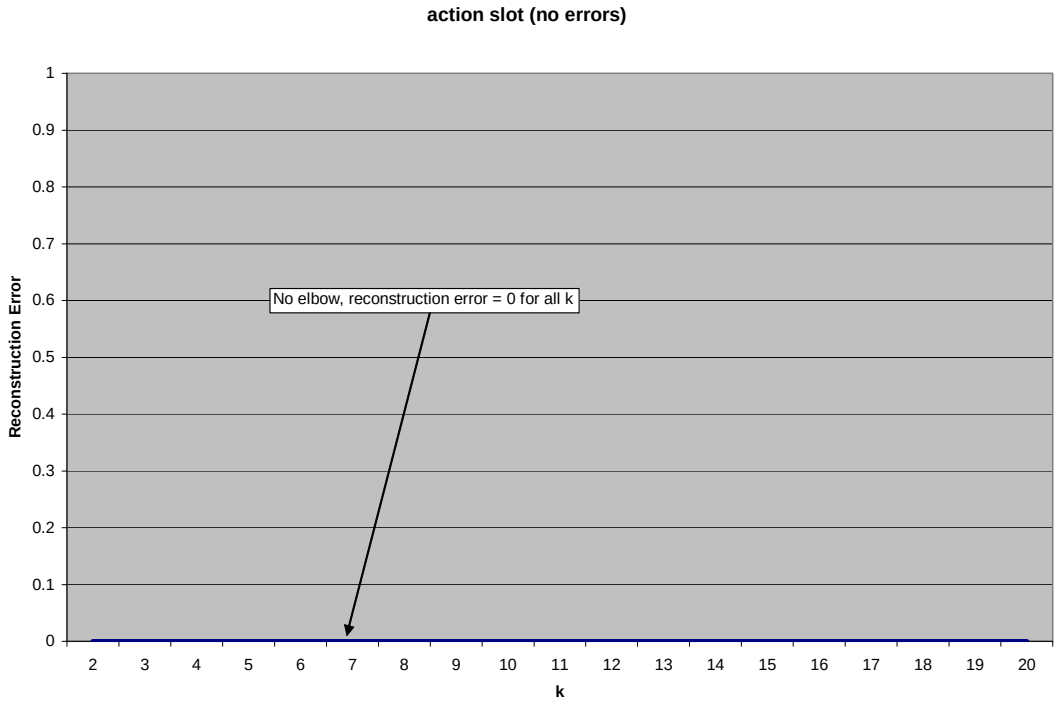

Figure 5.1: Reconstruction Error Curve for Action Slot (No Errors).

We calculated the reconstruction error for the other slots and found the "elbow" as shown in the Figure 5.2 through Figure 5.8.

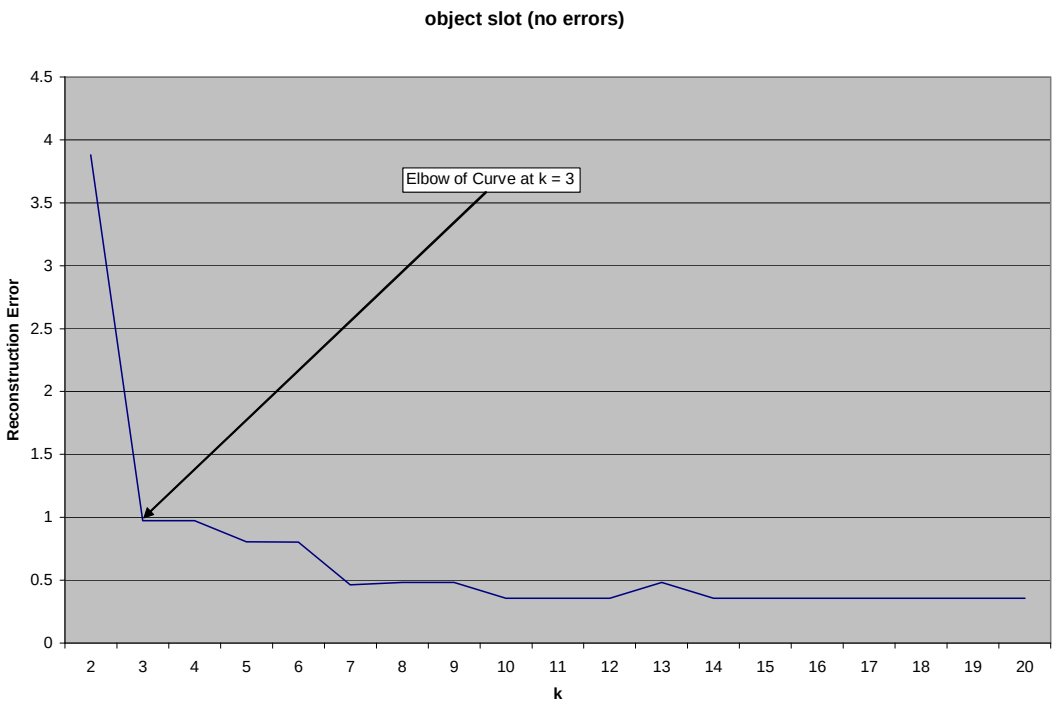

Figure 5.2: Reconstruction Error Curve for Object Slot (No Errors).

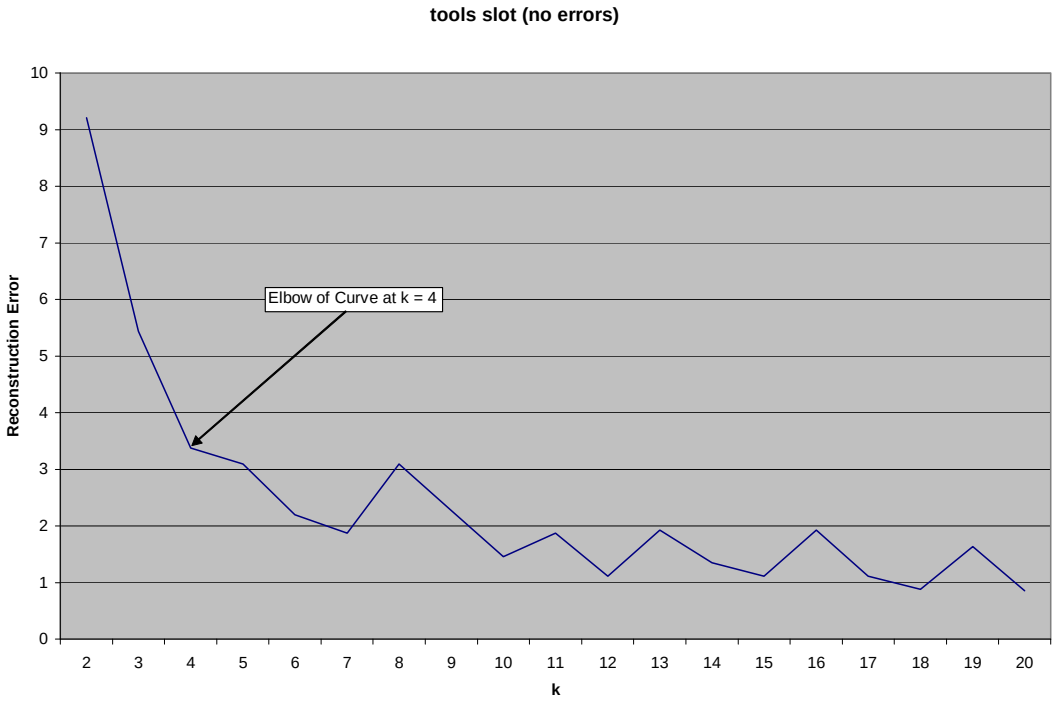

Figure 5.3: Reconstruction Error Curve for Tools Slot (No Errors).

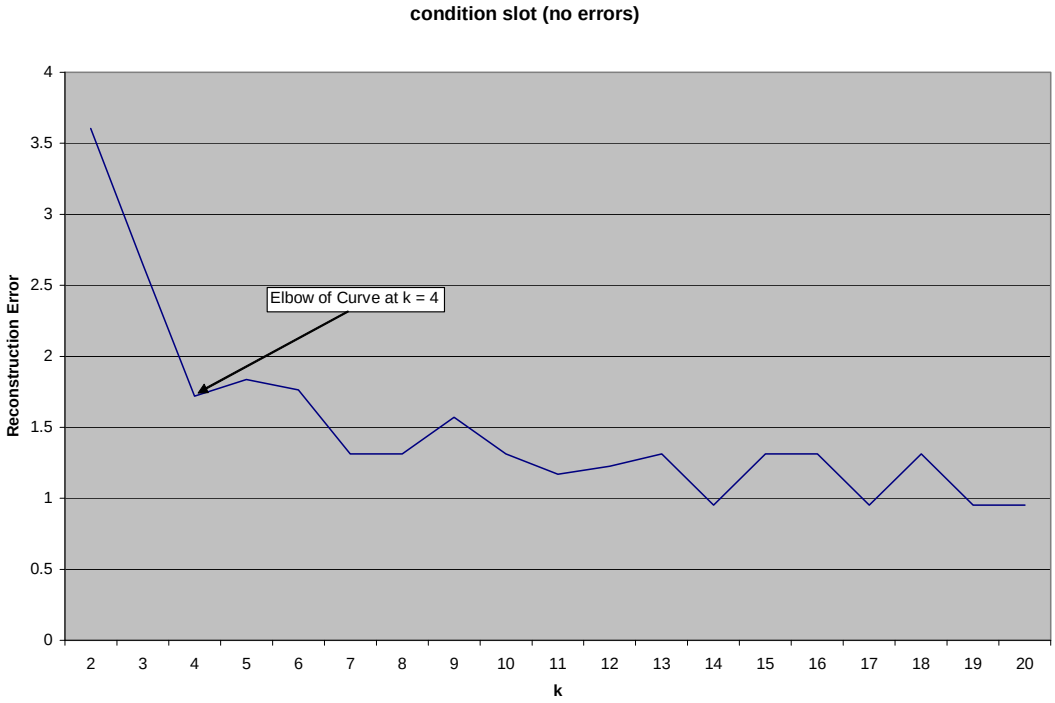

Figure 5.4: Reconstruction Error Curve for Condition Slot (No Errors).

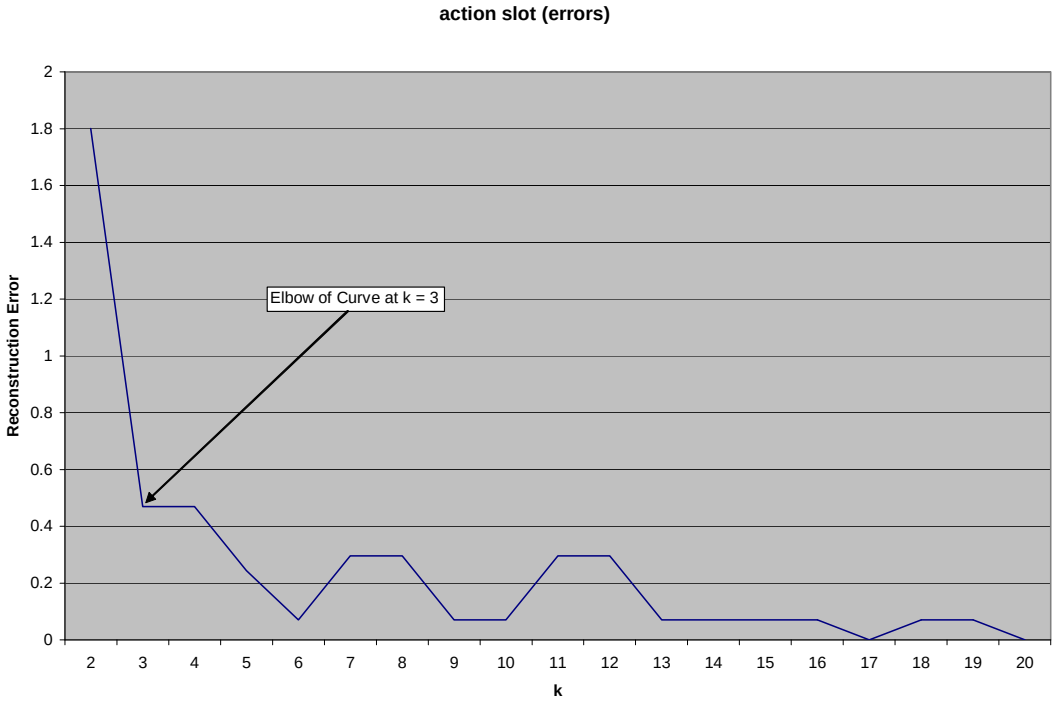

Figure 5.5: Reconstruction Error Curve for Action Slot (Errors).

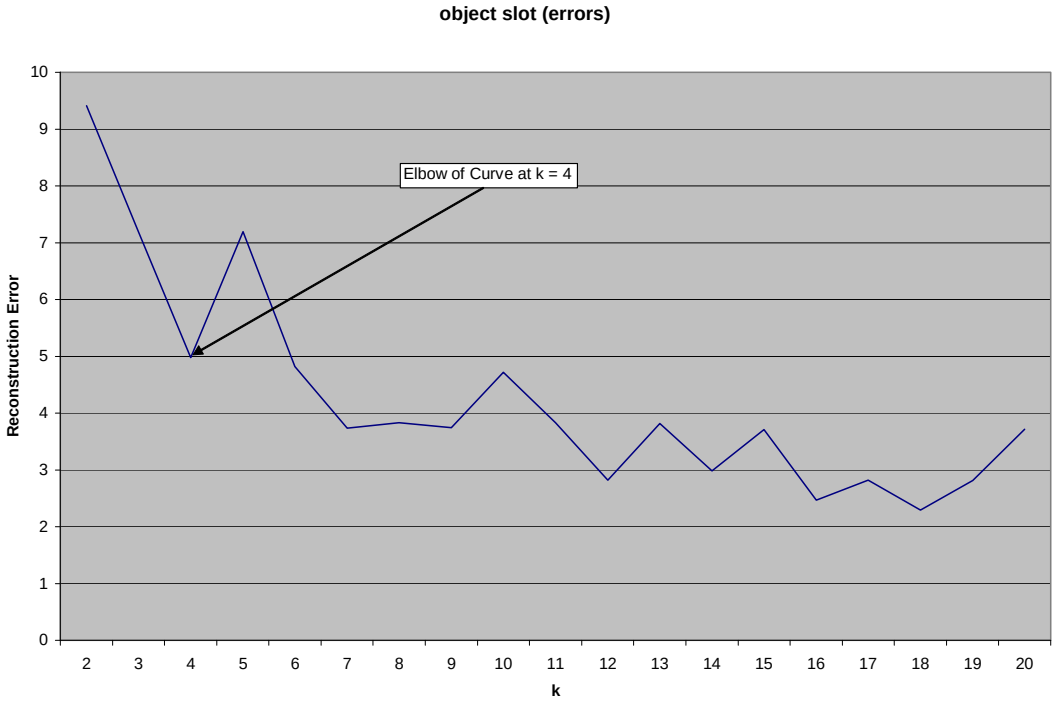

Figure 5.6: Reconstruction Error Curve for Object Slot (Errors).

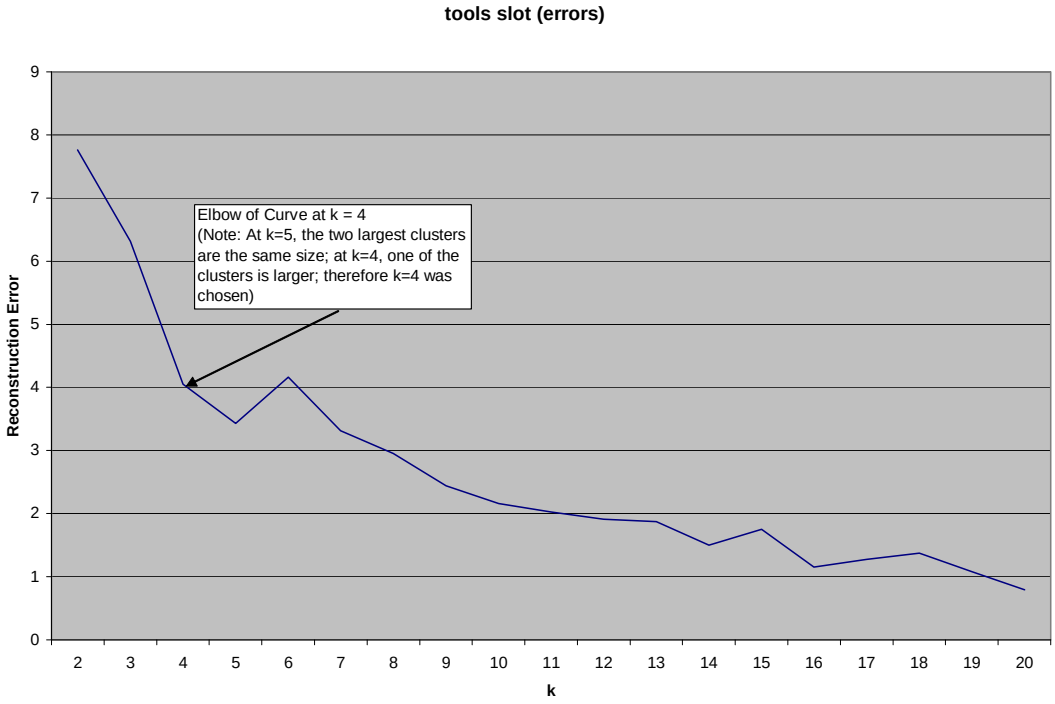

Figure 5.7: Reconstruction Error Curve for Tools Slot (Error).

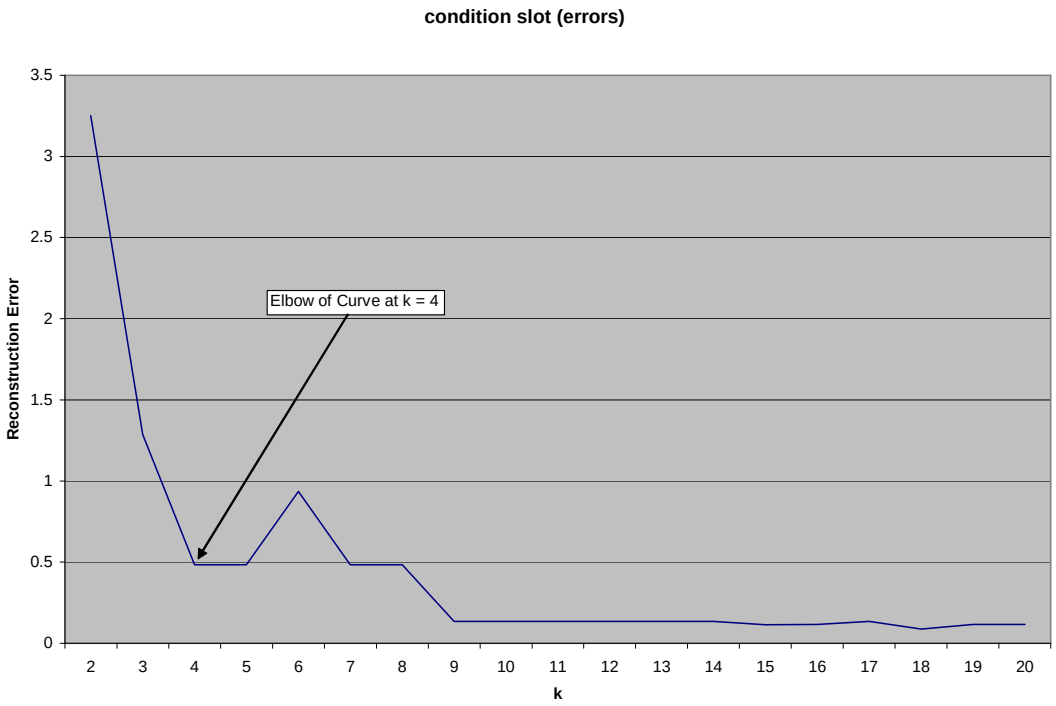

Figure 5.8: Reconstruction Error Curve for Condition Slot (Errors).

We used the largest cluster in Table 5.4 and Table 5.5 for the *k* determined by *reconstruction error* as shown in Table 5.6.

Slot	No Errors / Errors	k	Cluster
action	no errors	2	0
object	no errors	3	2
tools	no errors	4	1
condition	no errors	4	2
action	errors	3	2
object	errors	4	2
tools	errors	4	0
condition	errors	4	3

Table 5.6: Largest Cluster Determined By Reconstruction Error.

5.6 Best Fusion Weights (Peakedness)

The other method for determining *k*, which we propose, is to find the one where the largest cluster is the one with the highest peak in comparison to the other clusters.

For example, if we plot the number of weight vectors in each cluster for the condition slot (no errors) with $k = 11$, we get the plot shown Figure 5.9.

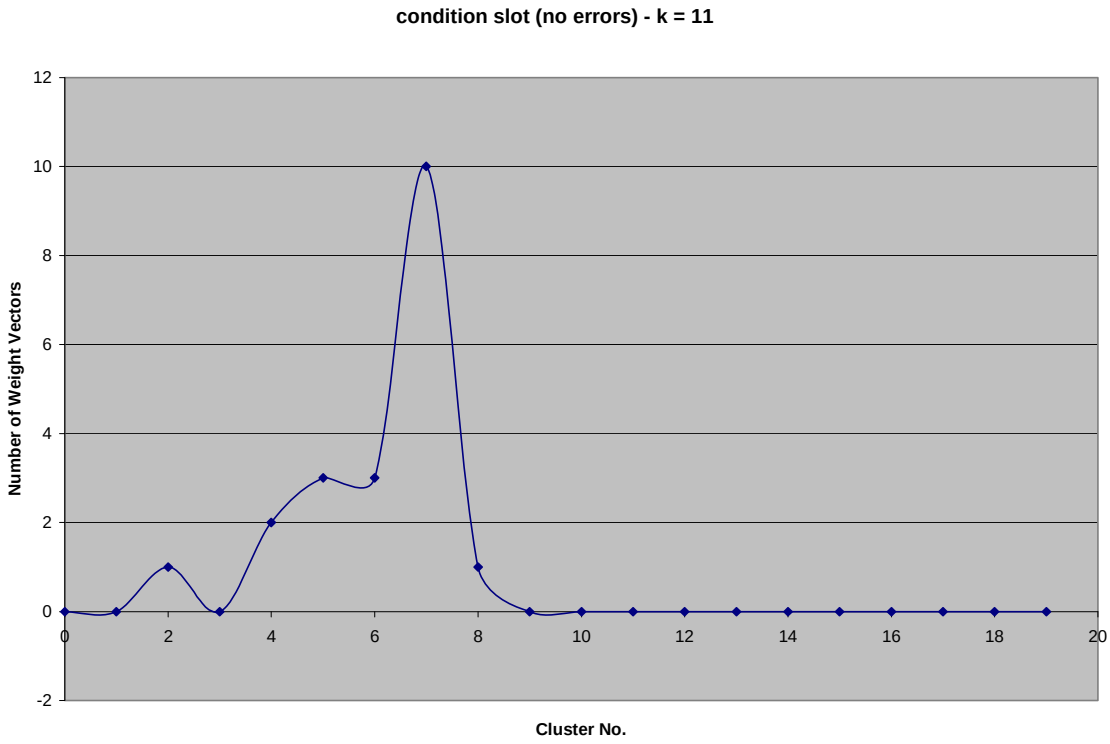

Figure 5.9: Example of Peakedness.

There is a clear peak at Cluster No. 7 with 10 weight vectors in it.

We can find the *k* with the highest peak by calculating the "peakedness" for each *k* using the formula below, and then using the one with the highest "peakedness" value:

peakedness = (*vectors/cluster*) / AVERAGE(*vectors/cluster*)

In the example above,

peakedness = 10 / 3.33 = 3.00

As shown in Table 5.7, we used the largest cluster in Table 5.4 and Table 5.5 for the *k* with the highest peak. For the case when the *peakedness* for two *k* were the same, the smaller *k* was used.

Slot	No Errors / Errors	k	Cluster
action	no errors	2	0
object	no errors	10	8
tools	no errors	12	5
condition	no errors	11	7
action	errors	17	11
object	errors	15	9
tools	errors	19	9
condition	errors	4	3

Table 5.7: Largest Cluster Determined By Peakedness.

5.7 Choosing the Best Fusion Weights

Table 5.8 shows the sets of "best" weights for each slot using both methods of determining *k*:

- Centroid (of the largest cluster)
- Closest to Centroid (set closest, in a Euclidean sense, to the centroid of the largest cluster)
- Frequency (most frequently occurring set in the largest cluster; "n/a" means all the sets in the cluster were different)

		Reconstruction Error			Peakedness		
Slot	**Weight**	**Centroid**	**Closest**	**Frequent**	**Centroid**	**Closest**	**Frequent**
action	realWorldW	-0.20	-0.20	-0.20	-0.20	-0.20	-0.20
	historyW	0.20	0.20	0.20	0.20	0.20	0.20
	speechW	0.60	0.60	0.60	0.60	0.60	0.60
object	realWorldW	-0.18	-0.20	-0.20	-0.17	-0.20	-0.20
	historyW	0.58	0.60	0.60	0.57	0.60	0.60
	fieldOfVisionW	-0.11	-0.20	-0.20	-0.18	-0.20	-0.20
	pointingW	0.14	0.20	0.20	0.17	0.20	0.20
	speechW	0.97	1.00	1.00	0.95	1.00	1.00
tools	realWorldW	0.95	1.00	1.00	1.00	1.00	1.00
	historyW	0.55	0.60	0.60	0.60	0.60	0.60
	fieldOfVisionW	-0.85	-1.00	-1.00	-0.83	-1.00	-1.00
	pointingW	0.80	1.00	1.00	0.83	1.00	1.00
	speechW	0.95	1.00	1.00	1.00	1.00	1.00
condition	realWorldW	0.20	0.20	0.20	0.20	0.20	0.20
	historyW	0.56	0.60	0.60	0.56	0.60	0.60
	speechW	0.04	0.20	0.20	0.04	0.20	0.20

no errors

		Reconstruction Error			Peakedness		
Slot	**Weight**	**Centroid**	**Closest**	**Frequent**	**Centroid**	**Closest**	**Frequent**
action	realWorldW	0.20	0.20	0.20	0.20	0.20	0.20
	historyW	0.60	0.60	0.60	0.60	0.60	0.60
	speechW	1.00	1.00	1.00	1.00	1.00	1.00
object	realWorldW	0.09	0.12	n/a	0.09	0.12	n/a
	historyW	0.83	0.84	n/a	0.92	0.84	n/a
	fieldOfVisionW	-0.73	-0.63	n/a	-0.79	-0.63	n/a
	pointingW	0.02	-0.06	n/a	0.01	-0.06	n/a
	speechW	1.06	0.92	n/a	1.00	0.92	n/a
tools	realWorldW	1.10	1.24	n/a	0.96	1.08	n/a
	historyW	0.56	0.60	n/a	0.58	0.54	n/a
	fieldOfVisionW	-1.32	-1.08	n/a	-0.78	-0.92	n/a
	pointingW	0.30	0.52	n/a	0.34	0.52	n/a
	speechW	0.70	0.76	n/a	0.67	0.55	n/a
condition	realWorldW	0.88	0.84	1.00	0.88	0.84	1.00
	historyW	0.19	0.12	0.20	0.19	0.12	0.20
	speechW	0.17	0.12	0.20	0.17	0.12	0.20

errors

Table 5.8: Sets of Best Weights.

As shown, in some cases, the different methods identified the same set of weights.

Next, we calculated the Fusion Error Rate (FER) for the 20 logs with errors using the sets of "best" weights as determined. The results are shown in Table 5.9 and Table 5.10.

Weights	N.*.*	E.*.*
realWorldW	-0.20	0.20
historyW	0.20	0.60
speechW	0.60	1.00
Log 1a	37%	35%
Log 1b	55%	57%
Log 1c	47%	50%
Log 2a	39%	40%
Log 2b	38%	37%
Log 2c	33%	33%
Log 3	33%	27%
Log 4	27%	24%
Log 5	23%	23%
Log 6a	42%	45%
Log 6b	41%	38%
Log 6c	31%	29%
Log 6d	30%	30%
Log 6e	27%	28%
Log 6f	36%	34%
Log 7a	40%	38%
Log 7b	38%	37%
Log 7c	37%	36%
Log 7d	42%	40%
Log 7e	31%	30%
Average	36%	35%

FER (action slot)

Weights	N.R.Ce	N.*.Cl/F	N.P.Ce	E.R.Ce	E.*.Cl	E.P.Ce
realWorldW	-0.18	-0.20	-0.17	0.09	0.12	0.09
historyW	0.58	0.60	0.57	0.83	0.84	0.92
fieldOfVisionW	-0.11	-0.20	-0.18	-0.73	-0.63	-0.79
pointingW	0.14	0.20	0.17	0.02	-0.06	0.01
speechW	0.97	1.00	0.95	1.06	0.92	1.00
Log 1a	46%	47%	47%	41%	35%	36%
Log 1b	43%	44%	44%	41%	35%	35%
Log 1c	54%	56%	54%	53%	48%	49%
Log 2a	49%	49%	49%	48%	43%	43%
Log 2b	53%	56%	56%	45%	38%	38%
Log 2c	46%	46%	46%	41%	37%	36%
Log 3	36%	36%	36%	36%	36%	37%
Log 4	30%	30%	30%	28%	31%	32%
Log 5	23%	24%	23%	22%	25%	26%
Log 6a	55%	57%	56%	48%	39%	41%
Log 6b	45%	46%	45%	42%	36%	36%
Log 6c	51%	51%	51%	48%	47%	47%
Log 6d	47%	47%	47%	42%	39%	40%
Log 6e	45%	45%	45%	40%	37%	37%
Log 6f	52%	54%	46%	44%	39%	39%
Log 7a	45%	45%	45%	42%	37%	37%
Log 7b	43%	43%	43%	41%	34%	34%
Log 7c	48%	48%	48%	44%	41%	41%
Log 7d	46%	49%	46%	44%	36%	36%
Log 7e	35%	35%	35%	34%	30%	29%
Average	44.5%	45.3%	44.6%	41.2%	37.3%	37.4%

FER (object slot)

Table 5.9: Fusion Error Rate (FER) for Action and Object Slots.

Weights	N.R.Ce	N.*.Cl/F	N.P.Ce	E.R.Ce	E.R.Cl	E.P.Ce	E.P.Cl
realWorldW	0.95	1.00	1.00	1.10	1.24	0.96	1.08
historyW	0.55	0.60	0.60	0.56	0.60	0.58	0.54
fieldOfVisionW	-0.85	-1.00	-0.83	-1.32	-1.08	-0.78	-0.92
pointingW	0.80	1.00	0.83	0.30	0.52	0.34	0.52
speechW	0.95	1.00	1.00	0.70	0.76	0.67	0.55
Log 1a	61%	63%	60%	57%	56%	56%	54%
Log 1b	53%	55%	53%	53%	55%	53%	55%
Log 1c	58%	61%	58%	59%	57%	57%	57%
Log 2a	37%	38%	37%	36%	35%	36%	37%
Log 2b	55%	57%	55%	54%	53%	53%	52%
Log 2c	54%	54%	54%	53%	52%	52%	52%
Log 3	47%	48%	47%	46%	45%	46%	45%
Log 4	73%	74%	73%	74%	73%	74%	73%
Log 5	38%	38%	38%	36%	36%	36%	36%
Log 6a	61%	64%	60%	55%	57%	55%	59%
Log 6b	45%	47%	45%	45%	44%	44%	46%
Log 6c	38%	40%	39%	38%	36%	34%	32%
Log 6d	41%	42%	41%	38%	38%	36%	36%
Log 6e	34%	35%	34%	32%	31%	29%	31%
Log 6f	53%	58%	53%	51%	48%	52%	50%
Log 7a	32%	33%	32%	32%	31%	31%	30%
Log 7b	32%	35%	32%	31%	29%	29%	28%
Log 7c	35%	38%	36%	32%	32%	31%	31%
Log 7d	40%	42%	41%	38%	37%	37%	39%
Log 7e	43%	43%	43%	40%	40%	39%	38%
Average	46.5%	48.3%	46.6%	45.0%	44.2%	44.1%	44.0%

FER (tools slot)

Weights	N.*.Ce	N.*.Cl/F	E.*.Ce	E.*.Cl	E.*.F
realWorldW	0.20	0.20	0.88	0.84	1.00
historyW	0.56	0.60	0.19	0.12	0.20
speechW	0.04	0.20	0.17	0.12	0.20
Log 1a	20%	22%	16%	16%	16%
Log 1b	45%	43%	53%	53%	53%
Log 1c	39%	38%	49%	49%	49%
Log 2a	23%	25%	18%	18%	18%
Log 2b	26%	28%	21%	20%	21%
Log 2c	31%	32%	25%	25%	25%
Log 3	49%	48%	62%	63%	62%
Log 4	26%	25%	29%	29%	29%
Log 5	30%	30%	34%	34%	34%
Log 6a	26%	26%	24%	24%	24%
Log 6b	24%	27%	19%	19%	19%
Log 6c	23%	23%	20%	20%	20%
Log 6d	23%	24%	20%	20%	20%
Log 6e	24%	25%	19%	19%	19%
Log 6f	23%	25%	19%	19%	19%
Log 7a	29%	30%	24%	24%	24%
Log 7b	27%	29%	26%	26%	26%
Log 7c	30%	31%	26%	26%	26%
Log 7d	29%	29%	21%	21%	21%
Log 7e	20%	21%	15%	15%	15%
Average	28.36%	29.14%	27.05%	27.04%	27.08%

FER (condition slot)

Table 5.10: Fusion Error Rate (FER) for Tools and Condition Slots.

The weights are denoted using the following nomenclature:

X.Y.Z

where:

X = N, if the weights were determined using the "no errors" data

E, if the weights were determined using the "errors" data

Y = R, if *k* was determined using *reconstruction error*

P, if *k* was determined using *peakedness*

*, if the weights were the same for both methods of determining *k*

Z = Ce, if the weights were the centroid of the largest cluster

Cl, if the weights were the ones closest to the centroid of the largest cluster

F, if the weights were the most frequent weights in the largest cluster

Cl/F, if the weights closest to the centroid were also the most frequent

*, if the weights closest to the centroid were the same as the centroid and also the most frequent

The FER was calculated as follows:

$f = e / t$

where:

f is the Fusion Error Rate (FER)

e is the number of incorrect Fusion Sessions

t is the total number of Fusion Sessions in the log (Table 5.1)

The number of incorrect Fusion Sessions, *e*, was determined by comparing the slot value determined using the weights under consideration with the correct slot value. For example, if the number of slots determined incorrectly for Log 1a was 93, the FER would be calculated as follows:

$$f = e / t$$

$$f = 93 / 252$$

$$f = 37\%$$

It was determined that one set of weights could not produce the lowest FER for all logs, as illustrated in Figure 5.10 through Figure 5.13.

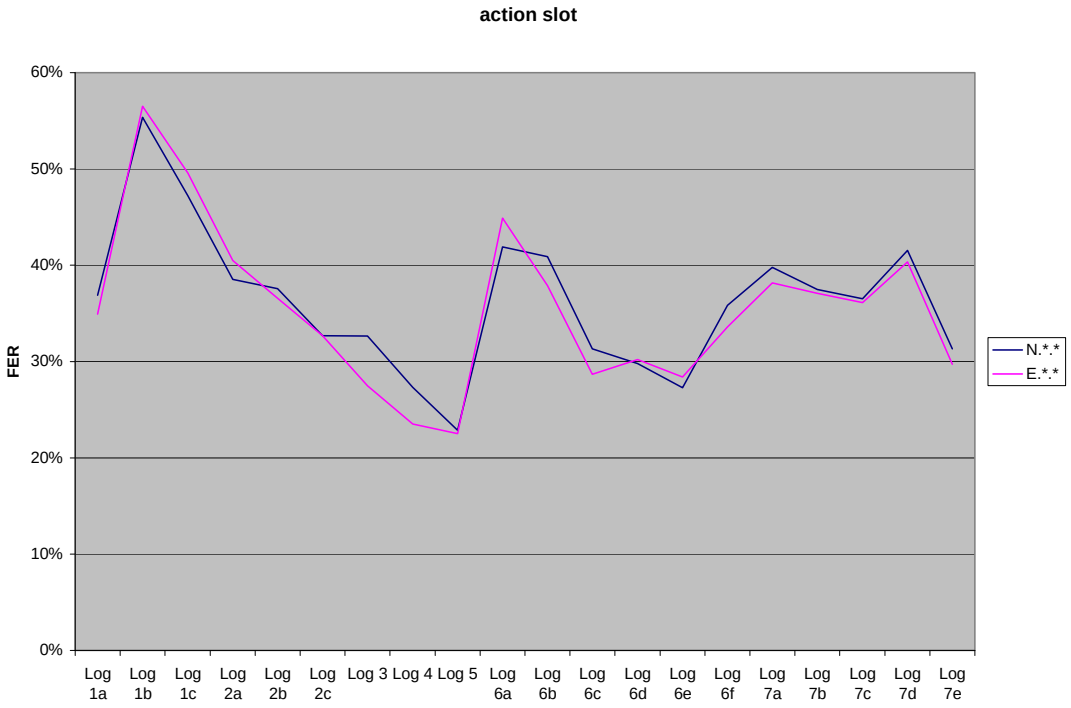

Figure 5.10: Fusion Error Rates (FER) for All Logs for Action Slot.

Figure 5.11: Fusion Error Rates (FER) for All Logs for Object Slot.

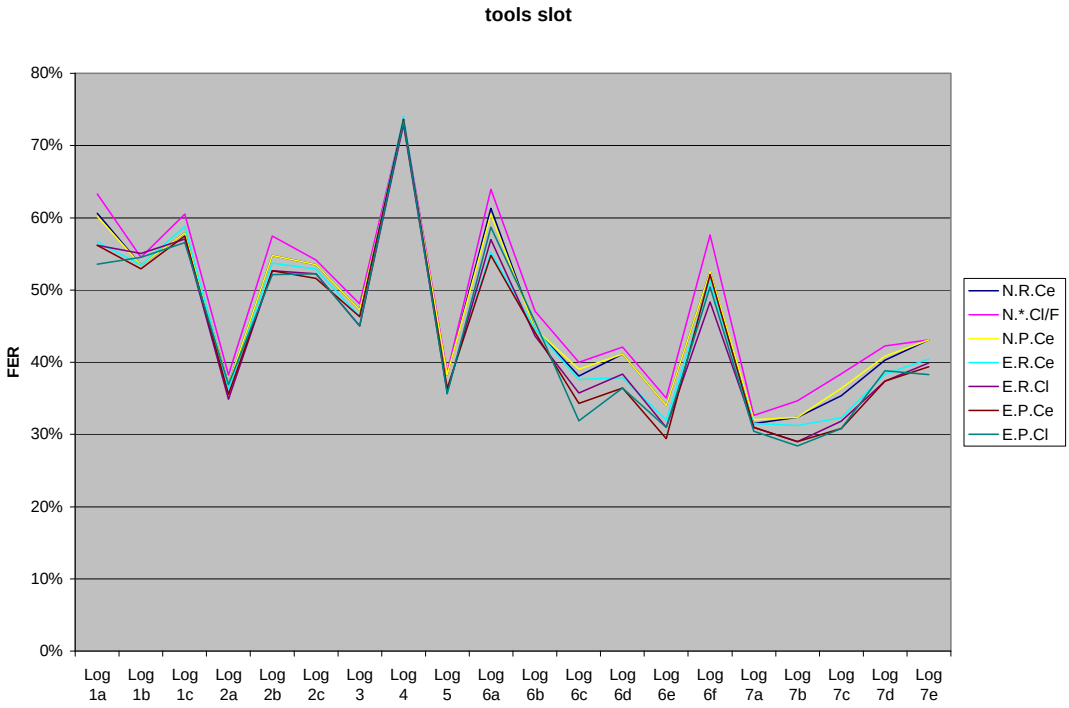

Figure 5.12: Fusion Error Rates (FER) for All Logs for Tools Slot.

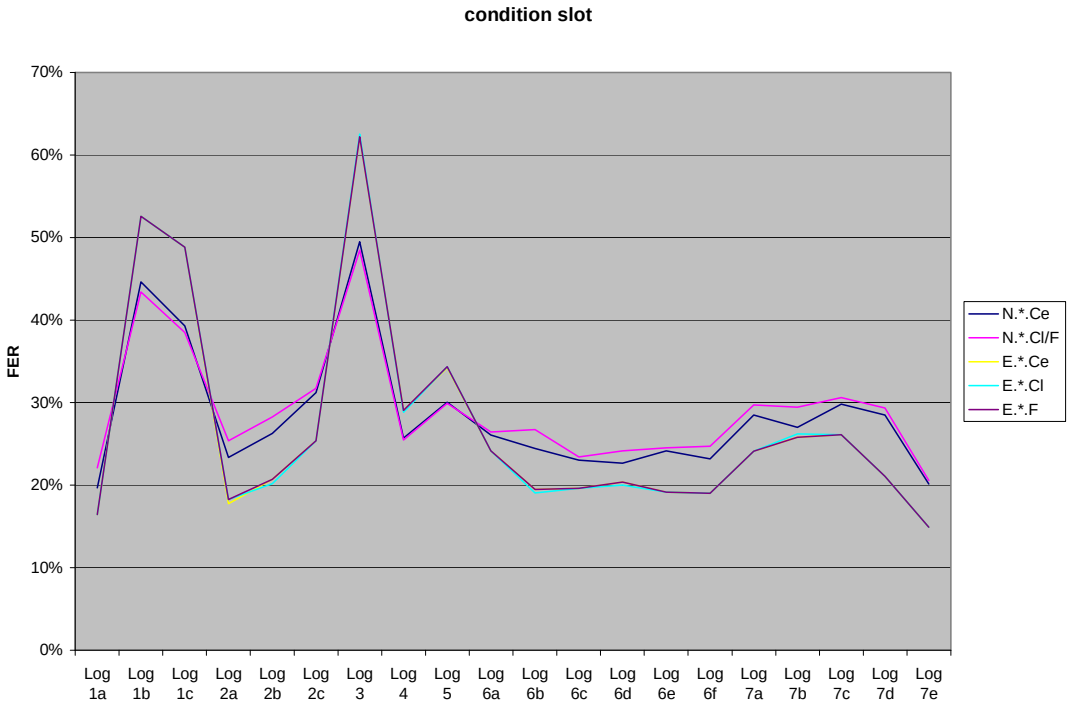

Figure 5.13: Fusion Error Rates (FER) for All Logs for Condition Slot.

However, averaging the FER across all logs (Table 5.9 and Table 5.10), the weights from the data with "errors" that are the closest to the centroid of the largest cluster using the *k* determined by *peakedness* produce the best average FER across all logs. These weights are in the columns labeled E.*.* and E.*.Cl in Table 5.9 and E.P.Cl and E.*.Cl in Table 5.10.

5.8 Comparison with Arbitrary Weights

In order to assess the quality of the optimum Fusion weights relative to any arbitrary weights, we plotted the FER for an arbitrary set of weights. We varied the weights from –1.0 to 1.0 in 0.5 increments, calculated the FER, and plotted the results in Figure 5.14.

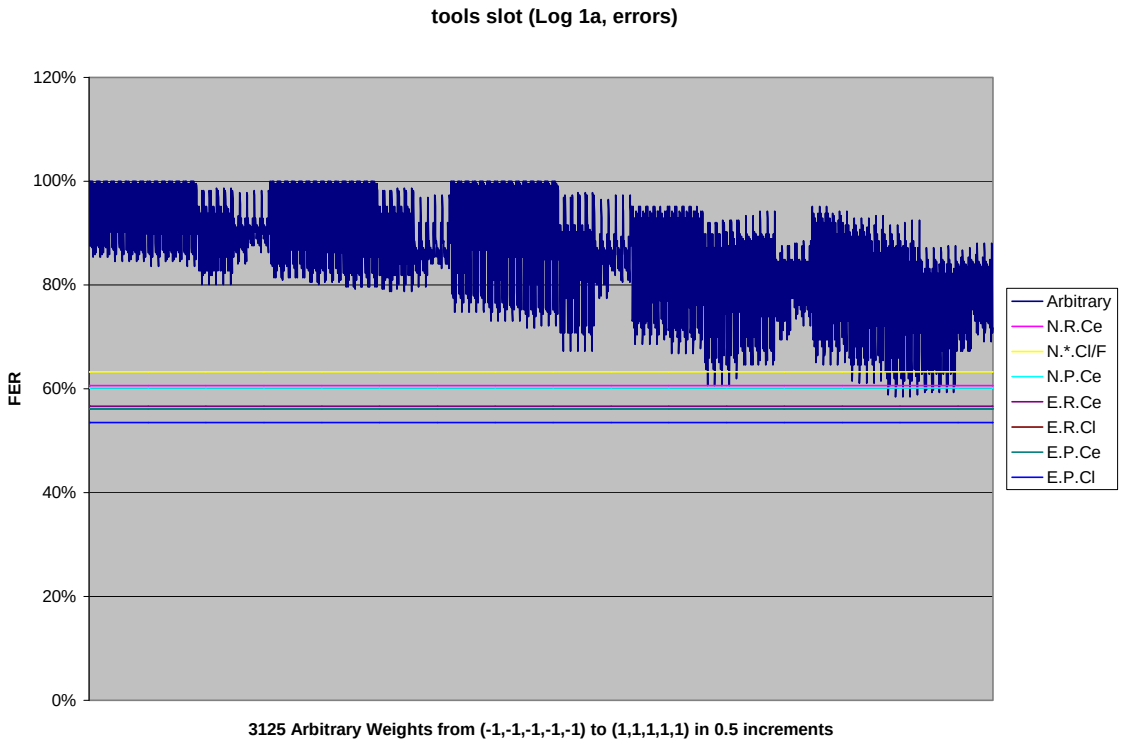

Figure 5.14: Fusion Error Rate (FER) for Arbitrary Weights.

For comparison, we included all the weights determined by clustering. As shown, the set of the Fusion weights is important.

5.9 Fusion Weights Used for Experiments

We used the set of weights from the data with "errors" that is the closest to the centroid of the largest cluster using the *k* determined by *peakedness,* which are summarized in Table 5.11.

		Reconstruction Error			Peakedness		
Slot	**Weight**	**Centroid**	**Closest**	**Frequent**	**Centroid**	**Closest**	**Frequent**
action	realWorldW	0.20	0.20	0.20	0.20	0.20	0.20
	historyW	0.60	0.60	0.60	0.60	0.60	0.60
	speechW	1.00	1.00	1.00	1.00	1.00	1.00
object	realWorldW	0.09	0.12	n/a	0.09	0.12	n/a
	historyW	0.83	0.84	n/a	0.92	0.84	n/a
	fieldOfVisionW	-0.73	-0.63	n/a	-0.79	-0.63	n/a
	pointingW	0.02	-0.06	n/a	0.01	-0.06	n/a
	speechW	1.06	0.92	n/a	1.00	0.92	n/a
tools	realWorldW	1.10	1.24	n/a	0.96	1.08	n/a
	historyW	0.56	0.60	n/a	0.58	0.54	n/a
	fieldOfVisionW	-1.32	-1.08	n/a	-0.78	-0.92	n/a
	pointingW	0.30	0.52	n/a	0.34	0.52	n/a
	speechW	0.70	0.76	n/a	0.67	0.55	n/a
condition	realWorldW	0.88	0.84	1.00	0.88	0.84	1.00
	historyW	0.19	0.12	0.20	0.19	0.12	0.20
	speechW	0.17	0.12	0.20	0.17	0.12	0.20

errors

Table 5.11: Fusion Weights Used for Experiments.

Chapter 6 Experimental Results

The hypothesis for this research project is that "Applying the Human Computer Interaction (HCI) concepts of using multiple modalities, dialog management, context, and semantics to Human Robot Interaction (HRI) will improve the performance of Instruction Based Learning (IBL) compared to only using speech". We tested the hypothesis by simulating a domestic robot that can be taught to clean a house using a multi-modal interface.

The hypothesis was tested with the following seven scenarios and three grounding modes (Optimistic, Cautious, and Pessimistic):

- Speech only (no modalities and no context)
- Speech + Real World Context (one type of context)
- Speech + Dialog History (another type of context)
- Speech + Pointing (one type of modality)
- Speech + Field of Vision (another type of modality)
- Speech + Head Nodding (another type of modality)
- Speech + All

All 166 tasks in the corpus were tested in each of the 21 combinations of dialog modes and scenarios.

The learning process was evaluated using two metrics:

How many times did the robot ask a "clarifying" question?

How many times did the robot learn the task correctly?

The measurements from each experiment are provided in the Appendix and analyzed in detail in this chapter.

A total of 210 experiments were performed, during which the robot was taught 3,486 tasks [40]. Each task took an average of 1.07 minutes to teach.

6.1 Speech Only

The results of the Speech Only (no modalities and no context) experiments are summarized in Figure 6.1.

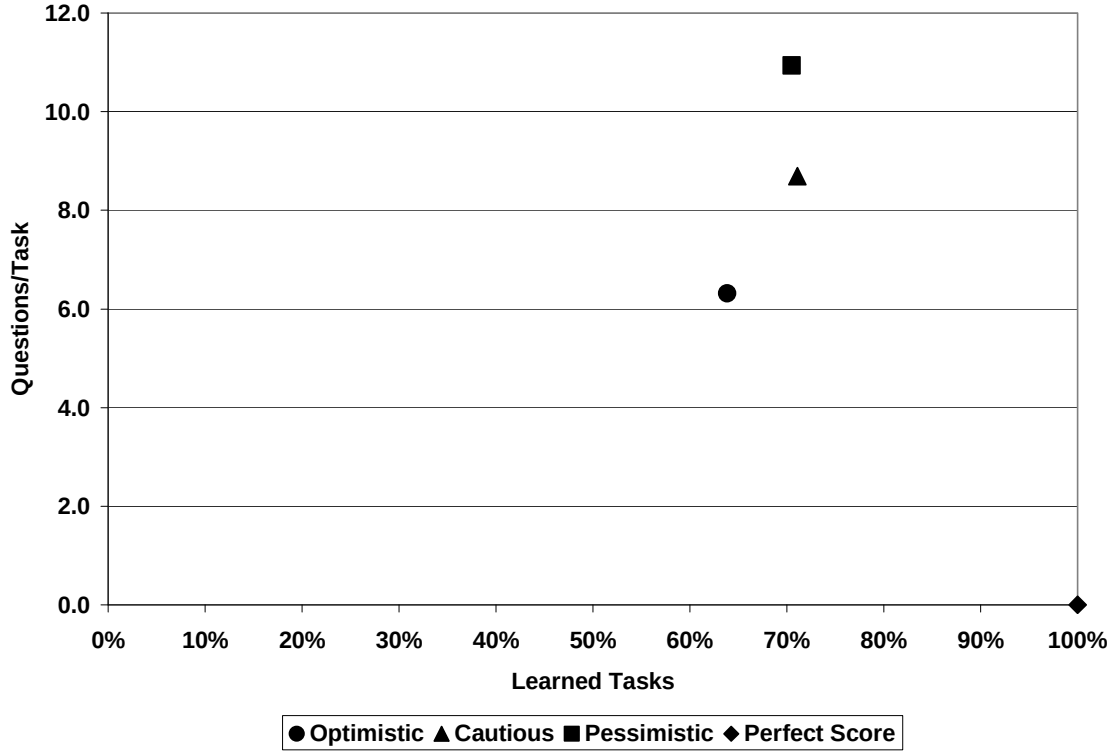

Figure 6.1: Speech Only Results.

Each of the three points represents the results of teaching the robot 166 tasks in one of the grounding modes.

The y-axis values, Questions/Task, were calculated by dividing the total number of Clarifying Questions by the total number of Tasks Learned Correctly as follows:

Questions/Task (Optimistic) = 670 / 106 = 6.3

Questions/Task (Cautious) = 1026 / 118 = 8.7

Questions/Task (Pessimistic) = 1280 / 117 = 10.9

The x-axis values, Learned Tasks, were calculated by dividing the total number of Tasks Learned Correctly by the total number of Tasks in Lesson Plan as follows:

Learned Task (Optimistic) = 106 / 166 = 64%

Learned Task (Cautious) = 118 / 166 = 71%

Learned Task (Pessimistic) = 117 / 166 = 70%

The results indicate that the number of questions asked by the robot is as expected, with Optimistic being the least and Pessimistic being the most. The accuracy of the tasks learned correctly is lower when the robot assumes its understanding is correct without asking any clarifying questions (Optimistic). Pessimistic grounding with more questions did not seem to improve the Learned Task accuracy over Cautious grounding with fewer questions. However, the difference between the two is only 1% and is probably statistically insignificant.

These results will be used in later sections as a baseline for determining if other modalities and context do indeed improve the human robot interaction.

Before examining the results of the other experiments, we will first look at which grounding mode is “better”.

A "perfect score" would be 0 Questions per Task and 100% Learned Tasks as shown in Figure 6.1. Thus, one way to measure which grounding mode is better, is to calculate the Euclidean distance from the "perfect score" to each point as follows:

$$d_i = \text{sqrt}((1 - x_i)^2 + (0 - y_i)^2)$$

where,

i = Grounding mode: Optimistic, Cautious, or Optimistic

x_i = Learned Tasks for grounding mode i

y_i = Normalized Questions/Task for grounding mode i

Using this method, a smaller d_i indicates a better human robot interaction. Table 6.1 shows these calculations for the Speech Only results of Figure 6.1.

Grounding	% Correct	Questions/Task	Normalized Questions/Task	Euclidean Distance
Optimistic	64%	6.3	0.58	0.68
Cautious	71%	8.7	0.79	0.85
Pessimistic	70%	10.9	1.00	1.04

Table 6.1: Euclidean Distance for Speech Only Results.

Using this approach indicates that Optimistic grounding is better than Cautious grounding, and Cautious grounding is better than Pessimistic grounding. However, this approach makes an implicit assumption about the importance of not asking too many questions versus understanding the task correctly. In the real world, this importance depends on the application. For example, the importance of getting the task correct would be much more important than not asking too many questions for a robot surgeon. On the other hand, getting the task somewhat correct might be less

important to a robot building a structure on Mars than not asking too many questions because of the time-delay in the communications between Mars and Earth.

We can address this issue by introducing a constant, which we will call λ, where,

λ = importance of understanding the task correctly (accuracy)

1 - λ = importance of not asking too many questions (succinctness)

$0 \leq \lambda \leq 1$

Now the HRI measure, which we will refer to as h_i, is calculated as follows:

$$h_i = \text{sqrt}(\lambda(1 - x_i)^2 + (1 - \lambda)(0 - y_i)^2)$$

where,

i = Grounding mode: Optimistic, Cautious, or Optimistic

x_i = Learned Tasks for grounding mode i

y_i = Normalized Questions/Task for grounding mode i

λ = importance of understanding the task correctly (accuracy)

A lower value of h_i is better than a higher one.

Figure 6.2 shows a plot of h_i for various values of λ. (The values used for λ in Figure 6.2 and all other plots of h_i for various values of λ are 0.0, 0.2, 0.4, 0.6, 0.8, and 1.0).

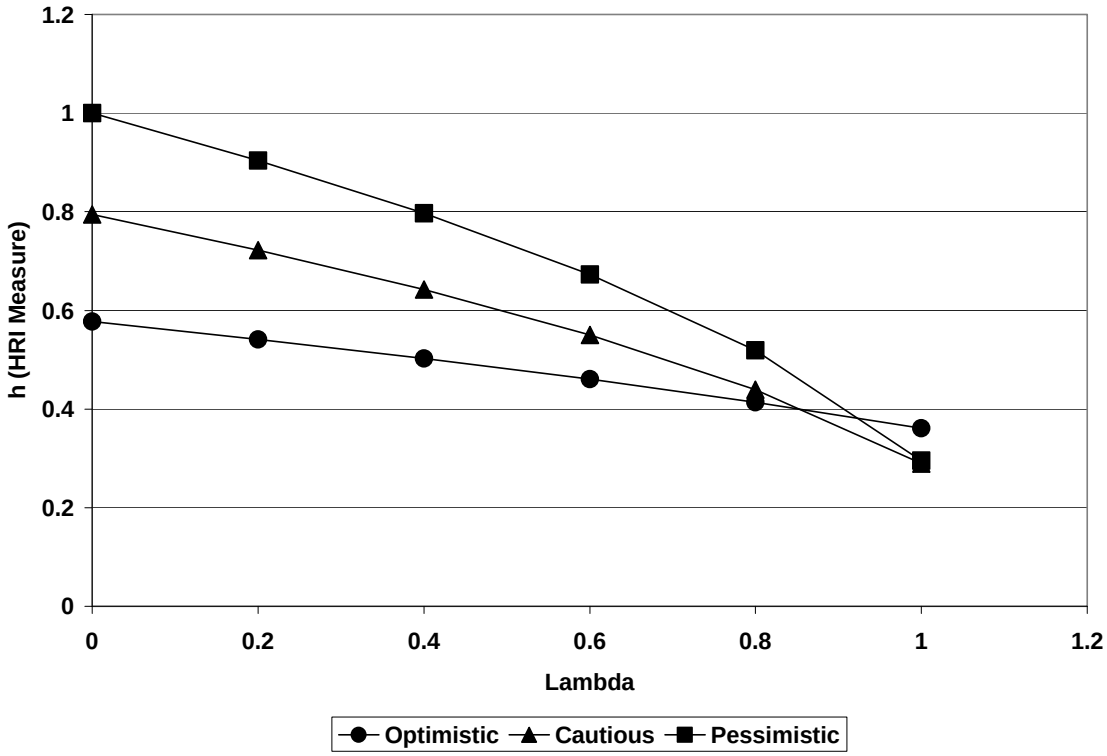

Figure 6.2: Speech Only Comparison of Grounding Modes.

Figure 6.2 illustrates that for λ less than 0.8 (i.e., the importance of accuracy is less than 80%), Optimistic grounding is the best (i.e., the HRI measure is the lowest). It also shows that as λ increases (i.e., accuracy becomes more important), Optimistic grounding becomes the worst (i.e., the HRI measure is the highest). Figure 6.2 shows that for all values of λ, Cautious grounding is better than Pessimistic.

6.2 Speech + Real World Context

The results of the Speech + Real World Context (one type of context) experiments are summarized and compared with the results of the Speech Only experiments in Figure 6.3.

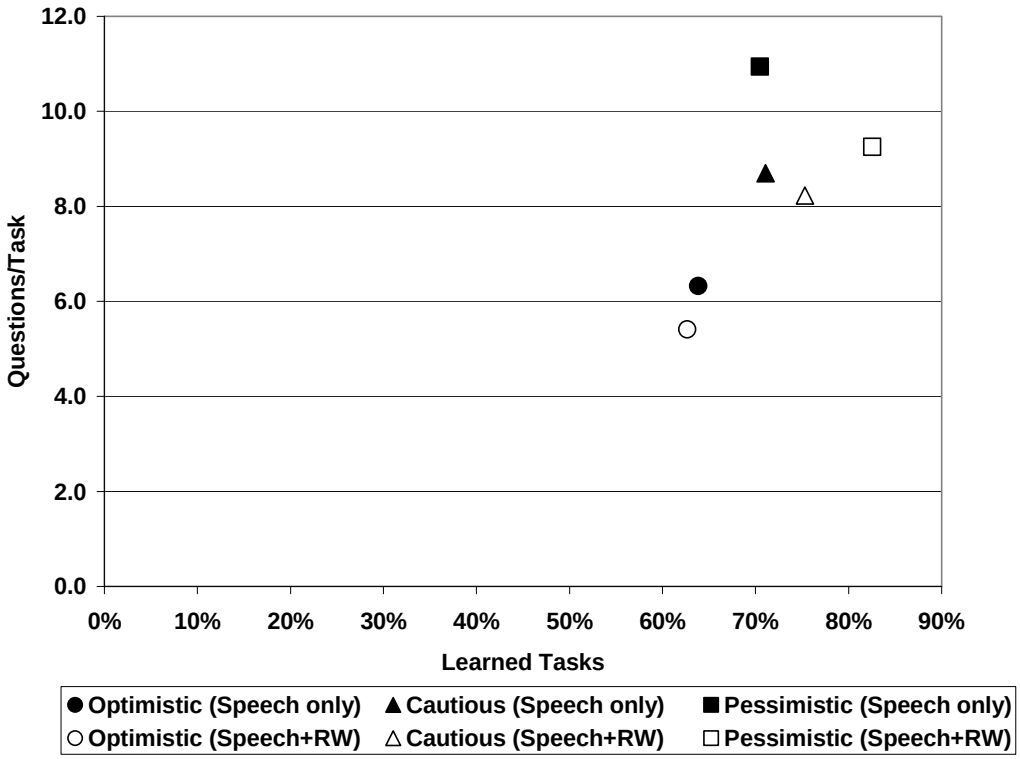

Figure 6.3: Speech + Real World Context Results.

Figure 6.3 illustrates that:

- Real World Context improves HRI (Human Robot Interaction) in terms of both Questions/Task and Learned Tasks for Cautious and Pessimistic grounding.
- Real World Context improves HRI in terms of Questions/Task, but not in terms of Learned Tasks for Optimistic grounding. This is probably due to the robot not verifying incorrect slot values assumed from Real World Context. Thus, the robot asks fewer questions because it assumes more from its Real World Context, but it is wrong more often because its assumed data are incorrect.

To determine if the hypothesis that Real World Context improves HRI is true, a dependent t-test was performed comparing the Speech Only *h* with the Speech + Real

World Context *h* for various values of λ [68]. Table 6.2 shows that Real World Context improves HRI for all values of λ with a less than 16% probability that the results are due to chance.

Lambda	0.0	0.2	0.4	0.6	0.8	1.0
Speech + Real World Context Mean	0.70	0.64	0.57	0.50	0.40	0.27
Speech Only Mean	0.79	0.72	0.65	0.56	0.46	0.32
Probability results are due to chance	5%	6%	6%	8%	10%	16%

Table 6.2: Dependent T-Test of Speech Only and Speech + Real World Context.

Figure 6.4 displays a comparison of grounding modes for the Speech + Real World Context experiments for various values of λ.

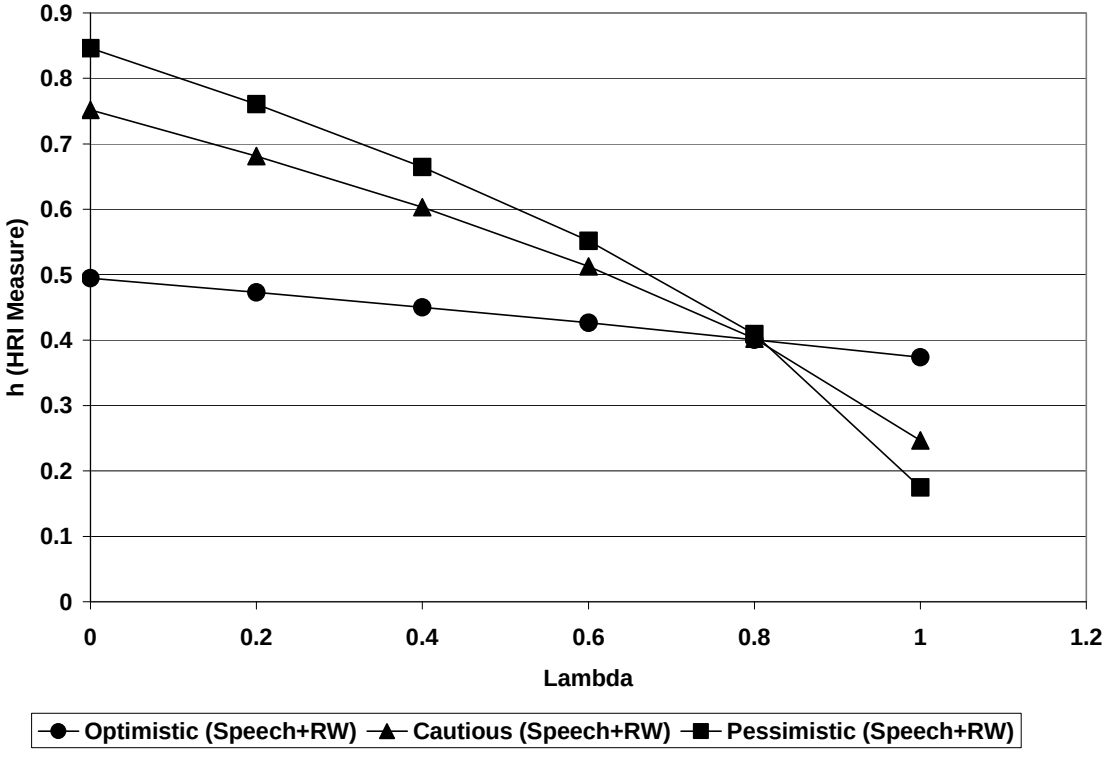

Figure 6.4: Speech + Real World Context Comparison of Grounding Modes.

Figure 6.4 shows that for λ less than 0.8 (i.e., the importance of accuracy is less than 80%), Optimistic grounding is the best (i.e., the HRI measure is the lowest), Cautious grounding is worse, and Pessimistic grounding is the worst (i.e., the HRI

measure is the highest). It also shows that as λ increases (i.e., accuracy becomes more important), Pessimistic grounding becomes the best, Cautious grounding is worse, and Optimistic grounding is the worst.

6.3 Speech + Dialog History

The results of the Speech + Dialog History (another type of context) experiments are summarized and compared with the results of the Speech Only experiments in Figure 6.5.

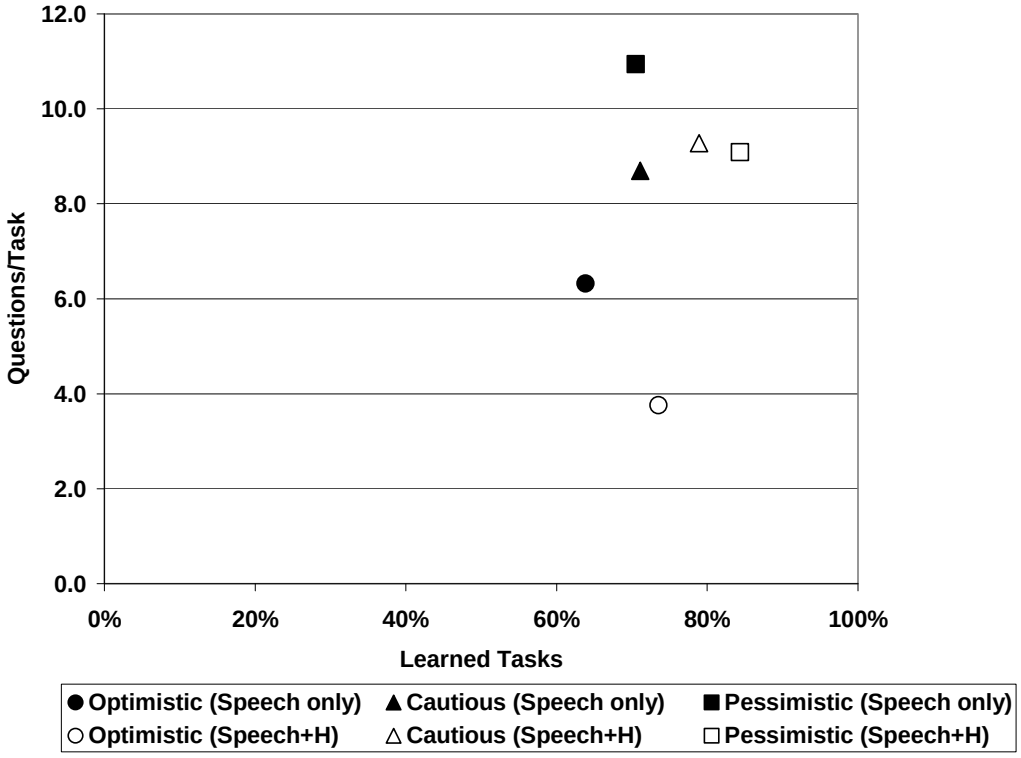

Figure 6.5: Speech + Dialog History Results.

Figure 6.5 shows:

- Dialog History improves HRI in terms of both Questions/Task and Learned Tasks for Optimistic and Pessimistic grounding.

- Dialog History improves HRI in terms of Learned Tasks, but not in terms of Questions/Task for Cautious grounding. This is probably due to the robot not verifying incorrect slot values, assumed from Dialog History, early enough in the dialog, requiring additional questions to correct later on in the dialog.

To determine if the hypothesis that Dialog History improves HRI is true, a dependent t-test was performed comparing the Speech Only *h* with the Speech + Dialog History *h* for various values of λ [68]. Table 6.3 shows that Dialog History improves HRI for all values of λ with a less than 16% probability that the results are due to chance.

Lambda	0.0	0.2	0.4	0.6	0.8	1.0
Speech + Dialog History Mean	0.67	0.61	0.55	0.47	0.37	0.21
Speech Only Mean	0.79	0.72	0.65	0.56	0.46	0.32
Probability results are due to chance	16%	15%	13%	11%	7%	1%

Table 6.3: Dependent T-Test of Speech Only and Speech + Dialog History.

Figure 6.6 shows a comparison of grounding modes for the Speech + Dialog History experiments for various values of λ.

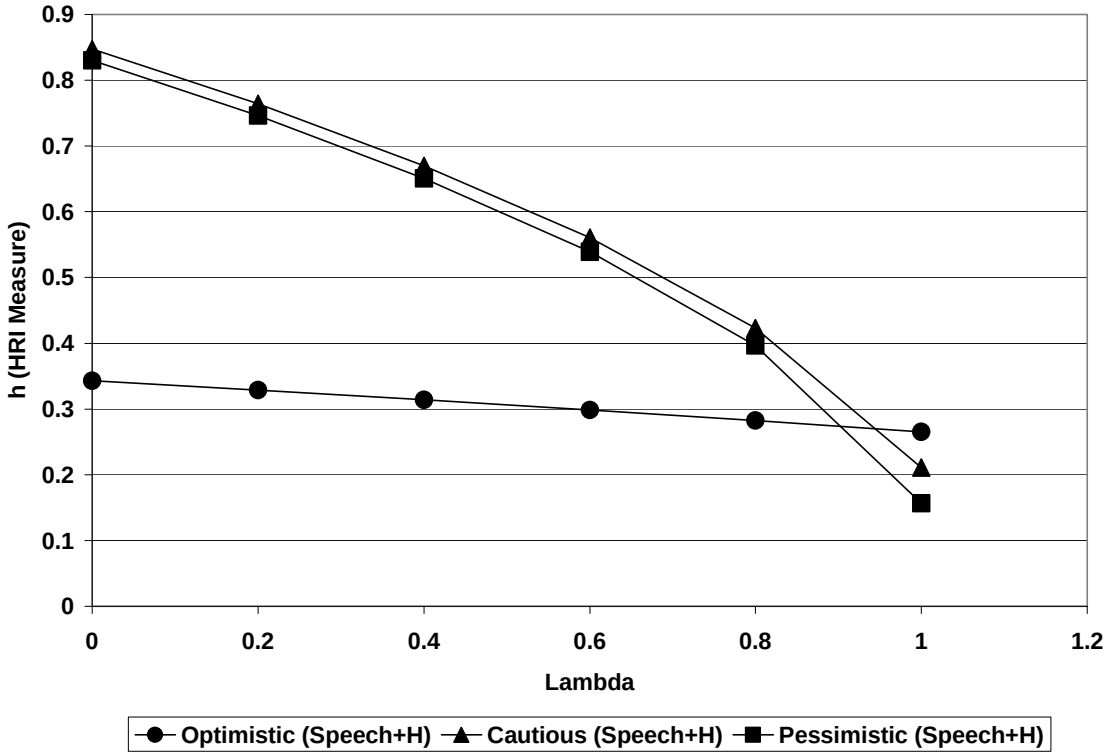

Figure 6.6: Speech + Dialog History Comparison of Grounding Modes.

Figure 6.6 illustrates that for λ less than 0.8 (i.e., the importance of accuracy is less than 80%), Optimistic grounding is the best (i.e., the HRI measure is the lowest), Pessimistic grounding is worse, and Cautious grounding is the worst (i.e., the HRI measure is the highest). It also shows that as λ increases (i.e., accuracy becomes more important), Pessimistic grounding becomes the best, Cautious grounding is worse, and Optimistic grounding is the worst. It also shows that for all values of λ, Pessimistic grounding is better than Cautious grounding.

6.4 Speech + Pointing

The results of the Speech + Pointing (one type of modality) experiments are summarized and compared with the results of the Speech Only experiments in Figure 6.7.

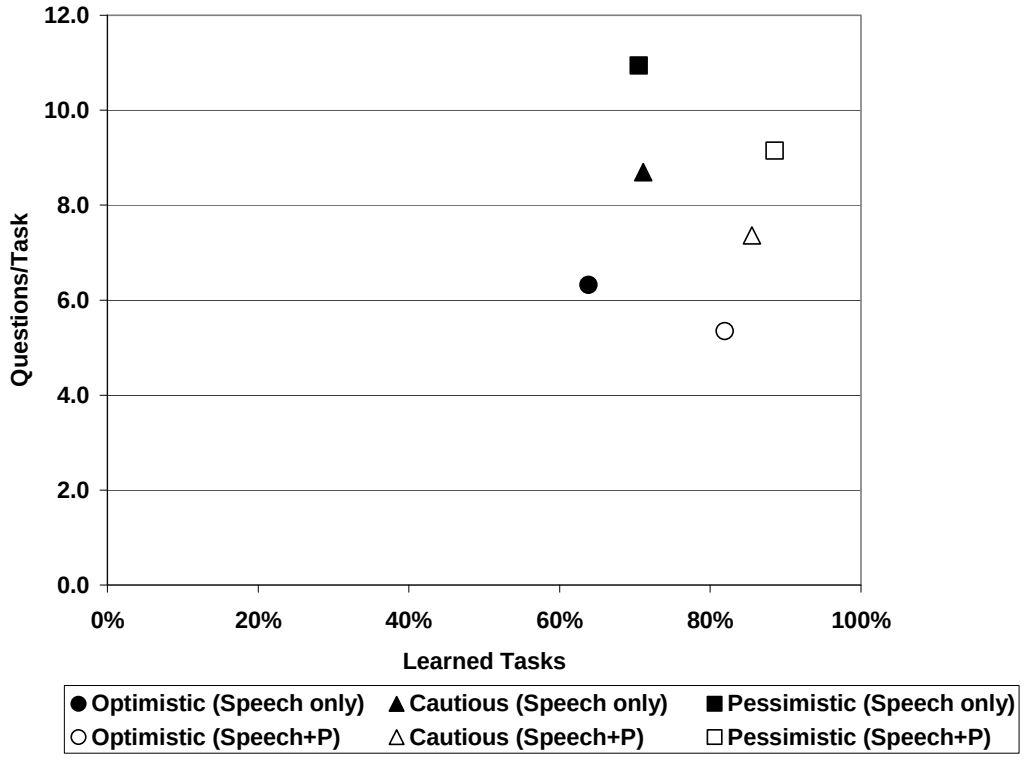

Figure 6.7: Speech + Pointing Results.

Figure 6.7 shows:

- Pointing improves HRI in terms of both Questions/Task and Learned Tasks for Optimistic, Cautious, and Pessimistic grounding.

To determine if the hypothesis that Pointing improves HRI is true, a dependent t-test was performed comparing the Speech Only *h* with the Speech + Pointing *h* for various values of λ [68]. Table 6.4 shows that Pointing improves HRI for all values of λ with a less than 2% probability that the results are due to chance.

Lambda	0.0	0.2	0.4	0.6	0.8	1.0
Speech + Pointing Mean	0.67	0.60	0.53	0.44	0.33	0.15
Speech Only Mean	0.79	0.72	0.65	0.56	0.46	0.32
Probability results are due to chance	1.4%	0.9%	0.5%	0.2%	0.2%	0.3%

Table 6.4: Dependent T-Test of Speech Only and Speech + Pointing.

Figure 6.8 shows a comparison of grounding modes for the Speech + Pointing experiments for various values of λ.

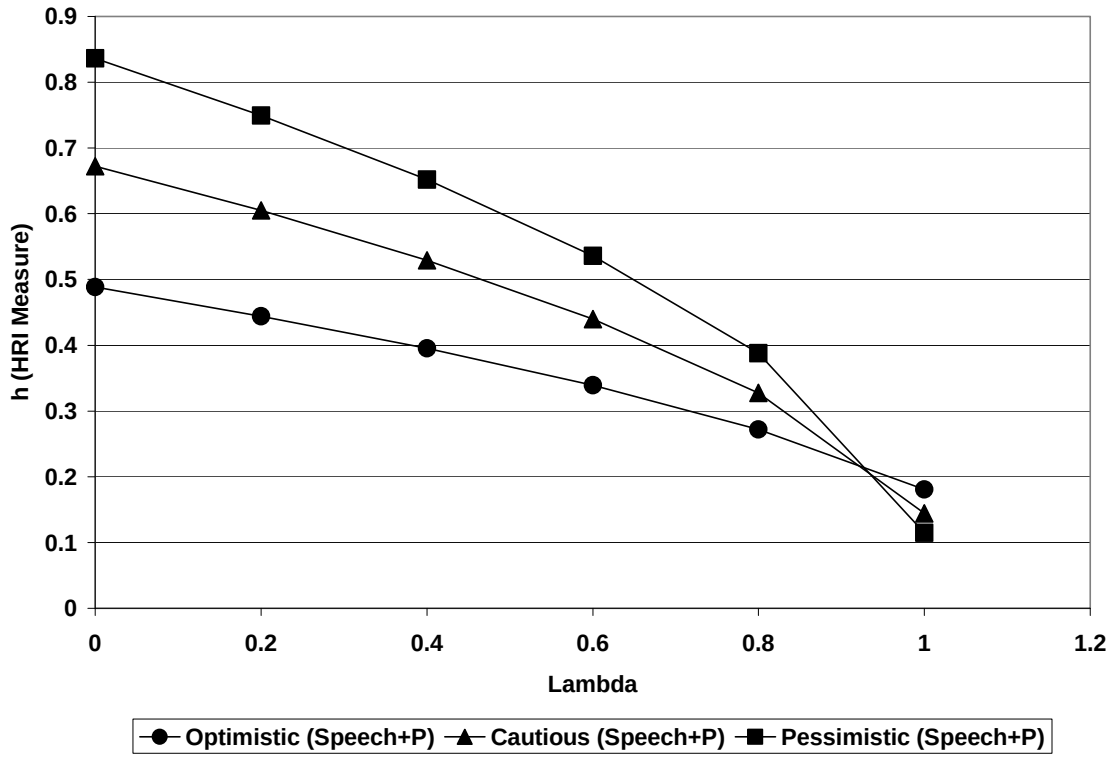

Figure 6.8: Speech + Pointing Comparison of Grounding Modes.

Figure 6.8 shows that for λ less than 0.8 (i.e., the importance of accuracy is less than 80%), Optimistic grounding is the best (i.e., the HRI measure is the lowest), Cautious grounding is worse, and Pessimistic grounding is the worst (i.e., the HRI measure is the highest). It also shows that as λ increases (i.e., accuracy becomes more important), Pessimistic grounding becomes the best, Cautious grounding is worse, and Optimistic grounding is the worst.

6.5 Speech + Field of Vision

The results of the Speech + Field of Vision (another type of modality) experiments are summarized and compared with the results of the Speech Only experiments in Figure 6.9.

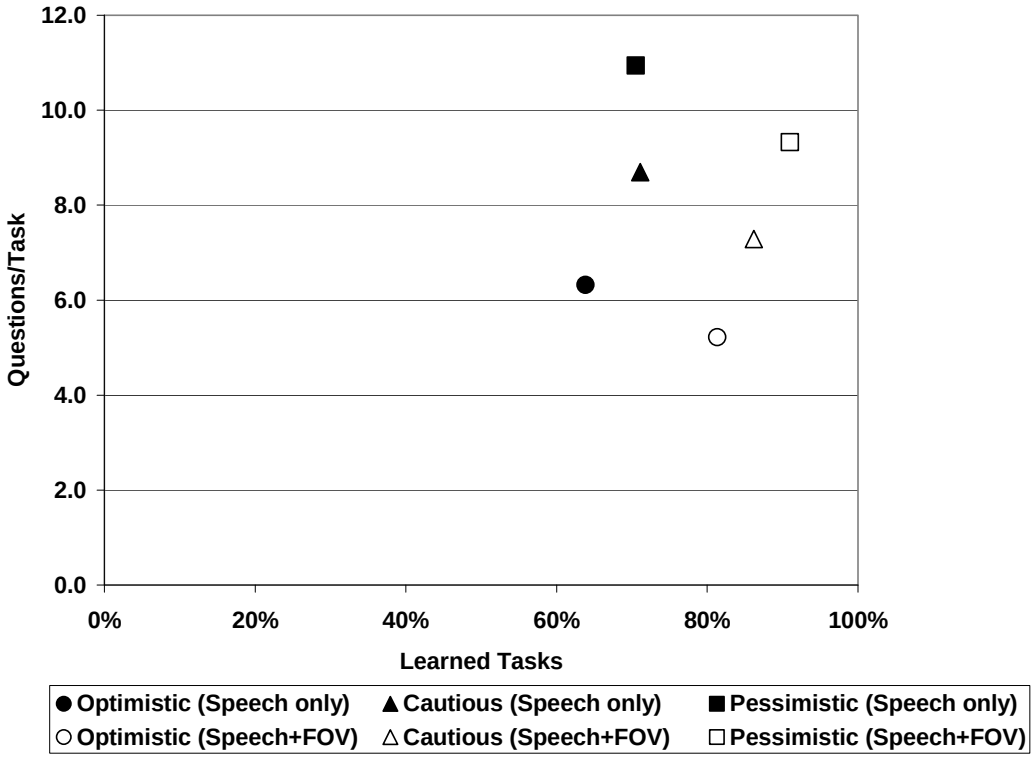

Figure 6.9: Speech + Field of Vision Results.

Figure 6.9 shows:

- Field of Vision improves HRI in terms of both Questions/Task and Learned Tasks for Optimistic, Cautious, and Pessimistic grounding.

To determine if the hypothesis that Field of Vision improves HRI is true, a dependent t-test was performed comparing the Speech Only *h* with the Speech + Field of Vision *h* for various values of λ [68]. Table 6.5 shows that Field of Vision

improves HRI for all values of λ with a less than 0.6% probability that the results are due to chance.

Lambda	0.0	0.2	0.4	0.6	0.8	1.0
Speech + Field of Vision Mean	0.67	0.60	0.52	0.44	0.33	0.14
Speech Only Mean	0.79	0.72	0.65	0.56	0.46	0.32
Probability results are due to chance	0.6%	0.3%	0.1%	0.1%	0.2%	0.4%

Table 6.5: Dependent T-Test of Speech Only and Speech + Field of Vision.

Figure 6.10 displays a comparison of grounding modes for the Speech + Field of Vision experiments for various values of λ.

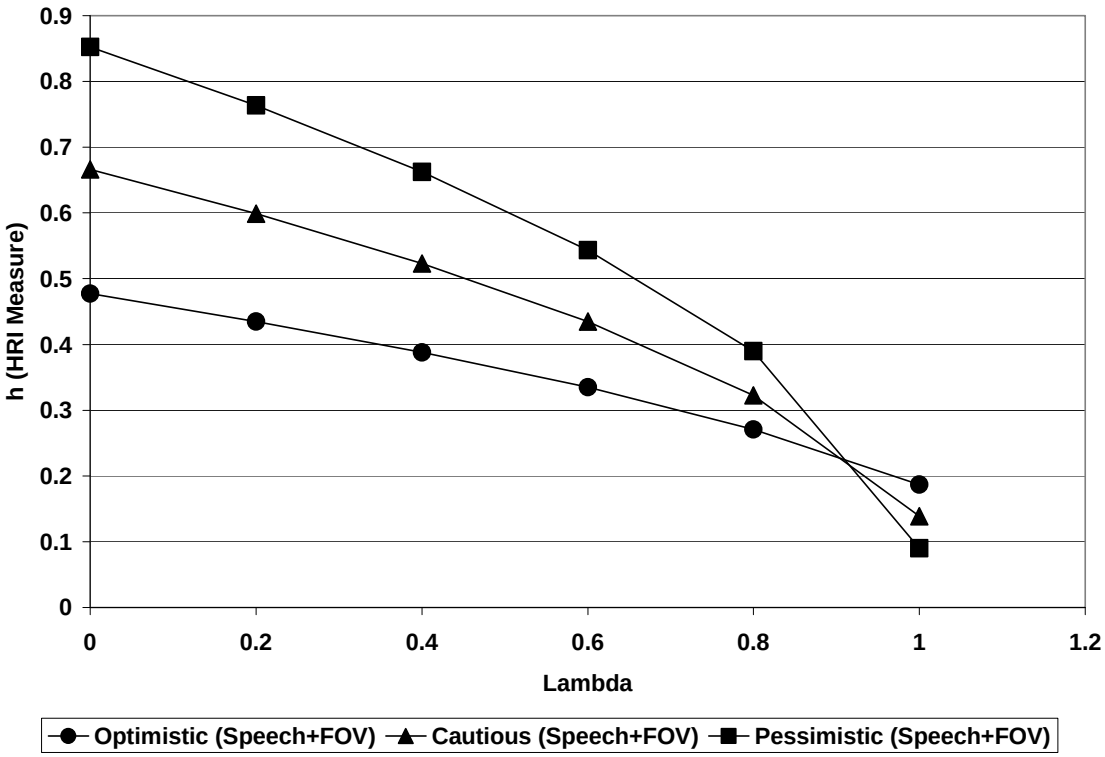

Figure 6.10: Speech + Field of Vision Comparison of Grounding Modes.

Figure 6.10 illustrates that for λ less than 0.8 (i.e., the importance of accuracy is less than 80%), Optimistic grounding is the best (i.e., the HRI measure is the lowest), Cautious grounding is worse, and Pessimistic grounding is the worst (i.e., the HRI measure is the highest). It also shows that as λ increases (i.e., accuracy becomes more

important), Pessimistic grounding becomes the best, Cautious grounding is worse, and Optimistic grounding is the worst.

6.6 Speech + Head Nodding

The results of the Speech + Head Nodding (another type of modality) experiments are summarized and compared with the results of the Speech Only experiments in Figure 6.11.

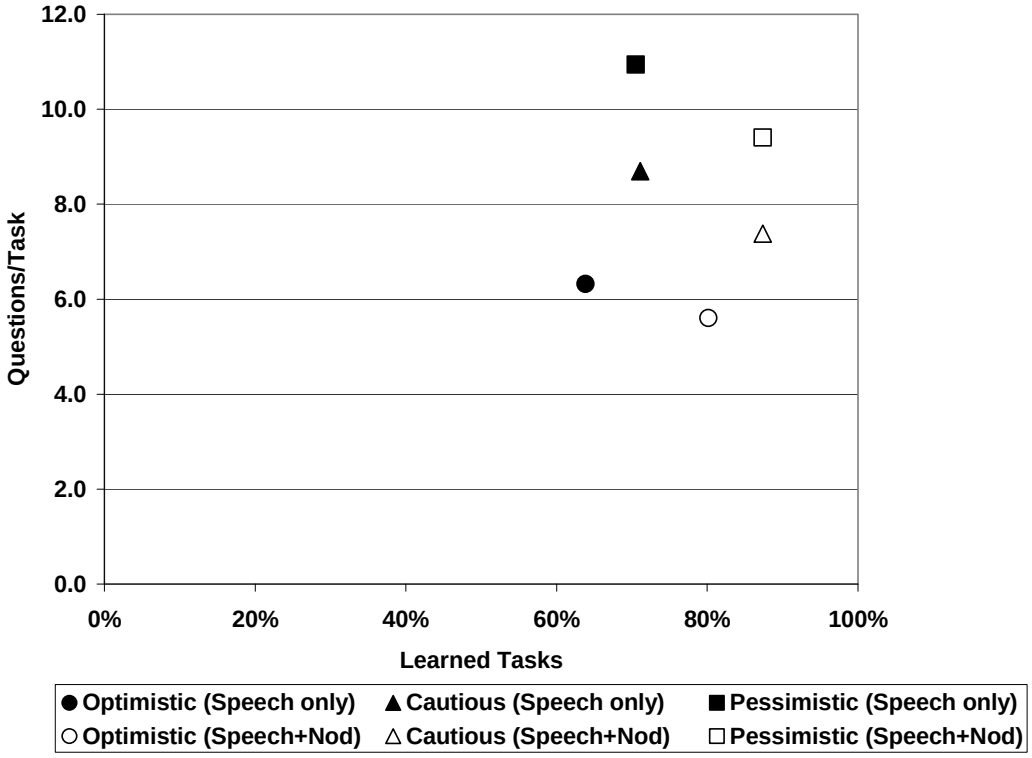

Figure 6.11: Speech + Head Nodding Results.

Figure 6.11 illustrates:

- Head Nodding improves HRI in terms of both Questions/Task and Learned Tasks for Optimistic, Cautious, and Pessimistic grounding.

- Pessimistic grounding with more questions did not seem to improve the Learned Task accuracy over Cautious grounding with fewer questions.

To determine if the hypothesis that Head Nodding improves HRI is true, a dependent t-test was performed comparing the Speech Only *h* with the Speech + Head Nodding *h* for various values of λ [68]. Table 6.6 shows that Head Nodding improves HRI for all values of λ with a less than 2% probability that the results are due to chance.

Lambda	0.0	0.2	0.4	0.6	0.8	1.0
Speech + Head Nodding Mean	0.68	0.61	0.54	0.45	0.34	0.15
Speech Only Mean	0.79	0.72	0.65	0.56	0.46	0.32
Probability results are due to chance	1.99%	1.24%	0.58%	0.12%	0.01%	0.01%

Table 6.6: Dependent T-Test of Speech Only and Speech + Head Nodding.

Figure 6.12 displays a comparison of grounding modes for the Speech + Head Nodding experiments for various values of λ.

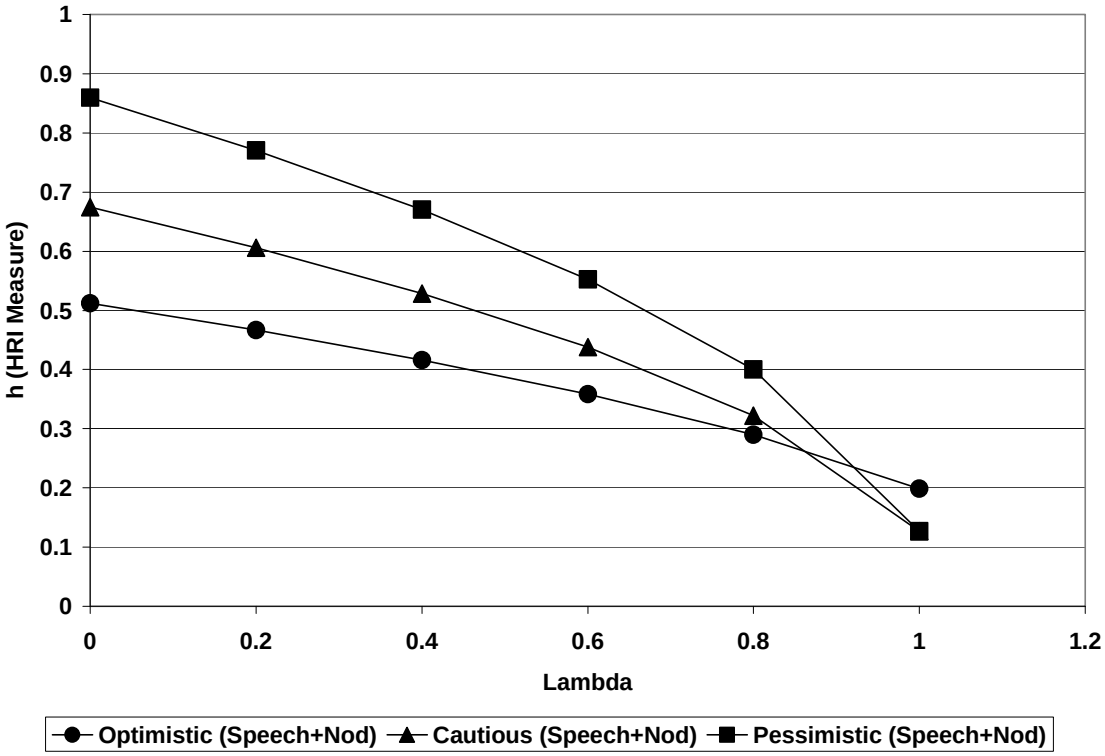

Figure 6.12: Speech + Head Nodding Comparison of Grounding Modes.

Figure 6.12 shows that for λ less than 0.8 (i.e., the importance of accuracy is less than 80%), Optimistic grounding is the best (i.e., the HRI measure is the lowest), Cautious grounding is worse, and Pessimistic grounding is the worst (i.e., the HRI measure is the highest). It also shows that as λ increases (i.e., accuracy becomes more important), Optimistic grounding becomes the worst (i.e., the HRI measure is the highest). Figure 6.12 shows that for most values of λ, Cautious grounding is better than Pessimistic.

6.7 Speech + All

The results of the Speech + All experiments are summarized and compared with the results of the Speech Only experiments in Figure 6.13. All refers to Real World Context, Dialog History, Pointing, Field of Vision, and Head Nodding.

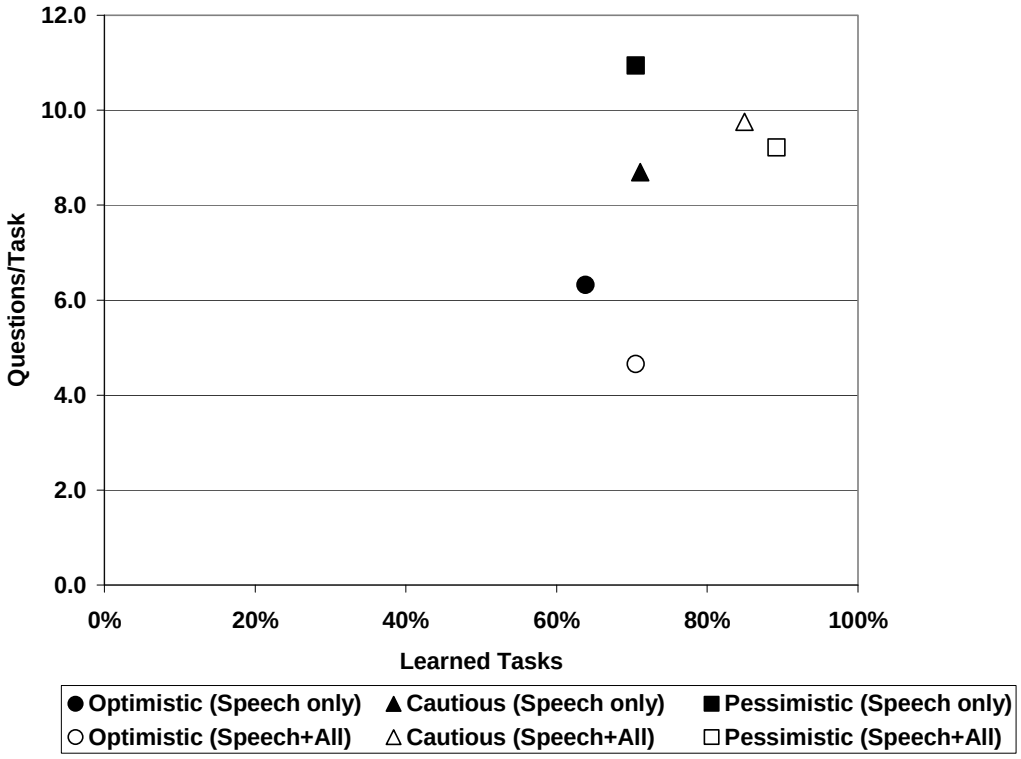

Figure 6.13: Speech + All Results.

Figure 6.13 shows:

- All (Real World Context, Dialog History, Pointing, Field of Vision, and Head Nodding) improves HRI in terms of both Questions/Task and Learned Tasks for Optimistic and Pessimistic grounding.
- All improves HRI in terms of Learned Tasks, but not in terms of Questions/Task for Cautious grounding. This is probably due to the robot not verifying incorrect slot values, assumed from Dialog History, early enough in the dialog, requiring

additional questions to correct later on in the dialog. This phenomenon was also identified in the results of Speech + Dialog History.

To determine if the hypothesis that All improves HRI is true, a dependent t-test was performed comparing the Speech Only *h* with the Speech + All *h* for various values of λ [68]. Table 6.7 shows that All improves HRI for all values of λ with a less than 24% probability that the results are due to chance.

Lambda	0.0	0.2	0.4	0.6	0.8	1.0
Speech + All Mean	0.72	0.65	0.58	0.49	0.38	0.18
Speech Only Mean	0.79	0.72	0.65	0.56	0.46	0.32
Probability results are due to chance	24%	22%	19%	14%	7%	3%

Table 6.7: Dependent T-Test of Speech Only and Speech + All.

Figure 6.14 illustrates a comparison of grounding modes for the Speech + All experiments for various values of λ.

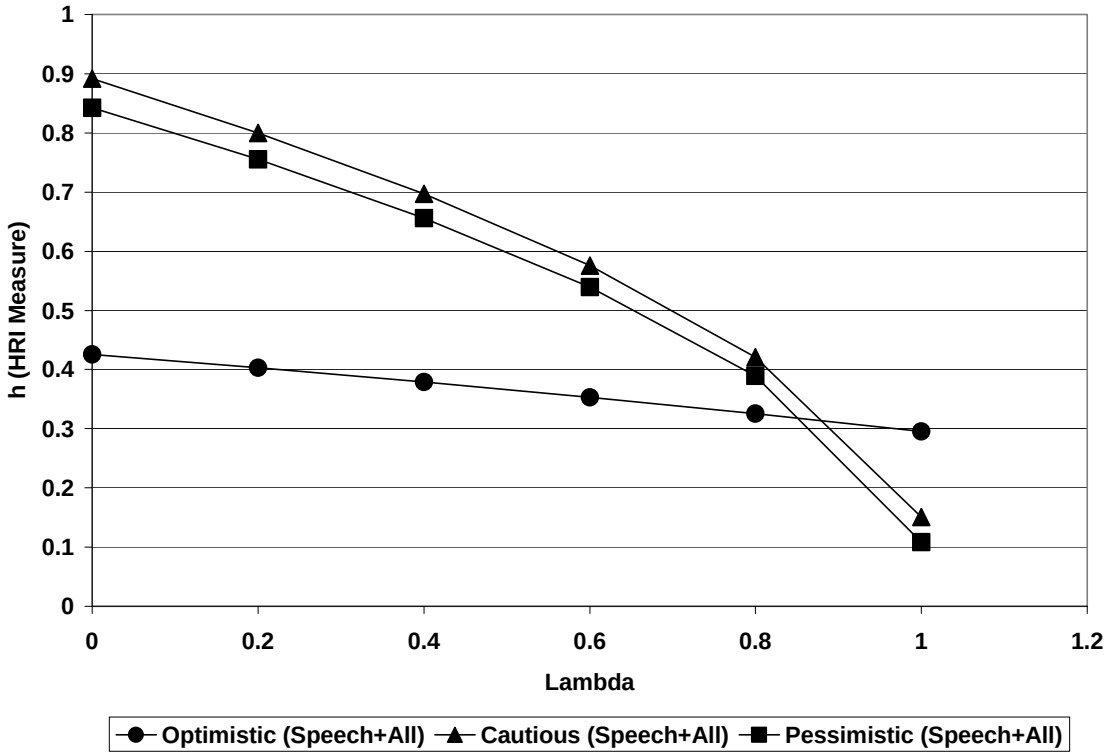

Figure 6.14: Speech + All Comparison of Grounding Modes.

Figure 6.14 shows that for λ less than 0.8 (i.e., the importance of accuracy is less than 80%), Optimistic grounding is the best (i.e., the HRI measure is the lowest), Pessimistic grounding is worse, and Cautious grounding is the worst (i.e., the HRI measure is the highest). It also shows that as λ increases (i.e., accuracy becomes more important), Optimistic grounding becomes the worst (i.e., the HRI measure is the highest).

Figure 6.14 displays that for all values of λ, Pessimistic grounding is better than Cautious.

6.8 Dialog Management

In the earlier sections of this chapter, we analyzed the impact of modality and context on HRI. In this section, we will analyze the impact of dialog management by looking at the experimental results from the perspective of the dialog management grounding modes. Figure 6.15 illustrates the comparison of the results for each of combinations of modalities, contexts, and grounding modes.

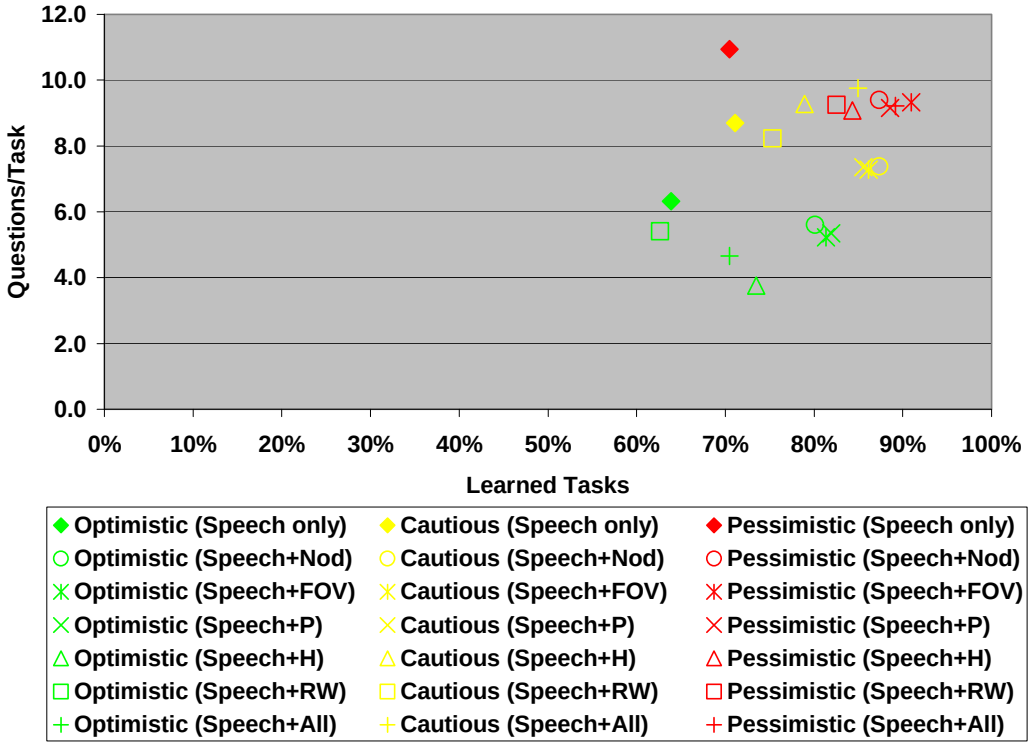

Figure 6.15: Results from All Experiments.

To determine if the hypothesis that dialog management improves HRI is true, a dependent t-test was performed comparing the *h* of the grounding modes for various values of λ [68]. With a less than 1% probability that the results are due to chance, Table 6.8 shows that Optimistic grounding improves HRI more than Cautious

grounding for values of λ up through 0.8. However, when λ is 1.0, Cautious grounding improves HRI more than Optimistic grounding.

Lambda	0.0	0.2	0.4	0.6	0.8	1.0
Cautious Mean	0.76	0.68	0.60	0.50	0.38	0.19
Optimistic Mean	0.47	0.44	0.41	0.37	0.32	0.27
Probability results are due to chance	0.1%	0.1%	0.1%	0.2%	0.9%	0.1%

Table 6.8: Dependent T-Test of Optimistic and Cautious Grounding.

With a less than 0.1% probability that the results are due to chance, Table 6.9 illustrates that Optimistic grounding improves HRI more than Pessimistic grounding for values of λ up through 0.8. However, when λ is 1.0, Pessimistic grounding improves HRI more than Optimistic grounding.

Lambda	0.0	0.2	0.4	0.6	0.8	1.0
Pessimistic Mean	0.87	0.78	0.68	0.56	0.41	0.15
Optimistic Mean	0.47	0.44	0.41	0.37	0.32	0.27
Probability results are due to chance	0.00%	0.00%	0.00%	0.00%	0.05%	0.09%

Table 6.9: Dependent T-Test of Optimistic and Pessimistic Grounding.

With a less than 6% probability that the results are due to chance, Table 6.10 shows that Cautious grounding improves HRI more than Pessimistic grounding for values of λ up through 0.8. However, when λ is 1.0, Pessimistic grounding improves HRI more than Cautious grounding.

Lambda	0.0	0.2	0.4	0.6	0.8	1.0
Pessimistic Mean	0.87	0.78	0.68	0.56	0.41	0.15
Cautious Mean	0.76	0.68	0.60	0.50	0.38	0.19
Probability results are due to chance	2%	2%	2%	3%	6%	1%

Table 6.10: Dependent T-Test of Cautious and Pessimistic Grounding.

6.9 Comparison of Results

Table 6.2 through Table 6.7 provide experimental evidence that the following combinations of modality and context do indeed improve HRI over speech by itself

for all values of λ (i.e., ratio of importance of accuracy to succinctness) as hypothesized:

- Speech + Real World Context (one type of context)
- Speech + Dialog History (another type of context)
- Speech + Pointing (one type of modality)
- Speech + Field of Vision (another type of modality)
- Speech + Head Nodding (another type of modality)
- Speech + All (Real World Context, Dialog History, Pointing, Field of Vision, and Head Nodding)

Table 6.8 through Table 6.10 provide experimental evidence that the dialog management grounding mode that improves HRI the best is dependent on the value of λ (i.e., ratio of importance of accuracy to succinctness) as might be expected, with Optimistic grounding being the best for values of λ less than or equal to 0.8, and Pessimistic grounding being the best for values greater than 0.8. The Pareto principle (a.k.a., the 80-20 rule) says that, for many events, 80% of the effects come from 20% of the causes [42]. The Pareto principle is often generalized to apply to any phenomenon where the 80-20 ratio occurs. The experimental evidence indicates that the Pareto principle also applies here because Optimistic grounding is the best for 80% of the values of λ, and Pessimistic grounding is the best for 20% of the values of λ.

The experimental evidence also suggests that Cautious grounding is of limited use since it is never the best choice for any values of λ that we tested.

Chapter 7 Semantic Integration with Neural Networks

We have discussed one method for fusing the inputs from multiple modalities and contexts. The method multiplies the confidence score provided by the modalities and contexts for each input by a Fusion Weight, sums the products, and then uses the input with highest product sum.

Other methods, such as neural networks could be used, as studied in this dissertation. The inputs to the neural network are the confidence scores from each modality and context input and the outputs are binary: this is the Best choice or this is Not the Best choice.

Two of the log files shown in Table 5.1 were used, one as a training set (object slots from Log 1a) and one as a test set (object slots from Log 7e). The training set consists of 4,674 Not Best samples and 204 Best samples. The test set consists of 2,969 Not Best samples and 185 Best samples. Both sets are unbalanced, in that the number of Not Best samples is much greater than the number of Best samples. This is because, of the numerous possibilities suggested by the five modality and context inputs, only one can be the Best.

The experiments consist of:

- Use two Bayesian classifiers, one using Euclidean distance and the other using Mahalanobis distance [2], to classify the test data based on the statistics of the training set.

- Use a neural network with various hidden nodes and one output node, 1 for Best and 0 for Not Best.
- Use a neural network with various hidden nodes and two output nodes, one for Best and one for Not Best.
- Use a strict output constraint where the result is 0, if the output is less than 0.1; 1 if the output is greater than 0.9; and inconclusive otherwise.
- Use a "fuzzy" output constraint where for one output, the result is 0, if the output is less than 0.5; 1 if the output is greater than 0.5; and inconclusive otherwise (i.e., the output is equal to 0.5); for two outputs, the result is 0, for the smaller output; 1 for the larger output; and inconclusive, if they are equal.
- Use the unbalanced training set.
- Use a small balanced training set consisting of all the Best samples plus an equal amount of randomly selected Not Best samples.
- Use a large balanced training set consisting of all the Not Best samples plus an equal amount of duplicated Best samples.

The set-up and results for each of these is discussed in this chapter.

7.1 Bayesian Classifier

Two different Bayesian Classifiers were used. One classified based on the Euclidean distance between the test sample and the training sample means; the other classified based on the Mahalanobis distance between the test sample and the training sample means weighted by their standard deviations.

The Euclidean distance was calculated as follows:

$$d_{Ej} = \text{sqrt}[\Sigma_{i=1\rightarrow5} (x_i - \mu_{ij})^2]$$

where,

μ_{ij} = mean of the *ith* feature (i.e., input) of the *jth* mean training vector

j = 1 for Best

j = 2 for Not Best

The classification rule was:

if $d_{E1} < d_{E2}$, then x is Best

else x is Not Best

The Mahalanobis distance was calculated as follows:

$$d_{Mj} = \text{sqrt}[\Sigma_{i=1\rightarrow5} ((x_i - \mu_{ij}) / \sigma_{ij})^2]$$

where,

μ_{ij} = mean of the *ith* feature of the *jth* mean training vector

σ_{ij} = standard deviation of the *ith* feature of the *jth* mean training vector

j = 1 for Best

j = 2 for Not Best

The classification rule was:

if $d_{M1} < d_{M2}$, then x is Best

else x is Not Best

The confusion matrix and percent correctly classified are shown in Table 7.1 for both Bayesian classifiers.

		Classified As		
		Best	Not Best	Unknown
Input	Best	144	41	0
	Not Best	285	2684	0
Classified Correctly = 90%				

Euclidean

		Classified As		
		Best	Not Best	Unknown
Input	Best	171	14	0
	Not Best	505	2464	0
Classified Correctly = 84%				

Mahalanobis

Table 7.1: Bayesian Classifier Confusion Matrices.

7.2 Neural Network Classifier

The neural network classifier experiments were divided by training set and number of output nodes as follows:

- Unbalanced training set and one output node
- Small balanced training set and one output node
- Large balanced training set and one output node
- Unbalanced training set and two output nodes
- Small balanced training set and two output nodes
- Large balanced training set and two output nodes

For each set of experiments, the number of hidden nodes in one layer was varied and the results for both the strict and fuzzy output rules were calculated.

7.2.1 Unbalanced Training Set and One Output Node

The neural network was set up with five inputs, one layer of h hidden nodes, and one output node. The nodes were fully interconnected between layers. Each node used the sigmoid function. The neural network was trained using the standard back-propagation learning algorithm with an η of 0.05.

The maximum number of iterations and Mean Square Error (MSE) requirement were determined experimentally. An initial run was made with h = 1, 2, 4, 8, 16, and

32. The MSE achieved is shown in Figure 7.1.

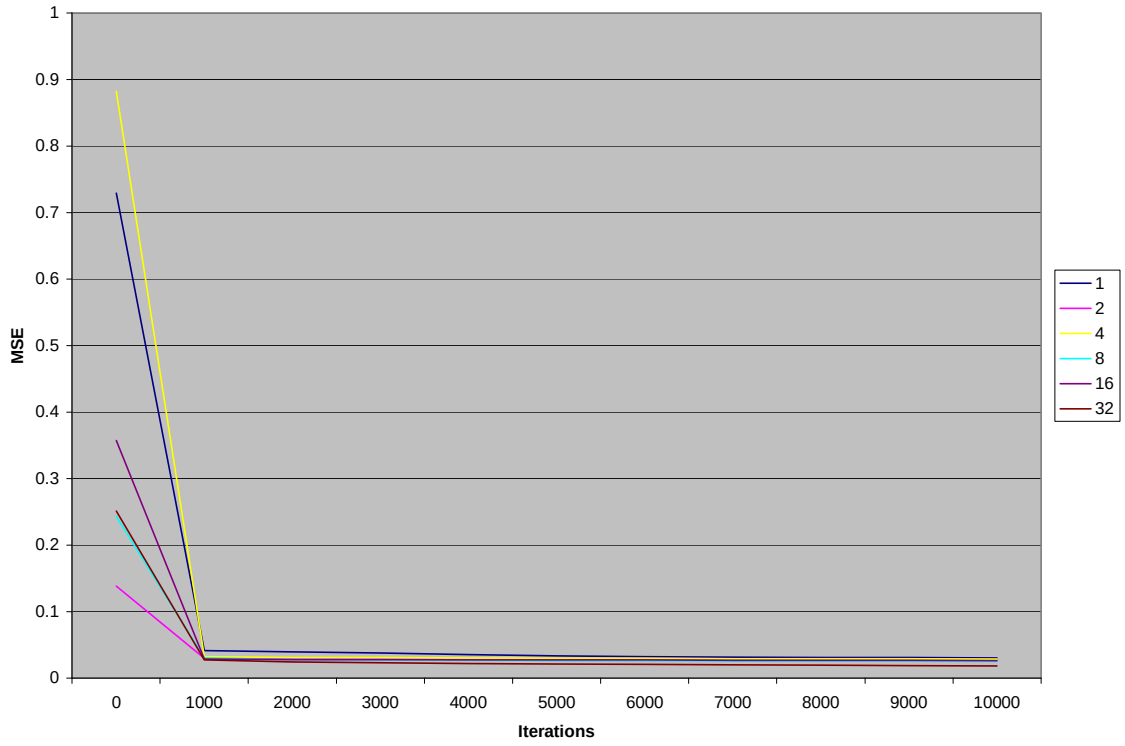

Figure 7.1: Training Results for h = 1, 2, 4, 8, 16, and 32.

The MSE curve has an elbow at 1,000 iterations. A closer look at the elbow reveals the following MSE in Table 7.2.

h	MSE
1	0.0416
2	0.0297
4	0.0320
8	0.0305
16	0.0289
32	0.0276
Average	0.0317

Table 7.2: MSE at 1,000 Iterations.

Since the MSE leveled off after the elbow, finding a maximum number of iterations that would allow the neural network to converge at around 0.03 in the

minimum amount of time would allow as many combinations to be tried as possible. Using $h = 5$ as a starting point, the neural network was trained 20 times to determine how long it took to converge to an MSE of 0.03. The results are shown in Table 7.3.

Run	Iterations until MSE = 0.03
7	491
16	521
5	529
14	839
18	1,313
10	1,411
12	1,413
11	1,606
4	1,942
3	2,482
8	3,333
1	3,576
2	3,756
6	4,071
17	4,659
15	7,324
9	>10,000
13	>10,000
Median	2,212

Table 7.3: Iterations Until MSE = 0.03 for h = 5.

The median of the 20 runs was 2,212, which means half of the runs converged to 0.03 (i.e., the steady state of the neural network) in less than 2,212 iterations. The value of 2,300 was chosen as the maximum number of iterations; and minimum MSE requirement was set at 0.0001. Theoretically, this should have given at least half the runs with an MSE better than the typical MSE of 0.03, and would allow enough latitude for a MSE 300 times better than the worst case MSE.

Although 2,300 does not seem like a large number of iterations, each iteration actually involves 4,878 (the size of the training set) updates of the neural network

weights.

The number of hidden nodes, *h*, was varied from 5 to 100 in increments of 5.

The percent classified correctly for this experiment are compared with that of the Bayesian classifiers in Table 7.4 and Figure 7.2. The confusion matrices for each run are given in the Appendix.

h	Training	Strict Test	Fuzzy Test	Euclidean	Mahalanobis
5	83%	80%	96%	90%	84%
10	84%	82%	96%	90%	84%
15	85%	81%	95%	90%	84%
20	91%	82%	98%	90%	84%
25	87%	81%	98%	90%	84%
30	86%	81%	98%	90%	84%
35	93%	81%	98%	90%	84%
40	2%	1%	98%	90%	84%
45	96%	94%	96%	90%	84%
50	96%	95%	95%	90%	84%
55	95%	93%	95%	90%	84%
60	2%	4%	18%	90%	84%
65	2%	2%	18%	90%	84%
70	95%	94%	96%	90%	84%
75	96%	95%	96%	90%	84%
80	95%	93%	98%	90%	84%
85	2%	4%	6%	90%	84%
90	96%	95%	95%	90%	84%
95	92%	82%	98%	90%	84%
100	3%	5%	78%	90%	84%

Table 7.4: Correctly Classified – Unbalanced Training Set – One Output.

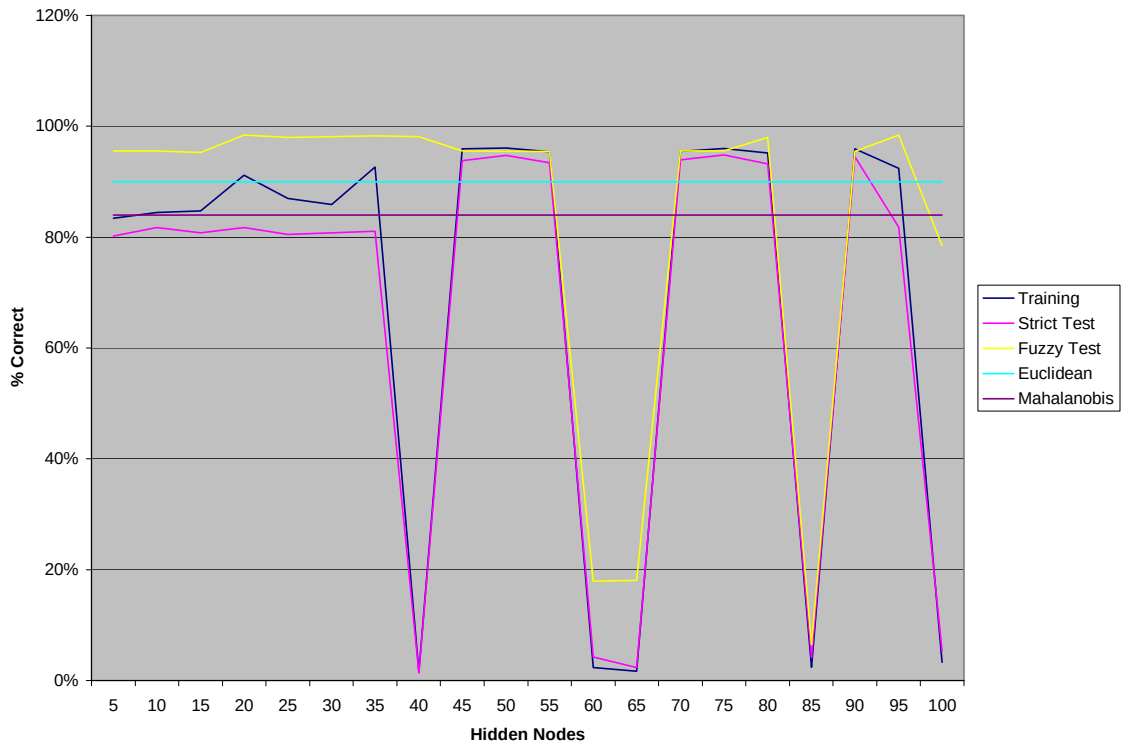

Figure 7.2: Correctly Classified – Unbalanced Training Set – One Output.

The neural network appears to be unstable with 40 or more hidden nodes. With less than 40 hidden nodes, the fuzzy output classifier performs better than both of the Bayesian classifiers and the strict output classifier performs worse than both of them, or

Fuzzy > Euclidean > Mahalanobis > Strict

Also the performance does not seem to vary significantly with the number of hidden nodes.

7.2.2 Small Balanced Training Set and One Output Node

The neural network was set up with five inputs, one layer of *h* hidden nodes, and one output node. The nodes were fully interconnected between layers. Each node

used the sigmoid function. The neural network was trained using the standard back-propagation learning algorithm with an η of 0.05.

The maximum number of iterations was set at 27,000. It is n times greater than the number used for Unbalanced Training Set, where n is the inverse of the ratio between the number of samples in the Unbalanced Training Set (4,878) and the Small Balanced Training Set (405). The number of hidden nodes, h, was varied from 5 to 35 in increments of 5. The maximum value of h was chosen to be less than the number in the first experiment where the neural network appeared to be unstable.

The percent classified correctly for this experiment are compared with that of the Bayesian classifiers in Table 7.5 and Figure 7.3. The confusion matrices for each run are given in the Appendix.

h	Training	Strict Test	Fuzzy Test	Euclidean	Mahalanobis
5	54%	70%	85%	90%	84%
10	46%	64%	86%	90%	84%
15	61%	67%	91%	90%	84%
20	72%	76%	91%	90%	84%
25	69%	77%	91%	90%	84%
30	69%	76%	91%	90%	84%
35	72%	76%	89%	90%	84%

Table 7.5: Correctly Classified – Small Balanced Training Set – One Output.

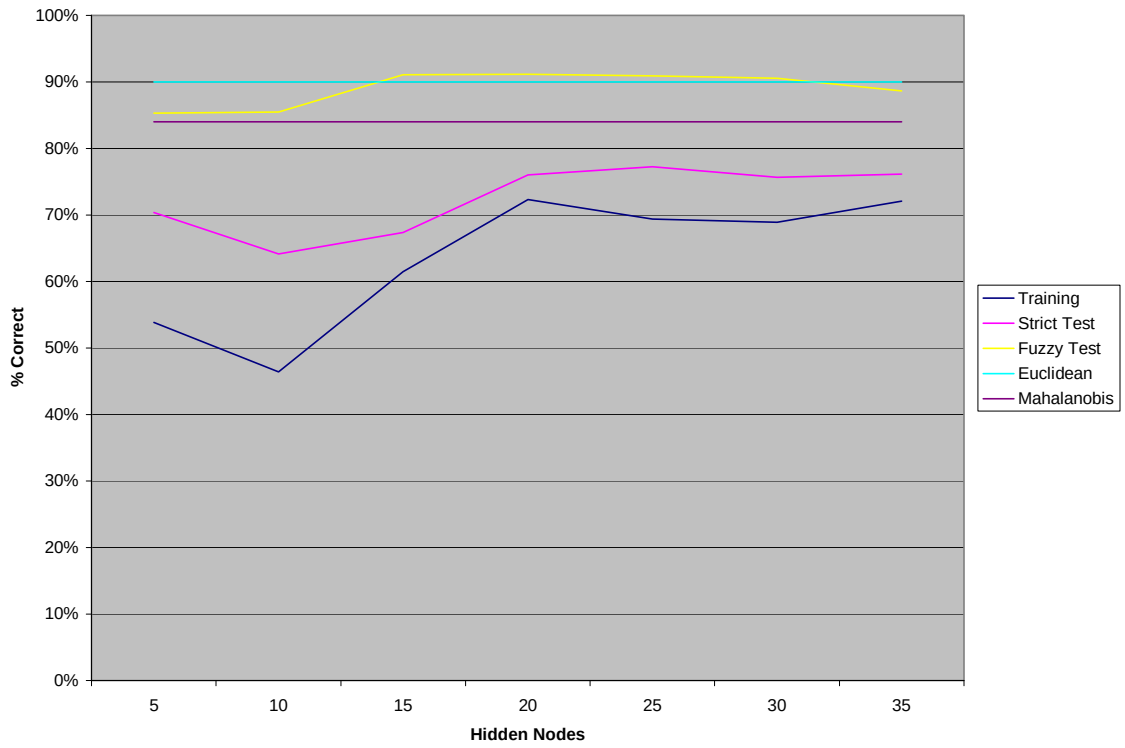

Figure 7.3: Correctly Classified – Small Balanced Training Set – One Output.

The fuzzy output classifier performs worse than the Euclidean classifier, but better than the Mahalanobis classifier for h = 5 and 10. For h = 15 to 35, it performs about the same as the Euclidean classifier. The strict output classifier performs worse than both of them, or:

Euclidean > Fuzzy > Mahalanobis > Strict

The performance seems to improve with more hidden nodes.

7.2.3 Large Balanced Training Set and One Output Node

The neural network was set up with five inputs, one layer of *h* hidden nodes, and one output node. The nodes were fully interconnected between layers. Each node used the sigmoid function. The neural network was trained using the standard back-

propagation learning algorithm with an η of 0.05. The maximum number of iterations was set at 1,000. It is n times smaller than the number used for Unbalanced Training Set, where n is the inverse of the ratio between the number of samples in the Unbalanced Training Set (4,878) and the Large Balanced Training Set (9,349). The number of hidden nodes, h, was varied from 5 to 35 in increments of 5.

The percent classified correctly for this experiment are compared with that of the Bayesian classifiers in Table 7.6 and Figure 7.4. The confusion matrices for each run are given in the Appendix.

h	Training	Strict Test	Fuzzy Test	Euclidean	Mahalanobis
5	20%	10%	84%	90%	84%
10	47%	67%	84%	90%	84%
15	51%	70%	85%	90%	84%
20	43%	62%	86%	90%	84%
25	43%	62%	85%	90%	84%
30	40%	62%	85%	90%	84%
35	43%	62%	85%	90%	84%

Table 7.6: Correctly Classified – Large Balanced Training Set – One Output.

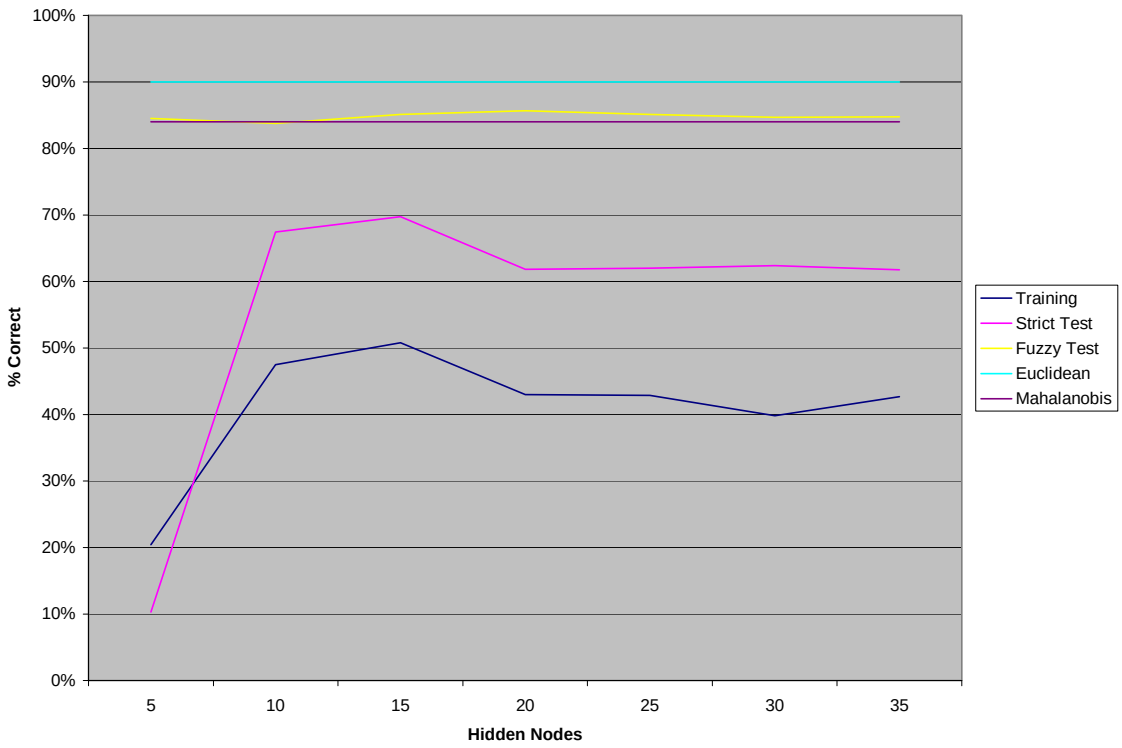

Figure 7.4: Correctly Classified – Large Balanced Training Set – One Output.

Both the strict output and fuzzy output classifiers perform worse than the Euclidean classifier. The fuzzy output classifier performs about the same as the Mahalanobis classifier, and the strict output classifier performs worse than the Mahalanobis classifier, or:

Euclidean > Fuzzy = Mahalanobis > Strict

The performance does seem to vary with the number of hidden nodes. The strict classifier performs best with 15 hidden nodes and the fuzzy classifier with 20 hidden nodes.

7.2.4 Unbalanced Training Set and Two Output Nodes

The neural network was set up with five inputs, one layer of *h* hidden nodes, and

two output nodes. The nodes were fully interconnected between layers. Each node used the sigmoid function. The neural network was trained using the standard back-propagation learning algorithm with an η of 0.05. The maximum number of iterations was set at 2,300. The number of hidden nodes, h, was varied from 5 to 35 in increments of 5.

The percent classified correctly for this experiment are compared with that of the Bayesian classifiers in Table 7.7 and Figure 7.5. The confusion matrices for each run are given in the Appendix.

h	Training	Strict Test	Fuzzy Test	Euclidean	Mahalanobis
5	83%	79%	94%	90%	84%
10	81%	79%	95%	90%	84%
15	83%	79%	98%	90%	84%
20	84%	80%	95%	90%	84%
25	83%	81%	98%	90%	84%
30	83%	80%	96%	90%	84%
35	84%	80%	96%	90%	84%

Table 7.7: Correctly Classified – Unbalanced Training Set – Two Outputs.

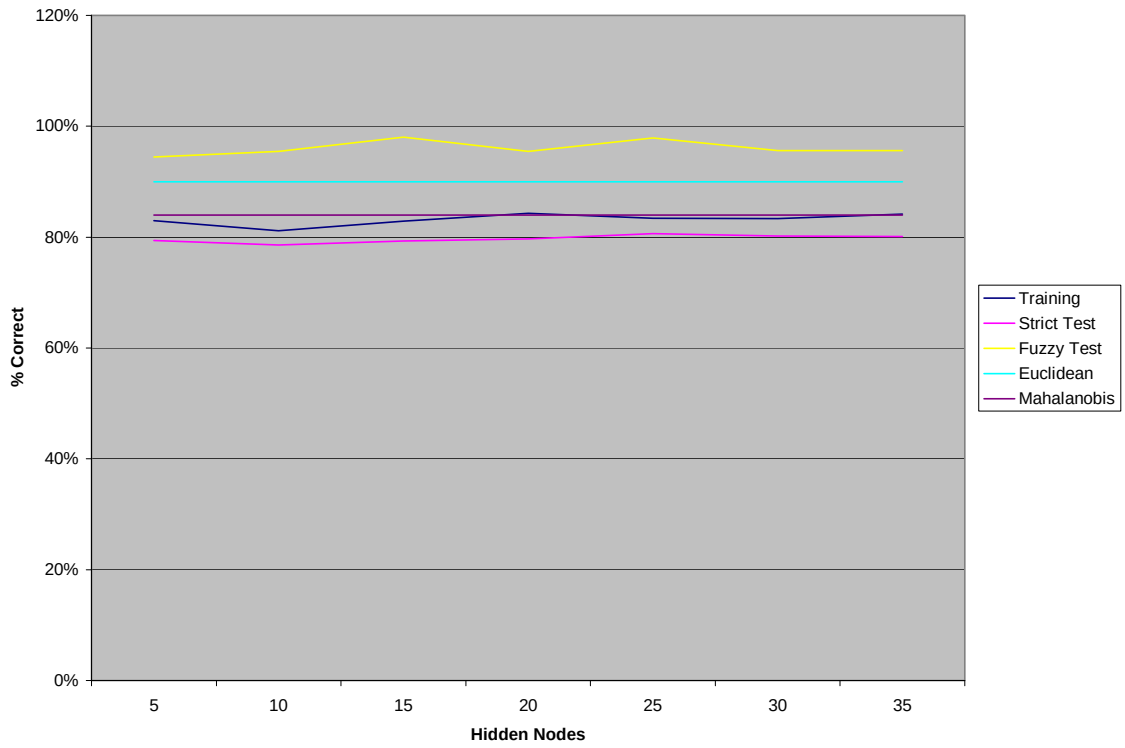

Figure 7.5: Correctly Classified – Unbalanced Training Set – Two Outputs.

The fuzzy output classifier performs better than both of the Bayesian classifiers and the strict output classifier performs worse than both of them, or:

Fuzzy > Euclidean > Mahalanobis > Strict

Also the performance does not seem to vary significantly with the number of hidden nodes.

These results are similar to those for the Unbalanced Training set with one output node and 5 to 35 hidden nodes.

7.2.5 Small Balanced Training Set and Two Output Nodes

The neural network was set up with five inputs, one layer of *h* hidden nodes, and two output nodes. The nodes were fully interconnected between layers. Each node

used the sigmoid function. The neural network was trained using the standard back-propagation learning algorithm with an η of 0.05. The maximum number of iterations was set at 27,000. The number of hidden nodes, h, was varied from 5 to 35 in increments of 5.

The percent classified correctly for this experiment are compared with that of the Bayesian classifiers in Table 7.8 and Figure 7.6. The confusion matrices for each run are given in the Appendix.

h	Training	Strict Test	Fuzzy Test	Euclidean	Mahalanobis
5	12%	2%	85%	90%	84%
10	15%	3%	86%	90%	84%
15	43%	63%	86%	90%	84%
20	45%	64%	91%	90%	84%
25	59%	65%	91%	90%	84%
30	42%	63%	92%	90%	84%
35	61%	66%	91%	90%	84%

Table 7.8: Correctly Classified – Small Balanced Training Set – Two Outputs.

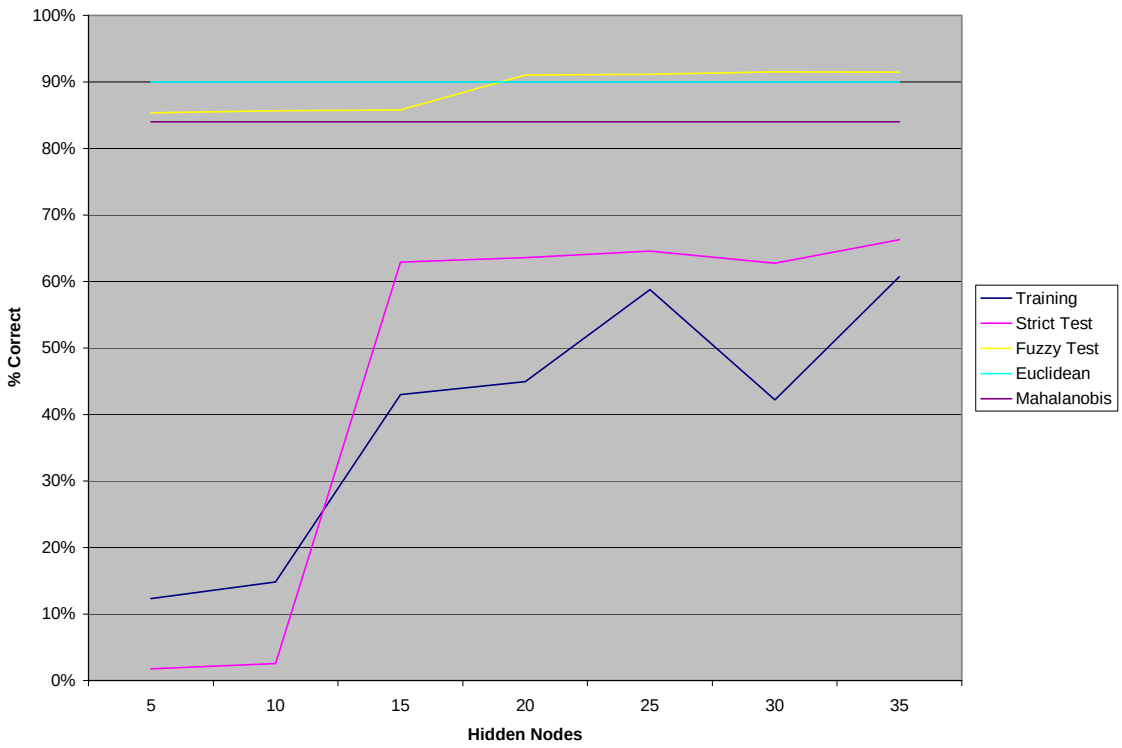

Figure 7.6: Correctly Classified – Small Balanced Training Set – Two Outputs.

The fuzzy output classifier performs worse than the Euclidean classifier, but better than the Mahalanobis classifier for h = 5 to 15. For h = 20 to 35, it performs about the same as the Euclidean classifier. The strict output classifier performs significantly worse than both of them, or

Euclidean > Fuzzy > Mahalanobis > Strict

The performance seems to improve with more hidden nodes.

These results are similar to those for the Small Balanced Training set with one output node.

7.2.6 Large Balanced Training Set and Two Output Nodes

The neural network was set up with five inputs, one layer of *h* hidden nodes, and

two output nodes. The nodes were fully interconnected between layers. Each node used the sigmoid function. The neural network was trained using the standard back-propagation learning algorithm with an η of 0.05. The maximum number of iterations was set at 1,000. The number of hidden nodes, h, was varied from 5 to 35 in increments of 5.

The percent classified correctly for this experiment are compared with that of the Bayesian classifiers in Table 7.9 and Figure 7.7. The confusion matrices for each run are given in the Appendix.

h	Training	Strict Test	Fuzzy Test	Euclidean	Mahalanobis
5	12%	1%	89%	90%	84%
10	37%	58%	83%	90%	84%
15	13%	2%	86%	90%	84%
20	40%	61%	85%	90%	84%
25	14%	4%	84%	90%	84%
30	40%	58%	85%	90%	84%
35	41%	62%	86%	90%	84%

Table 7.9: Correctly Classified – Large Balanced Training Set – Two Outputs.

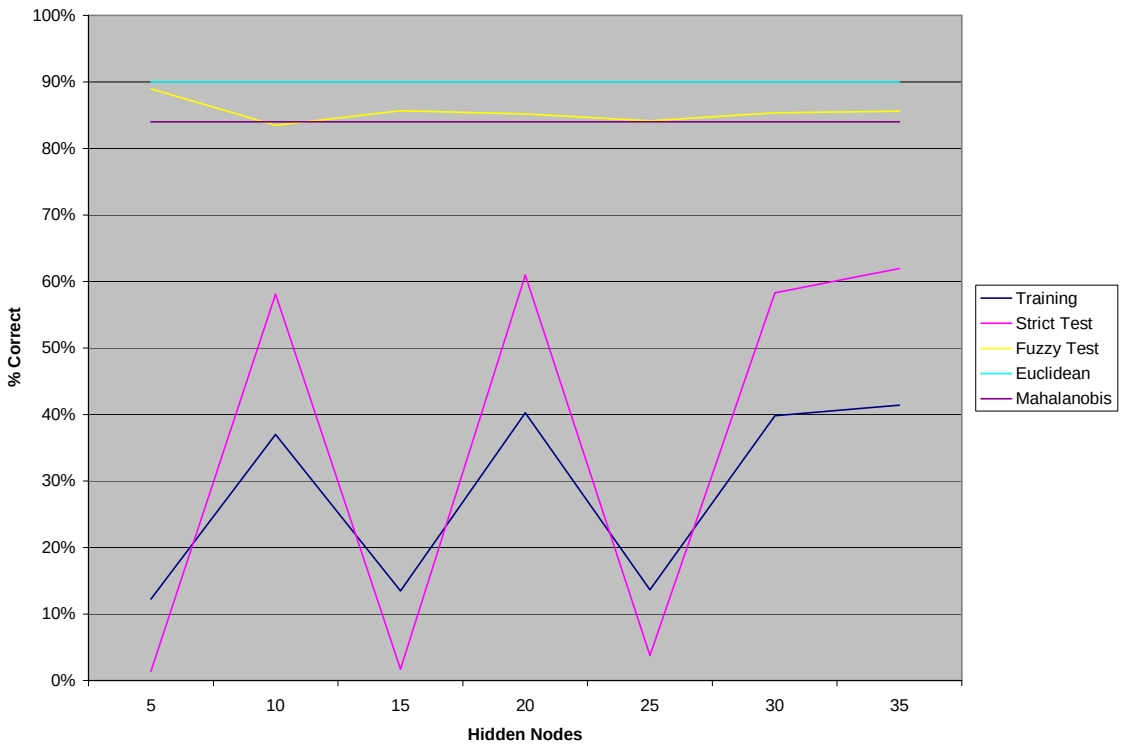

Figure 7.7: Correctly Classified – Large Balanced Training Set – Two Outputs.

The fuzzy output classifier performs worse than the Euclidean classifier, and about the same as the Mahalanobis classifier. The strict output classifier performs significantly worse than both of them, or:

Euclidean > Fuzzy > Mahalanobis > Strict

The performance seems to be much better with an even number of hidden nodes.

These results are not similar to those for the Large Balanced Training set with one output node.

7.3 Analysis of Results

The only neural network configurations that consistently outperformed the Bayesian classifiers were the Unbalance Training Set with one or two fuzzy output

nodes. The percent classified correctly for these two are compared with that of the Bayesian classifiers in Figure 7.8.

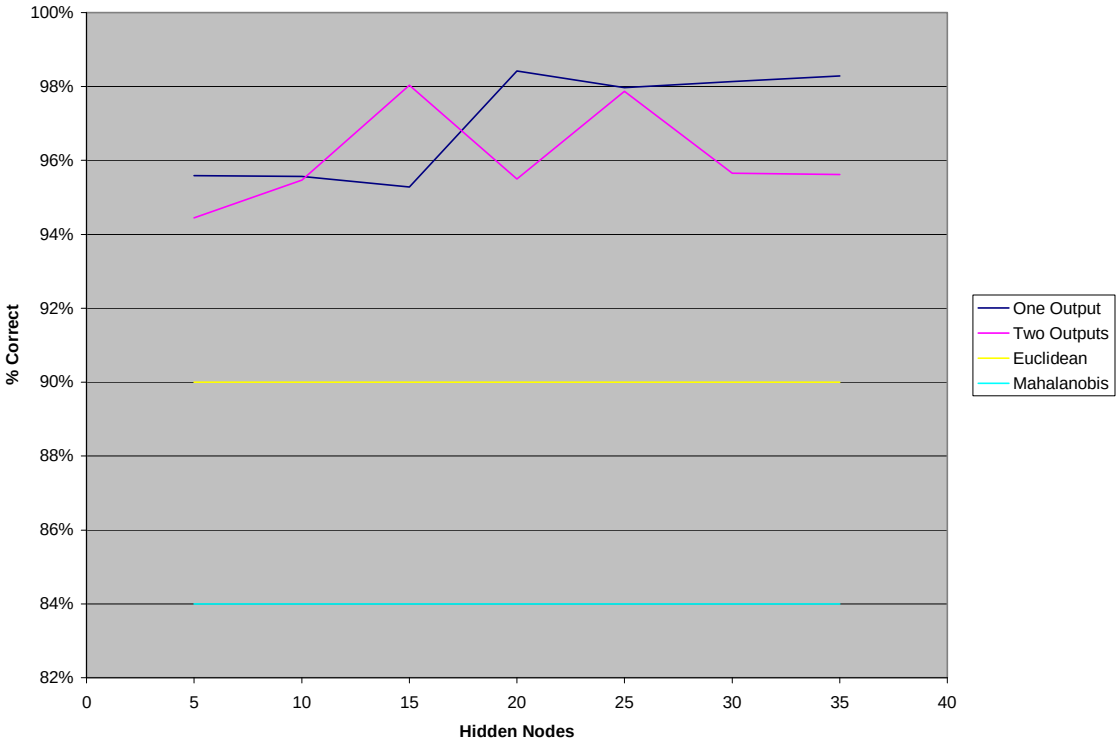

Figure 7.8: Correctly Classified – Unbalanced Training Set – Fuzzy Outputs.

Configurations of 20 to 35 hidden nodes, one fuzzy output, and the Unbalanced Training Set appear to perform the best in comparison to both Bayesian classifiers.

Because the test set is unbalanced, the performance on an individual class basis must also be examined. The percent of Not Best classified correctly for these two are compared with that of the Bayesian classifiers in Figure 7.9, and the percent of Best classified correctly are compared in Figure 7.10.

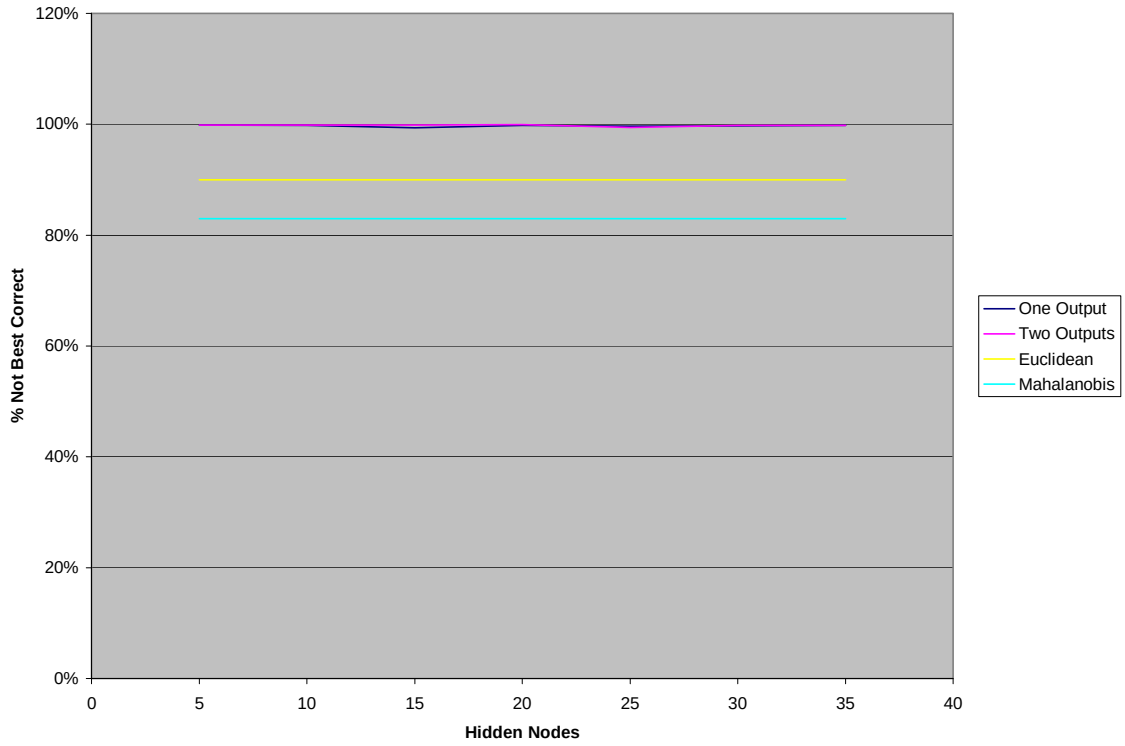

Figure 7.9: Classified Not Best – Unbalanced Training Set – Fuzzy Outputs.

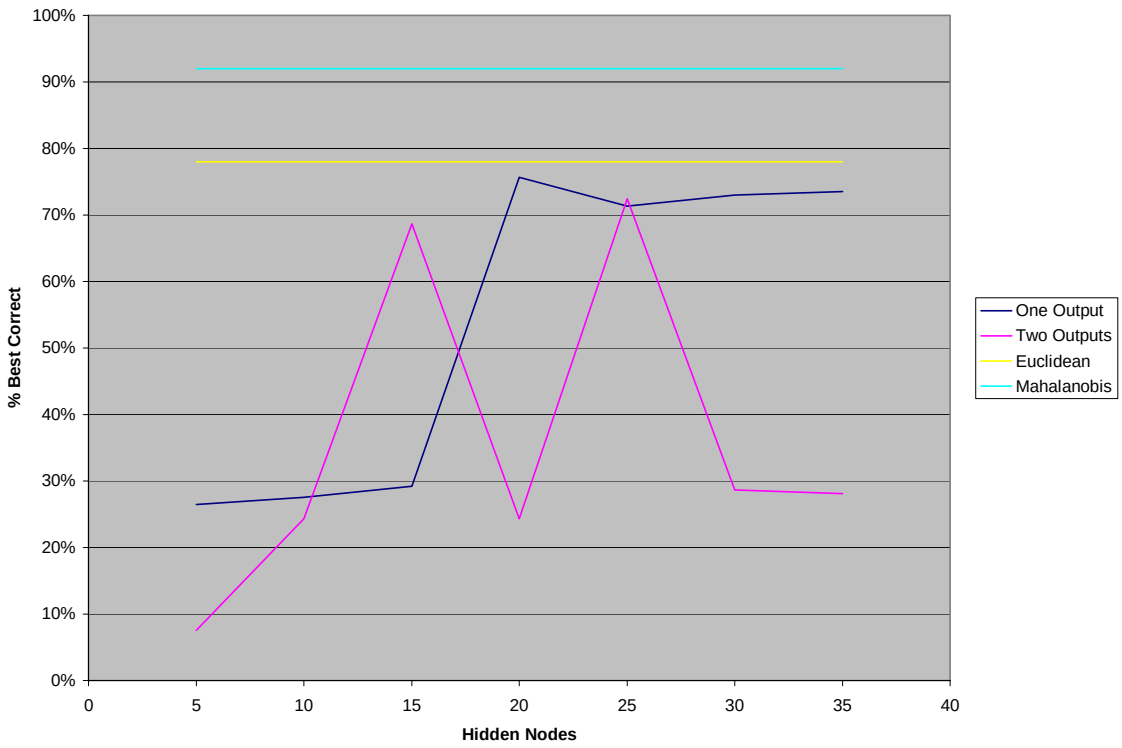

Figure 7.10: Classified Best – Unbalanced Training Set – Fuzzy Outputs.

Figure 7.9 shows that both fuzzy output classifiers classify Not Best practically 100% correct and outperform both Bayesian classifiers, or:

One Fuzzy Output = Two Fuzzy Outputs > Euclidean > Mahalanobis

However, Figure 7.10 illustrates that both Bayesian classifiers outperform both fuzzy output classifiers at classifying Best, and that the Mahalanobis classifier is the best. Or:

Mahalanobis > Euclidean > One Fuzzy Output > Two Fuzzy Outputs

Because of this, the results of these experiments are inconclusive.

Some type of ensemble of classifiers (e.g., Mahalanobis for classifying the Best and One Fuzzy Output for classifying the Not Best) is a possible area for further

study.

7.4 Comparison of Results with Fusion Weight Method

We need to calculate the FER for the Bayesian and neural network classifiers to compare them with the fusion algorithm used in the experiments. As explained earlier, FER is calculated as follows:

$$f = e / t$$

where:

f is the Fusion Error Rate (FER)

e is the number of incorrect Fusion Sessions

t is the total number of Fusion Sessions in the log

The number of incorrect Fusion Sessions, e, can be determined as follows:

$$e = x + y + z$$

where:

x is the number of Fusion Sessions where none of the Fusion choices were correct

y is the number of remaining Fusion Sessions where Best was classified as Not Best or Unknown

z is the number of remaining Fusion Sessions where Not Best was classified as Best

The total number of Fusion Sessions, t, for the test set (object slots from Log 7e) is 226. The number of Fusion Sessions where none of the Fusion choices were correct, x, for the test set (object slots from Log 7e) is 41. The number of remaining Fusion Sessions is then, 226 less 41 or 185. The number of remaining Fusion Sessions where Best was classified as Not Best or Unknown, y, can be taken directly from the confusion matrices in Table 7.1 and the Appendix.

The number of remaining Fusion Sessions where Not Best was classified as Best, z, can be determined as follows:

$$z = z_e(t - x - y)$$

where:

z_e = % of Not Best was classified as Best

$$z_e = z_1/(z_1 + z_2 + z_3)$$

where:

z_1 is the number of Not Best classified as Best

z_2 is the number of Not Best classified as Not Best

z_3 is the number of Not Best classified as Unknown

The number of Not Best classified as Best (z_1), Not Best (z_2), and Unknown (z_3) can be taken directly from the confusion matrices in Table 7.1 and the Appendix.

Table 7.10 shows the calculation of the FER for the Bayesian classifiers and the best neural network classifiers.

Classifier	x	y	z_1	z_2	z_3	z_e	z	f (FER)	Source
Euclidean	41	41	285	2684	0	9.6%	13.8	42%	Table 7.1
Mahalanobis	41	14	505	2464	0	17.0%	29.1	37%	Table 7.1
Unbalanced Training Set - One Fuzzy Output (h = 20)	41	45	5	2964	0	0.2%	0.2	38%	Appendix
Unbalanced Training Set - Two Fuzzy Outputs (h = 25)	41	51	16	2953	0	0.5%	0.7	41%	Appendix

Table 7.10: FER for Bayesian and Neural Network Classifiers.

Table 5.9 shows that the FER for the test set (object slots from Log 7e) using the fusion algorithm used in the experiments is 30%. This means that the fusion algorithm used in the experiments would outperform a fusion algorithm based on the following Bayesian or neural network classifiers:

- Two Bayesian classifiers, one using Euclidean distance and the other using Mahalanobis distance [2], to classify the test data based on the statistics of the training set.
- Neural network with one layer of hidden nodes and one output node.
- Neural network with one layer of hidden nodes and two output nodes.
- Neural network with a strict output constraint where the result is 0, if the output is less than 0.1; 1 if the output is greater than 0.9; and inconclusive otherwise.
- Neural network with a "fuzzy" output constraint where for one output, the result is 0, if the output is less than 0.5; 1 if the output is greater than 0.5; and inconclusive otherwise (i.e., the output is equal to 0.5); for two outputs, the result is 0, for the smaller output; 1 for the larger output; and inconclusive, if they are equal.

- Neural network training with an unbalanced training set.
- Neural network training with a small balanced training set consisting of all the Best samples plus an equal amount of randomly selected Not Best samples.
- Neural network training with a large balanced training set consisting of all the Not Best samples plus an equal amount of duplicated Best samples.

Chapter 8 Conclusions

8.1 Main Contributions

The hypothesis for this research is:

> **Applying the Human Computer Interaction (HCI) concepts of using multiple modalities, dialog management, context, and semantics to Human Robot Interaction (HRI) will improve the performance of Instruction Based Learning (IBL) compared to only using speech.**

Table 6.2 through Table 6.7 provide experimental evidence that the following combinations of modality and context do indeed improve HRI over speech by itself for all values of λ (i.e., ratio of importance of accuracy to succinctness) as hypothesized:

- Speech + Real World Context (one type of context)
- Speech + Dialog History (another type of context)
- Speech + Pointing (one type of modality)
- Speech + Field of Vision (another type of modality)
- Speech + Head Nodding (another type of modality)
- Speech + All (Real World Context, Dialog History, Pointing, Field of Vision, and Head Nodding)

Figure 5.14 shows that choosing the Fusion weights for Semantic Integration using the algorithm proposed in Chapter 5 improves HRI over using arbitrary Fusion weights.

The Fusion Error Rate (FER) results in Table 7.10 compared with those in Table 5.9 implies that the fusion algorithm used in the experiments for Semantic Integration would outperform a fusion algorithm based on the Bayesian or neural network classifiers that were tested.

Table 6.8 through Table 6.10 provide experimental evidence that the dialog management grounding mode that improves HRI the most is dependent on the value of λ as might be expected, with Optimistic grounding being the best for values of λ less than or equal to 0.8, and Pessimistic grounding being the best for values greater than 0.8. The experimental evidence also suggests that Cautious grounding is of limited use since it is never the best for any values of λ that were tested. The evidence also indicates that the Pareto principle (a.k.a., the 80-20 rule) applies to which grounding mode is the best [42].

8.2 Theoretical Contributions

In addition to the main contributions, the theoretical contributions of this dissertation include:

1. Codifying tasks as: action, object, tools, condition, and recursive subtasks. This method of codifying tasks is applicable to a wide variety of robot activities including household domestic chores, gardening, carpentry, assembly work, sorting trash for recycling, and exploration of other planets. It is also applicable to non-robotic applications such as smart homes (e.g., turn on the lights when the sun sets) or smart cars (e.g., take me to the store).

2. Using "peakedness" for determining k in K-Means Clustering. K-Means clustering is an unsupervised learning algorithm to classify a set of data into k clusters. A drawback of this algorithm is that k must be determined beforehand. We proposed determining k by calculating the "peakedness" for each k, and then using the one with the highest "peakedness" value. This method of determining k is applicable to any clustering application.
3. Developed and implemented an algorithm/heuristic for determining Fusion Weights. We chose a method of fusing the inputs from multiple modalities and contexts. The method multiplies the confidence score provided by the modalities and contexts for each input by a Fusion Weight, sums the products, and then uses the input with highest product sum. We proposed an algorithm for determining the Fusion Weights that is applicable to any robotic or non-robotic application (e.g., automated airline ticket agent) that uses this method of fusion.
4. Using the algorithm it was determined that the set of Fusion Weights closest to the centroid of the largest cluster using the k determined by peakedness produces the best average Fusion Error Rate (FER).
5. Developed an HRI measure (h_i) for comparing which combination of modalities, context, and dialog grounding is best for a desired level of learning accuracy and dialog succinctness. This measure is applicable to any HRI or HCI application that employs a dialog between a human and a system.

8.3 Application-Oriented Contributions

In addition to the main and theoretical contributions, the application-oriented contributions of this dissertation that can be applied to solve similar problems include:

1. Developed a set of Dialog Plans to manage Robot Learning, defining five plans to handle the entire dialog for Robot Learning: Initial Dialog Plan, Single Task Dialog Plan, Robot Learning Dialog Plan, Compound Task Dialog Plan, and Multiple Task Dialog Plan. The goal of a Dialog Plan is to translate a user utterance into a list of valid tasks. In pursuing its goal, a Dialog Plan can ask questions and receive additional utterances from the user and launch other Dialog Plans as sub-plans in achieving its goal. These Dialog Plans are applicable to any robotic or non-robotic application where a human is teaching a system to perform a task.
2. Developed a state table driven algorithm that allows the user to provide a list of sub-tasks using adverbs such as, “before”, “after”, and “next”. It can handle the adverb at the beginning of a sentence (e.g., “After loading the dishwasher, wipe down the sink.”) or in the middle (e.g., “Wipe down the sink after loading the dishwasher.”). This algorithm is applicable to any robotic or non-robotic application where a human is teaching a system to perform a task.
3. Implemented a dialog-grounding algorithm that supports Optimistic (verify no answers), Cautious (verify some answers), and Pessimistic (verify all answers) grounding. To verify what the user means, a Dialog Plan invokes the “check

assumption" method. If the dialog is in Optimistic mode, "check assumption" always returns true. If the dialog is in Pessimistic mode, "check assumption" will ask the user to verify the assumption with a "yes" or "no". If the dialog is in Cautious mode, it will ask the user to verify the assumption, if there is a possibility the assumption is not true. Cautious mode is flexible enough that the decision whether to verify or not can be fixed (as in these experiments) or adaptive (e.g., set by heuristics). For example, the robot may start out verifying every answer and then verify less as its confidence in understanding what the user is saying grows. This algorithm is applicable to any HRI or HCI application that employs a dialog between a human and a system.

4. Used already known tasks as Real World Context to resolve missing and ambiguous slot values. This method of developing a Real World Context is applicable to any robotic or non-robotic application where a human is teaching a system to perform a task.
5. Implemented an algorithm for searching Real World Context using wild cards and calculating confidence scores based on occurrence of values. The RealWorldResolver searches the Real World Context for tasks that match the partially filled single-task frame (treating unfilled slots as wildcards) and gets a list of values from the matching tasks for the specified slot. Then, it calculates a confidence score for each unique value in the list. This algorithm for using a Real World Context database is applicable to any robotic or non-robotic application where a human is teaching a system to perform a task.

6. Developed an algorithm for determining possible tool sets from Field of Vision. The FieldOfVisionResolver generates sets of tools by taking permutations of the Things in the robot's field of vision. Then, it provides a list of tool sets. To prevent an excessive number of tool sets, FieldOfVisionResolver prunes the list of tool sets. This algorithm is applicable to any computer vision system where the system needs to determine potential sets of things in the field of vision.
7. Developed XML-based Lesson Plans. The XML Lesson Plans are simple to edit and read and are portable across any computer language that supports XML. In these experiments, we used a tool, GrammarParser, to parse the Lesson Plans into Phoenix frame and grammar files. The HRI Environment Simulator used the same Lesson Plan files to extract the Things in a Room to simulate Pointing and Field of Vision and to provide the Lesson Plan text to guide the user in teaching the robot to perform the Tasks in the Lesson Plan. The Lesson Plan files were also used to extract the Real World Context database for the experiments and to extract a "golden" Task Database with which to compare the results. The XML-based Lesson Plan is applicable to any robotic or non-robotic application where a human is teaching a system to perform a task.
8. Developed a set of Phoenix parser frames. We designed eight frames that corresponds to the robot learning dialog between the user and the robot, namely, single-task, connector, learning, rbd, answer, quit, requestRepeat, and greet. These frames are applicable to any robotic or non-robotic Phoenix parser based application where a human is teaching a system to perform a task. They also

provide the general guidelines for using another parser in these types of applications.

8.4 Limitations

Although we were able to gather enough evidence to support the research hypothesis, there were some limitations:

1. Real vision system. We simulated Field of Vision, Pointing, and Head Nodding
2. Object and tool recognition. We assumed that the robot could assign a name (e.g., sponge) to the computer vision image of an object or tool.
3. Room recognition. We assumed the robot could assign a name (e.g., kitchen) to the computer vision image of a room.
4. Deeper parsing (e.g., recognition of verb, noun, and adverb) and larger vocabulary. We parsed the speech to fill slots of the eight-frame Robot Learning dialog.
5. Real or simulated robot task management, sensory, and mechanical systems. We only measured if the task taught by the user was learned by the robot in a symbolic sense, i.e., the robot correctly translated the speech, gestures, and dialog into a canonical task with an action, object, set of tools, and condition.
6. Real RbD learning. We simulated Robot Learning by Demonstration by saying, "I'll show you."
7. Real learning by example. We simulated learning by example by saying that two tasks are similar with words like, "is like" or "is similar to."

8. More sophisticated sub-task error recovery. In the current implementation, if the user detects an error in the sub-tasks of a Compound task, the only method of recovery is to start over again. A more sophisticated dialog, including user grounding (i.e., user says, "Where are we?") and sentences like, "Load the dishwasher first instead of wiping down the sink", would allow a more graceful recovery.
9. Different algorithm for Semantic Integration (e.g., ensemble of classifiers or adaptive Fusion weights). We chose a method of fusing the inputs from multiple modalities and contexts. The method multiplies the confidence score provided by the modalities and contexts for each input by a Fusion Weight, sums the products, and then uses the input with highest product sum.
10. Real speech synthesis. Although the Audio Feedback component does more than just display character strings, we simulated the sound of speech by displaying text on the HRI Environment Simulator screen.
11. Integrated speech recognition software. The integration of e-Speaking into the rest of the system was not as elegant as with Sphinx 4.0, but it provided keyboard-free input of speech into the system. A simple Java window program allowed us to use e-Speaking to speak words into the system.
12. More tasks. Our corpus included 166 tasks.

8.5 Future Work

Areas for further study include a real computer vision system, real or simulated robot task management, sensory, and mechanical systems, and different algorithms for Semantic Integration.

8.5.1 Real Computer Vision System

We simulated Field of Vision, Pointing, and Head Nodding in our experiments. Implementing a real computer vision system would allow us to explore object, tool, and room recognition (i.e., assigning a name to a computer vision image). In the COGNIRON project, this is called the home-tour, where a user shows a robot companion around his or her private home and teaches the robot important locations and objects, using speech and gestures [82].

Adding object, tool, and room recognition would require expanding the Speech Recognition component to recognize declarative (e.g., “That is a sponge”), and possibly interrogative (e.g., “Where is the sponge”) sentences. The current implementation only recognizes imperative sentences (e.g., clean the sink with a sponge). Adding declarative and interrogative sentences to the Speech Recognition component’s repertoire would require deeper parsing (e.g., recognition of verb, noun, and adverb) and a larger vocabulary.

Another implication of adding object and tool recognition is that the PointingResolver could be changed to accept any Thing as an object or tool, even if it is not in the RealWorldContext database. This is more reasonable with Pointing than Field of Vision, because Pointing is an active signal from the user, where Field of

Vision is a passive signal from the environment. Then, the robot could ask what the name is for the item the robot was pointing at and whether it was a tool, object, or both.

8.5.2 Task Management, Sensory, and Mechanical Systems

In a fully functioning robot, The Task Management component would break all tasks down into action primitives. The Robot Controller would take these primitives and execute them using the robot's sensory and mechanical systems.

We did not simulate the Task Management component because it was done by Blythe and Reilly who simulated a household robot agent, Mr. Fixit, who could, along with other tasks, vacuum and clean up broken cups [4]. We did not simulate the Robot Controller because it was beyond the scope of this dissertation.

Simulating or actually implementing these two components would allow us to implement real RbD learning (e.g., "I'll show you") and learning by example (e.g., "is like"). Implementing these two components would require expanding the Speech Recognition component to recognize declarative (e.g., "Here is how to wipe the sink") and possibly interrogative (e.g., "How do you wipe the sink with a sponge") sentences.

Implementing real RbD learning and learning by example would necessitate a more sophisticated sub-task error recovery. In the current implementation, if the user detects an error in the sub-tasks of a Compound task, the only method of recovery is to start over again. A more sophisticated dialog, including user grounding (i.e., user says, "Where are we?") and sentences like, "Load the dishwasher first instead of

wiping down the sink", would be required to support the collaborative dialog required to do real RbD learning and learning by example.

Implementing real RbD learning and learning by example and implementing or simulating the Task Management or Robot Controller components opens a completely new method of measuring task correctness by using standardized tests given to humans. This would allow a direct comparison between the skills of a human and those of a robot trying to learn human skills. This is also important because the best way to teach robots how to perform complex human tasks (e.g., carpentry, exploration, search and rescue, military) might be to send them to school with humans, meaning they will have to learn like a human. Fortunately, we will only have to teach one robot because we can "copy" what they learned to another robot. Of course, copying learned skills between heterogeneous robots opens up another area for further study.

Although actually implementing the sensory and mechanical systems of a Robot Controller sophisticated enough to perform household-cleaning tasks might be beyond current technology, simulating them probably is not. Avatars have been used to simulate robots [50, 48, 24] and simulation could be used to verify how well the robot learned a task.

8.5.3 Different Semantic Integration Algorithms

We chose one method of fusing the inputs from multiple modalities and contexts. The method multiplies the confidence score provided by the modalities and contexts

for each input by a Fusion Weight, sums the products, and then uses the input with highest product sum.

We also examined using neural networks for semantic integration. The results of these experiments indicated that hybrid classifiers (e.g., Mahalanobis for classifying the Best and One Fuzzy Output for classifying the Not Best) is an area for further study.

Other methods, such as adaptive Fusion weights are also areas for future work.

References

1. A. Alissandrakis, C. Nehaniv, and K. Dautenhahn 2007. Correspondence Mapping Induced State and Action Metrics for Robotic Imitation. *IEEE Transactions on Systems, Man, and Cybernetics – Part B*, 37(2), April 2007, 299 - 307.

2. E. Alpaydin 2004. *Introduction to Machine Learning*. The MIT Press, Cambridge, MA, USA.

3. G. Bekey, R. Ambrose, V. Kumar, A. Sanderson, B. Wilcox, and Y. Zheng 2006. Final Report. *World Technology Evaluation Center, Inc. (WTEC) Panel on International Assessment of Research and Development in Robotics*, January 2006.

4. J. Blythe and W. Reilly 1993. Integrating Reactive and Deliberative Planning in a Household Robot. *Carnegie Mellon University*, Pittsburgh, PA USA, November 1993.

5. R. Bolt 1980. Put that there: Voice and gesture at the graphics interface. In Proceedings of the *7th Annual Conference on Computer Graphics and Interactive Techniques* (Seattle, WA, USA, July 14-18, 1980), 262-270.

6. C. Breazeal 2004. cHRI Human-Robot Interaction from a Cognitive Viewpoint. In Proceedings of the *World Technology Evaluation Center, Inc. (WTEC) Workshop: Review of U.S. Research in Robotics* (National Science Foundation, Arlington, VA, USA, July 21-22, 2004).

7. C. Breazeal, A. Brooks, J. Gray, G. Hoffman, C. Kidd, H. Lee, J. Lieberman, A. Lockerd, and D. Chilongo 2004. Tutelage and Collaboration for Humaoid Robots. *International Journal of Humanoid Robots*, 1(2), 315-348.

8. C. Breazeal, D. Buchsbaum, J. Gray, D. Gatenby, and B. Blumberg 2005. Learning From and About Others: Towards Using Imitation to Bootstrap the Social Understanding of Others by Robots. *Artificial Life*, 11(1-2), 1-32.

9. C. Breazeal, C. Kidd, A. Thomaz, G. Hoffman, and M. Berlin 2005. Effects of nonverbal communication on efficiency and robustness in human-robot teamwork. In Proceedings of the *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Edmonton, Alberta, Canada, August 2-6, 2005), 708-713.

10. F. Brown, A. Agah, J. Gauch, T. Schreiber, and S. Speer 1999. A Cognitive Robot with Reconfigurable Mind for Studying Theories of Ambiguity Resolution. In Proceedings of the *IEEE International Conference on Systems, Man, and Cybernetics (SMC'99)* (Tokyo, Japan, October, 1999), 994-999.

11. F. Brown, A. Agah, J. Gauch, T. Schreiber, and S. Speer 1999. Ambiguity Resolution in Natural Language Understanding, Active Vision, Memory Retrieval, and Robot Reasoning and Actuation. In Proceedings of the *IEEE International Conference on Systems, Man, and Cybernetics (SMC'99)* (Tokyo, Japan, October, 1999), 988-993.

12. D. Bruemmer 2006. Behavior-Based Robotics. *Idaho National Laboratory*, Idaho Falls, ID, USA, May 30, 2006.

13. G. Bugmann 2003. Challenges in Verbal Instruction of Domestic Robots. In Proceedings of the *ASER '03 1st International Workshop on Advances in Service Robotics* (Bardolino, Italy, March 13-15, 2003), 112-116.

14. G. Bugmann, E. Klein, S. Lauria, J. Bos, and T. Kyriacou 2004. Corpus-Based Robotics: A Route Instruction Example. In Proceedings of the *8th International Conference on Intelligent Autonomous Systems (IAS-8)* (Amsterdam, Netherlands, March 10-13, 2004), 96-103.

15. G. Bugmann, S. Lauria, T. Kyriacou, E. Klein, J. Bos, and K. Coventry 2001. Using Verbal Instructions for Route Learning: Instruction Analysis. In Proceedings of the *TIMR 01 - Towards Intelligent Mobile Robots* (Manchester, UK, April 5, 2001), Technical Report Series, Department of Computer Science, Manchester University, ISSN 1361-6161, Report number UMC-01-4-1.

16. S. Calinon and A. Billard 2005. Recognition and Reproduction of Gestures using a Probabilistic Framework combining PCA, ICA and HMM. In Proceedings of the *22nd International Conference on Machine Learning* (Bonn, Germany, August 7-11, 2005), 105-112.

17. S. Calinon, F. Guenter, and A. Billard 2005. Goal-Directed Imitation in a Humanoid Robot 2005. In Proceedings of the *2005 IEEE International Conference on Robotics and Automation* (Barcelona, Spain, April 18-22, 2005), 299-304.

18. S. Calinon, F. Guenter, and A. Billard 2006. On Learning the Statistical Representation of a Task and Generalizing it to Various Contexts. In Proceedings of the *2006 IEEE International Conference on Robotics and Automation* (Orlando, FL, USA, May 15-19, 2006), 2978-2983.

19. S. Carbini, L. Delphin-Poulat, L. Perron, and J. E. Viallet 2006. From a Wizard of Oz Experiment to a Real Time Speech and Gesture Multi-modal Interface. *France Telecom R&D*, Lannion, France, 2006.

20. J. Chai, S. Pan, and M. Zhou 2005. MIND: A Context-based Multi-modal Interpretation Framework in Conversational Systems. In *Natural, Intelligent and Effective Interaction in Multimodal Dialogue Systems*, edited by O. Bernsen, L. Dybkjaer, and J. van Kuppevelt, Kluwer Academic Publishers, Norwell, MA, USA, 2005.

21. J. Chai, Z. Prasov, J. Blaim, and R. Jin 2005. Linguistic Theories in Efficient Multi-modal Reference Resolution: an Empirical Investigation. In Proceedings of the *10th International Conference on Intelligent User Interfaces (IUI-05)* (San Diego, CA, USA, January 9-12, 2005), 43-50.

22. P. Cohen and H. Levesque 1990. Persistence, intention, and commitment. Chapter 3 in *Intentions in Communication*, edited by P. Cohen, J. Morgan, and M. Pollack, MIT Press, Cambridge, MA, USA, 1990.

23. P. Cohen and H. Levesque 1991. Teamwork. *Nous*, Special Issue on Cognitive Science and AI, 25(4), 1991, 487-512.

24. A. Corradini, M. Mehta, and M. Charfuelan 2005. Interacting with an Animated Conversational Agent in a 3D Graphical Setting. In Proceedings of the *Workshop on Multi-modal Interaction for the Visualization and Exploration of scientific data at the Seventh International Conference on Multi-modal Interfaces (ICMI'05)* (Torento, Italy, October 3-6, 2005), 63-70.

25. D. Demirdjian, T. Ko, and T. Darrell 2005. Untethered Gesture Acquisition and Recognition for Virtual World Manipulation. *Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology*, Cambridge, MA, USA.

26. E. Den Os and L. Boves 2003. Towards Ambient Intelligence: Multi-modal Computers that Understand Our Intentions. *Max Planck Institute for Psycholinguistics*, Nijmegen, Netherlands and *Department of Language and Speech, University of Nijmegen,* Nijmegen, Netherlands, 2003.

27. F. DiSimoni 1978. The Token Test for Children. *DLM Teaching Resources*, Allen, TX, USA, 1978.

28. P. Dominey, M. Alvarez, B. Gao, M. Jeambrun, A. Cheylus, A. Weitzenfeld, A. Martinez, and A. Medrano 2005. Robot Command, Interrogation and Teaching via Social Interaction. In Proceeding of the *IEEE-RAS International Conference on Humanoid Robots* (Tsukuba, Japan, December 7, 2005), 475-480.

29. J. Drury, J. Scholtz, and H. Yanco 2004. Applying CSCW and HCI Techniques to Human-Robot Interaction. *The MITRE Corporation*, Bedford, MA, USA, *National Institute of Standards and Technology*, Gaithersburg, MD, USA, and *University of Massachusetts Lowell*, Lowell, MA, USA, 2004.

30. J. Drury, H. Yanco, W. Howell, B. Minten, and J. Casper 2006. Changing Shape: Improving Situation Awareness for a Polymorphic Robot. In Proceedings of the *2006 ACM Conference on Human-Robot Interaction* (Salt Lake City, UT, USA, March 2-4, 2006), 72-79.

31. J. Eisenstein and R. Davis 2004. Visual and Linguistic Information in Gesture Classification. In Proceedings of *2004 International Conference on Multi-modal Interfaces (ICMI'04)* (State College, PA, USA, October 13-15, 2004), 113-120.

32. R. Englert and G. Glass 2006. Architecture for Multi-modal Mobile Applications. *Deutsche Telekom Laboratories*, Berlin, Germany and *T-Systems International*, Berlin, Germany, 2006.

33. F. Flippo, A. Krebs, and I. Marsic 2003. A Framework for Rapid Development of Multi-modal Interfaces. In Proceedings of the *Fifth International Conference on Multimodal Interfaces (ICMI-PUI'03)* (Vancouver, Canada, November 5-7, 2003), 109-116.

34. F. Flippo 2003. A Natural Human-Computer Interface for Controlling Wheeled Robotic Vehicles. *Delft University of Technology, Department of Information Technology and Systems*, Delft, Netherlands, August 2003.

35. P. Gorniak and D. Roy 2005. Probabilistic Grounding of Situated Speech using Plan Recognition and Reference Resolution. In Proceedings of the *Seventh International Conference on Multi-modal Interfaces (ICMI'05)* (Torento, Italy, October 3-6, 2005), 138-143.

36. A. Green and K. Severinson-Eklundh 2001. Task-oriented Dialogue for CERO: a User-centered Approach. *Royal Institute of Technology (KTH*), Stockholm, Sweden, 2001.

37. A. Green, K. Severinson-Eklundh, B. Wrede, and S. Li 2006. Integrating Miscommunication Analysis in Natural Language Interface Design for a Service Robot. In Proceedings of the *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems* (Beijing, China, October 9-15, 2006), 4678-4683.

38. S. Harnad 1990. The Symbol grounding problem. *Physica D*, 42, 1990, 335-346, 1990.

39. P. Jensfelt, S. Ekvall, D. Kragic and D. Aarno 2005. Integrating SLAM and Object Detection for Service Robot Tasks. In Proceedings of the *IEEE/RSJ International Conference on Intelligent Robots and Systems, Workshop on Mobile Manipulators: Basic Techniques, New Trends and Applications* (Edmonton, Canada, August 2-6, 2005).

40. D. O. Johnson and A. Agah 2009. Human Robot Interaction Through Semantic Integration of Multiple Modalities, Dialog Management, and Contexts. *International Journal of Social Robotics*, in review.

41. D. O. Johnson and A. Agah 2009. Human Robot Interaction Through Semantic Integration of Real World Context and Dialog History. In Proceedings of the *ACM/IEEE International Conference on Human-Robot Interaction* (San Diego, CA, USA, March 11-13, 2009), in review.

42. J. Juran and A. Godfrey 1999. *Juran's Quality Handbook, Fifth Edition*. McGraw-Hill, New York, USA, 1999.

43. S. Kettebekov and R. Sharma 2001. Toward Natural Gesture/Speech Control of a Large Display. In Proceedings of the *8th IFIP International Conference on Engineering for Human-Computer Interaction (EHCI'01)* (Toronto, Ontario, Canada, May 11-13, 2001), 221-234.

44. O. Khatib, M. Peshkin, and Y. Matsuoka 2004. pHRI – physical Human-Robot Interaction. In Proceedings of the *World Technology Evaluation Center, Inc. (WTEC) Workshop: Review of U.S. Research in Robotics* (National Science Foundation, Arlington, VA, USA, July 21-22, 2004).

45. S. Kiesler and P. Hinds 2004. Introduction to this Special Issue on Human-Robot Interaction. *Human Computer Interaction,* 19(1-2), 2004, 1-8.

46. V. Klingspor, J. Demiris, and M.Kaiser 1997. Human-Robot-Communication and Machine Learning. *Universität Dortmund*, Dortmund, Germany, *University of Edinburgh*, Edinburgh, UK, and *Universität Karlsruhe*, Karlsruhe, Germany, 1997.

47. R. Kohavi 1995. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In Proceedings of the *International Joint Conference on Artificial Intelligence (IJCAI)* (Montreal, Quebec, Canada, August 20-25, 1995), 1137-1143.

48. S. Kopp, L. Gesellensetter, N. Krämer, and I. Wachsmuth 2004. A conversational agent as museum guide -- design and evaluation of a real-world application. In Proceedings of the *Third Hellenic Conference on Artificial Intelligence, SETN 2004* (Samos, Greece, May 5-8, 2004), 329-343.

49. D. Kosmopoulos and I. Maglogiannis 2006. Extraction of mid-level semantics from gesture videos using a Bayesian network. *International Journal of Intelligent Systems Technologies and Applications*, 1(3/4), 2006, 359-375.

50. A. Kranstedt, and I. Wachsmuth 2004. Situated Generation of Multi-modal Deixis in Task-Oriented Dialogue. *Artificial Intelligence Group, Faculty of Technology, University of Bielefeld,* Bielefeld, Germany, June 2004.

51. S. Lauria, G. Bugmann, T. Kyriacou, and E. Klein 2001. Instruction Based Learning: How to Instruct a Personal Robot to Find HAL. In Proceedings of the *9th European Workshop on Learning Robots, EWLR-9* (Prague, Czech Republic, September 8-9, 2001), 74-83.

52. A. Lockerd and C. Breazeal 2004. Tutelage and Socially Guided Robot Learning. *MIT Media Lab*, Cambridge, MA, USA, 2004.

53. L. Lopes and A. Teixeira 2000. Human-Robot Interaction through Spoken Language Dialogue. In Proceedings of the *2000 IEEE/RSJ International Conference on Intelligent Robots and Systems* (Takamatsu, Japan, October 30 - November 5, 2000), 528-534.

54. L. Lopes and Q. Wang 2002. Towards Grounded Human-Robot Communication. In Proceedings of the *IEEE International Workshop on Robot and Human Interactive Communication (ROMAN'2002)* (Berlin, Germany, September 25-27, 2002), 312-318.

55. M. MacMahon 2005. MARCO: A Modular Architecture for Following Route Instructions. *Department of Electrical and Computer Engineering, Intelligent Robotics Laboratory, University of Texas at Austin*, Austin, TX, USA, 2005.

56. J. MacQueen 1967. Some Methods for Classification and Analysis of Multivariate Observations. *5th Berkeley Symposium on Mathematical Statistics and Probability*, University of California Press, Berkley, CA, USA, 1, 281-297.

57. P. Maragos 2004. MUSCLE Network of Excellence, Work Package 6: Cross-Modal, Integration for Performance Improving in Multimedia, Deliverable 1: Report on the State-of-the-Art. *National Technical University of Athens*, Greece, September 2004.

58. N. Mavridis and D. Roy 2005. Grounded Situation Models for Robots: Bridging language, Perception, and Action. In Proceedings of the *AAAI-05 Workshop on Modular Construction of Human-Like Intelligence* (Pittsburgh, PA, USA, July 10, 2005), 32-39.

59. F. Metze, P. Gieselmann, H. Holzapfel, T. Kluge, I. Rogina, A. Waibel, M. Wölfel, J. Crowley, P. Reignier, and D. Vaufreydaz, F. Bérard, B. Cohen, J. Coutaz, S. Rouillard, V. Arranz, M. Bertrán, and H. Rodriguez 2005. The "FAME" Interactive Space. In Proceedings of the *2nd Joint Workshop on Multi-modal Interaction and Related Machine Learning Algorithms* (Edinburgh, UK, July 11-13, 2005), 4.

60. L. Morency, and T. Darrell 2006. Head Gesture Recognition in Intelligent Interfaces: The Role of Context in Improving Recognition. In Proceedings of the *International Conference on Intelligent User Interfaces* (Sydney, Australia, January 29 - February 1, 2006), 32-38.

61. L. Morency, C. Sidner, and T. Darrell 2005. Towards Context-Based Visual Feedback Recognition for Embodied Agents. Proceedings of the *Symposium on Conversational Informatics for Supporting Social Intelligence and Interaction (AISB'05)* (Hatfield, England, April 12-15, 2005), 69-72.

62. K. Nickel and R. Stiefelhagen 2007. Visual Recognition of Pointing Gestures for Human-Robot Interaction. *Image and Vision Computing*, 25(12), December 2007, 1875-1884.

63. M. Nicolescu and M. Mataric 2001. Learning and Interacting in Human-Robot Domains. *Special Issue of IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 31(5), September 2001, 419-430.

64. M. Nicolescu and M. Mataric 2005. Task Learning Through Imitation and Human-Robot Interaction. In *Models and Mechanisms of Imitation and Social*

Learning in Robots, Humans and Animals, edited by K. Dautenhahn and C. Nehaniv, Cambridge University Press, Cambridge, UK, 2005.

65. D. Olsen and M. Goodrich 2003. Metrics for Evaluating Human-Robot Interactions. *Computer Science Department, Brigham Young University*, Provo, UT, USA, 2003.

66. K. Pastra 2004. Viewing Vision-Language Integration as a Double-Grounding Case. In Proceedings of the *American Association of Artificial Intelligence Fall Symposium Series* (Arlington, VA, USA, October 22-24, 2004), 62-67.

67. A. Quattoni, M. Collins, and T. Darrell 2005. Incorporating Semantic Constraints into a Discriminative Categorization and Labeling Model. In Proceedings of the *Workshop on Semantic Knowledge in Vision, Tenth IEEE International Conference on Computer Vision* (Beijing, China, October 15-21, 2005).

68. R. Ravid 2005. *Practical Statistics for Educators, Third Edition*. University Press of America, Lanham, MD, USA, 129-131.

69. C. Robbins 2004. Speech and Gesture Based Multi-modal Interface Design. *Computer Science Department, New York University*, New York City, NY, USA, April 2004.

70. M. Rosenstein and A. Barto 2004. Supervised ActorCritic Reinforcement Learning. *Department of Computer Science, University of Massachusetts,* Amherst, MA, USA, 2004.

71. D. Roy 2005. Semiotic Schemas: A Framework for Grounding Language in Action and Perception. *Artificial Intelligence*, 167(1-2), 2005, 170-205.

72. K. Saenko, K. Livescu, M. Siracusa, K. Wilson, J. Glass, and T. Darrell 2005. Visual Speech Recognition with Loosely Synchronized Feature Streams. In Proceedings of the *Tenth IEEE International Conference on Computer Vision* (Beijing, China, October 15-21, 2005), 1424-1431.

73. J. Searle 1983. Intentionality: an essay in the philosophy of mind. *Cambridge University Press*, Cambridge, UK, 1983.

74. J. Searle 1969. Speech acts: an essay in the philosophy of language. *Cambridge University Press*, Cambridge, UK, 1969.

75. R. Sharma, M. Yeasin, N. Krahnstoever, I. Rauschert, G. Cai, I. Brewer, A. Maceachren, and K. Sengupta 2003. Speech-Gesture Driven Multi-modal Interfaces for Crisis Management. *Proceedings of the IEEE*, 9, September 2003, 1327-1354.

76. C. Sidner and C. Lee, and C. Kidd 2005. Engagement During Dialogues with Robots. In Proceedings of the *2005 AAAI Spring Symposium Series* (Stanford, CA, USA, March 21-23, 2005).

77. C. Sidner, C. Lee, L. Morency, and C. Forlines 2006. The Effect of Head-Nod Recognition in Human-Robot Conversation. In Proceedings of the *2006 ACM Conference on Human-Robot Interaction* (Salt Lake City, UT, USA, March 2-4, 2006), 290-296.

78. J. Suomela and A. Halme 2004. Human Robot Interaction – Case WorkPartner. In Proceedings of the *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2004* (Sendai, Japan, September 28 - October 2, 2004), 3327-3332.

79. L. Taycher, G. Shakhnarovich, D. Demirdjian, and T. Darrell 2006. Conditional Random People: Tracking Humans with {CRF}s and Grid Filters. In Proceedings of the *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)* (New York, NY, USA, June 17-22, 2006), 222-229.

80. S. Tellex and D. Roy 2006. Spatial Routines for a Simulated Speech-Controlled Vehicle. In Proceedings of the *2006 ACM Conference on Human-Robot Interaction* (Salt Lake City, UT, USA, March 2-4, 2006), 156-163.

81. A. Thomaz, G. Hoffman, and C. Breazeal 2006. Experiments in Socially Guided Machine Learning: Understanding How Humans Teach. In Proceedings of the *2006 ACM Conference on Human-Robot Interaction* (Salt Lake City, UT, USA, March 2-4, 2006), 359-360.

82. I. Toptsis, S. Li, B. Wrede, G. Fink 2004. A Multi-modal Dialog System for a Mobile Robot. *Faculty of Technology, Bielefeld University*, Bielefeld, Germany, 2004.

83. D. Traum, J. Bos, R. Cooper, S. Larsson, I. Lewin, C. Matheson, and M. Poesio 1999. A model of dialog moves and information state revision. Trindi Report D2.1, *Department of Linguistics, Göteborg University,* Gothenburg, Sweden, November 1999.

84. L. Vygotsky 1978. *Mind in Society: The Development of Higher Psychological Processes*. Harvard University Press, Cambridge, MA, USA.

85. W. Walker, P. Lamere, P. Kwok, B. Raj, R. Singh, E. Gouvea, P. Wolf, J. Woelfel 2004. Sphinx-4: A Flexible Open Source Framework for Speech Recognition. Technical Report, SMLI TR-2004-139, *Sun Microsystems*, Menlo Park, CA, USA, November 2004.

86. S. Wang and D. Demirdjian 2005. Inferring Body Pose using Speech Content. In Proceedings of the *International Conference on Multi-modal Interfaces* (Trento, Italy, October 3-7, 2005), 53-60.

87. N. Webb, C. Ursu, Y. Wilks, H. Hardy, M. Wu, and T. Strzalkowski 2005. Data-Driven Language Understanding for Spoken Language Dialog. In Proceedings of the *AAAI Workshop on Spoken Language Understanding* (Pittsburgh, PA, USA, July 9, 2005).

88. M. Weik 1961. The ENIAC Story. *O R D N A N C E, The Journal of the American Ordnance Association*, January-February 1961.

89. S. Wilske and G. Kruijff 2006. Service Robots dealing with indirect speech acts. *Language Technology Lab, German Research Center for Artificial Intelligence (DFKI)*, Saarbrucken, Germany, 2006.

90. J. Wolf and G. Bugmann 2005. Multi-modal Corpus Collection for the Design of User-Programmable Robots. Proc. *Towards Autonomous Robotics Systems 2005 (Taros'05)* (London, UK, September 12-14, 2005), 251-255.

91. H. Yanco, M. Baker, R. Casey, B. Keyes, P. Thoren, J. Drury, D. Few, C. Nielsen, and D. Bruemmer 2006. Analysis of Human-Robot Interaction for Urban Search and Rescue. *University of Massachusetts Lowell*, Lowell, MA, USA, *The MITRE Corporation*, Bedford, MA, USA, and *Idaho National Laboratories*, Idaho Falls, ID, USA, 2006.

92. H. Yanco and J. Drury 2004. Classifying Human-Robot Interaction: An Updated Taxonomy. In Proceedings of the *IEEE SMC 2004 International Conference on Systems, Man and Cybernetics* (The Hague, Netherlands, October 10-13, 2004), 2841-2846.

93. J. Zhang and A. Knoll 2003. A Two-Arm Situated Artificial Communicator for Human-Robot Cooperative Assembly. *IEEE Transactions on Industrial Electronics*, 50(4), August 2003, 651- 658.

94. *Dragon NaturallySpeaking 9 User's Guide*. Nuance Communications Inc., Burlington, MA, USA, Version 9.5, January 2007.

95. *The Phoenix Parser User Manual*. The Center for Spoken Language Research, University of Colorado, Boulder, CO, USA. March 2002.

96. *Google Scholar* Home Page. http://scholar.google.com.

97. *Microsoft Speech SDK Version 5.1* Home Page. http://msdn.microsoft.com/en-us/library/ms990097.aspx.

98. *Microsoft Windows XP* Home Page. www.microsoft.com/windows/windows-xp/default.aspx.

99. On-Line *e-Speaking User's Guide*. http://www.e-speaking.com/support.htm.

100. *SourceForge.net* Home Page. http://sourceforge.net.

101. *Speech for Java*. http://www.alphaworks.ibm.com/tech/speech.

102. *TRINDIKIT* Home Page. http://www.ling.gu.se/projekt/trindi/trindikit.

Appendix I – Lesson Plans

The Lesson Plan files are in XML format. The XML tags are explained below:

<lesson plan> Denotes the beginning and end of a Lesson Plan.

<text> Denotes the beginning and end of the Text portion of the Lesson Plan. The Text portion is divided into four sections:

- Task: This is the main task being taught (e.g., clean the kitchen daily).
- Steps: These are the sub-tasks of the main task (e.g., wipe down the sink after loading the dishwasher).
- Answers: These are the answers to the questions that the Robot might ask about the missing slots in the Steps (e.g., "a sponge" is the answer to the question, "What tools do you want me to use to WIPE DOWN the SINK?")
- How To: These are the Teaching Methods the user uses to teach the Robot how to perform a specific task or sub-task (e.g., RbD, Is Like, or Nested Sub-Tasks)

<room> Denotes which room is the "object" of the main task (e.g., kitchen).

<tasks> Denotes the beginning and the end of list of tasks which compose the main task.

<task> Denotes the beginning and the end of a specific task. The task is in the following format:

action,object,[tools],condition,[sub-tasks];

where:

tools = tool,tool,...,tool
sub-tasks = task task ... task
action = string of characters
object = string of characters
tool = string of characters
condition = string of characters

Lesson Plan 1

```
<lessonplan>
<text>
Task: Clean the kitchen daily

Steps:
Wipe down the sink after loading the dishwasher
Wipe down the stove top
Wipe down the counters
Vacuum the floor

Answers:
Load the dishwasher with nothing
Wipe down the sink with a sponge
Wipe down the stove top with a sponge
Wipe down the counters with a sponge
Vacuum the floor with a sweeper

How To:
Load the dishwasher - I'll show you
Wipe down the sink - I'll show you
Wipe down the stove top is similar to wipe down the sink
Wipe down the counters is similar to wipe down the sink
Vacuum the floor - I'll show you
</text>
<tasks>
<room>
KITCHEN
</room>
<task>
CLEAN,KITCHEN,[],DAILY,[
 LOADING,DISHWASHER,[],DAILY,[];
 WIPE DOWN,SINK,[SPONGE],DAILY,[];
 WIPE DOWN,STOVE TOP,[SPONGE],DAILY,[];
 WIPE DOWN,COUNTERS,[SPONGE],DAILY,[];
 VACUUM,FLOOR,[SWEEPER],DAILY,[];
];
</task>
</tasks>
</lessonplan>
```

Lesson Plan 2

```
<lessonplan>
<text>
Task: Clean the bathroom daily

Steps:
Wipe out the sink
Wipe the toilet seat and rim
Swoosh the toilet bowl with a brush
Wipe the mirror and faucet
Squeegee the shower door
Spray the entire shower and the curtain liner with shower mist after
every use

Answers:
Wipe out the sink with a sponge and a dry rag
Wipe the toilet seat with a sponge and a dry rag
Wipe the toilet rim with a sponge and a dry rag
Wipe the mirror with a sponge and a dry rag
Wipe the faucet with a sponge and a dry rag
Squeegee the shower door with a dry rag

How To:
Wipe out the sink - I'll show you
Wipe the toilet seat is similar to wipe out the sink
Wipe the toilet rim is similar to wipe out the sink
Swoosh the toilet bowl - I'll show you
Wipe the mirror is similar to wipe out the sink
Wipe the faucet is similar to wipe out the sink
Squeegee the shower door - I'll show you
Spray the entire shower - I'll show you
Spray the curtain liner is similar to spray the shower
</text>
<tasks>
<room>
BATHROOM
</room>
<task>
CLEAN,BATHROOM,[],DAILY,[
 WIPE OUT,SINK,[SPONGE,DRY RAG],DAILY,[];
 WIPE,TOILET SEAT,[SPONGE,DRY RAG],DAILY,[];
 WIPE,RIM,[SPONGE,DRY RAG],DAILY,[];
 SWOOSH,TOILET BOWL,[BRUSH],DAILY,[];
 WIPE,MIRROR,[SPONGE,DRY RAG],DAILY,[];
 WIPE,FAUCET,[SPONGE,DRY RAG],DAILY,[];
 SQUEEGEE,SHOWER DOOR,[DRY RAG],DAILY,[];
 SPRAY,SHOWER,[SHOWER MIST],AFTER EVERY USE,[];
 SPRAY,CURTAIN LINER,[SHOWER MIST],AFTER EVERY USE,[];
];
</task>
</tasks>
</lessonplan>
```

Lesson Plan 3

```
<lessonplan>
<text>
Task: Clean the bedroom daily

Steps:
Make the bed
Fold or hang clothing and put away jewelry
Straighten out the night-table surface

Answers:
Make the bed with no tools
Fold or hang clothing with no tools
Put away jewelry with no tools
Straighten out the night-table with nothing

How To:
Make the bed - I'll show you
Fold or hang clothing - I'll show you
Put away jewelry is similar to fold or hang clothing
Straighten out the night-table is similar to fold or hang clothing
</text>
<tasks>
<room>
BEDROOM
</room>
<task>
CLEAN,BEDROOM,[],DAILY,[
 MAKE,BED,[],DAILY,[];
 FOLD OR HANG,CLOTHING,[],DAILY,[];
 PUT AWAY,JEWELRY,[],DAILY,[];
 STRAIGHTEN OUT,NIGHT TABLE SURFACE,[],DAILY,[];
];
</task>
</tasks>
</lessonplan>
```

Lesson Plan 4

```
<lessonplan>
<text>
Task: Clean the family room daily
Task: Clean the living room daily
Task: Clean the foyer daily

Steps (same for all three rooms):
Pick up crumbs and dust bunnies with a handheld vacuum
Fluff the cushions and fold throws after use
Wipe tabletops and spot-clean cabinets when you see fingerprints
Straighten coffee-table books and magazines.
Throw out newspapers.
Put away CDs and videos.

Answers:
Fluff the cushions with your hands
Fold throws with you hands
Wipe tabletops with pledge and a dry rag
Spot clean cabinets with pledge and a dry rag
Straighten coffee table with your hands
Throw out newspapers with your hands
Put away CDs with your hands
Put away videos with your hands

How To:
Pick up crumbs and dust bunnies - I'll show you
Fluff the cushions - I'll show you
Fold throws - I'll show you
Wipe tabletops - I'll show you
Spot clean cabinets is similar to wipe tabletops
Straighten coffee table - I'll show you
Throw out newspapers is similar to straighten coffee table
Put away CDs with your hands - I'll show you
Put away videos with your hands is similar to put away CDs
Clean the living room is similar to clean the family room
Clean the foyer is similar to clean the family room
</text>
<tasks>
<room>
FAMILY ROOM, LIVING ROOM, FOYER
</room>
<task>
CLEAN,FAMILY ROOM,[],DAILY,[
 PICK UP,CRUMBS AND DUST BUNNIES,[HANDHELD VACUUM],DAILY,[];
 FLUFF,CUSHIONS,[HANDS],DAILY,[];
 FOLD,THROWS,[HANDS],DAILY,[];
 WIPE,TABLETOPS,[PLEDGE,DRY RAG],DAILY,[];
 SPOT CLEAN,CABINETS,[PLEDGE,DRY RAG],DAILY,[];
 STRAIGHTEN,COFFEE TABLE,[HANDS],DAILY,[];
 THROW OUT,NEWSPAPERS,[HANDS],DAILY,[];
 PUT AWAY,CDS,[HANDS],DAILY,[];
 PUT AWAY,VIDEOS,[HANDS],DAILY,[];
```

```
];
</task>
<task>
CLEAN,LIVING ROOM,[],DAILY,[
 PICK UP,CRUMBS AND DUST BUNNIES,[HANDHELD VACUUM],DAILY,[];
 FLUFF,CUSHIONS,[HANDS],DAILY,[];
 FOLD,THROWS,[HANDS],DAILY,[];
 WIPE,TABLETOPS,[PLEDGE,DRY RAG],DAILY,[];
 SPOT CLEAN,CABINETS,[PLEDGE,DRY RAG],DAILY,[];
 STRAIGHTEN,COFFEE TABLE,[HANDS],DAILY,[];
 THROW OUT,NEWSPAPERS,[HANDS],DAILY,[];
 PUT AWAY,CDS,[HANDS],DAILY,[];
 PUT AWAY,VIDEOS,[HANDS],DAILY,[];
];
</task>
<task>
CLEAN,FOYER,[],DAILY,[
 PICK UP,CRUMBS AND DUST BUNNIES,[HANDHELD VACUUM],DAILY,[];
 FLUFF,CUSHIONS,[HANDS],DAILY,[];
 FOLD,THROWS,[HANDS],DAILY,[];
 WIPE,TABLETOPS,[PLEDGE,DRY RAG],DAILY,[];
 SPOT CLEAN,CABINETS,[PLEDGE,DRY RAG],DAILY,[];
 STRAIGHTEN,COFFEE TABLE,[HANDS],DAILY,[];
 THROW OUT,NEWSPAPERS,[HANDS],DAILY,[];
 PUT AWAY,CDS,[HANDS],DAILY,[];
 PUT AWAY,VIDEOS,[HANDS],DAILY,[];
];
</task>
</tasks>
</lessonplan>
```

Lesson Plan 5

<lessonplan>
<text>
Task: Clean the kitchen in the spring

Steps:
Begin with a 15 Minute Kitchen Cleanup
Dust down the ceiling and corners of walls
Dust and clean all art and photographs along the wall
Dust and clean the ceiling fan
Take down draperies, curtains, and blinds to wash or have cleaned according to the manufacturer's directions
Clean the oven with oven cleaner
Clean the refrigerator
Clean the stove
Wipe down and clean the toaster, blender, and other small appliances.
Run the dishwasher empty
Wash down the countertops in your kitchen
Wipe down and clean out any drawers
Wash down the sink
Sweep and mop the floors

Answers:
Cleanup the kitchen with no tools
Dust down the ceiling with a feather duster
Dust down the corners of walls with a feather duster
Dust and clean art with a lightly wet clean cloth
Dust and clean photographs with a lightly wet clean cloth
Dust and clean the ceiling fan with Murphys Oil Soap
Take down draperies with your hands
Take down curtains with your hands
Take down blinds with your hands
Clean the refrigerator with damp cloth
Clean the stove with your damp cloth
Wipe down and clean the toaster with a damp cloth
Wipe down and clean the blender with a damp cloth
Wipe down and clean the other small appliances with a damp cloth
Run the dishwasher with vinegar
Wash down the countertops with Windex
Wipe down and clean out drawers with a damp cloth
Wash down the sink with a sponge
Sweep the floors with a broom
Mop the floors with a towel

How To:
Begin with a 15 Minute Kitchen Cleanup:
- Scrape off all the dishes into the trash or garbage disposal with a sponge
- Load the dishes into the dishwasher with your hands
- Pick up trash with your hands
- Put away all items with your hands
- Take out the trash with your hands

```
Dust down the ceiling - I'll show you
Dust down the corners of walls is similar to dust down the ceiling
Dust and clean art - I'll show you
Dust and clean photographs is similar to dust and clean art
Dust and clean the ceiling fan - I'll show you
Take down draperies  - I'll show you
Take down curtains is similar to take down draperies
Take down blinds is similar to take down draperies
Clean the oven - I'll show you
Clean the refrigerator - I'll show you
Clean the stove is similar to clean the refrigerator
Wipe down and clean the toaster - I'll show you
Wipe down and clean the blender is similar to wipe down and clean
the toaster
Wipe down and clean the other small appliances is similar to wipe
down and clean the toaster
Run the dishwasher - I'll show you
Wash down the countertops - I'll show you
Wipe down and clean out drawers is similar to wipe down and clean
the blender
Wash down the sink - I'll show you
Sweep the floors - I'll show you
Mop the floors - I'll show you
</text>
<tasks>
<room>
KITCHEN
</room>
<task>
CLEAN,KITCHEN,[],SPRING,[
 CLEANUP,KITCHEN,[],SPRING,[
  SCRAPE OFF,DISHES,[SPONGE],SPRING,[];
  LOAD,DISHWASHER,[HANDS],SPRING,[];
  PICK UP,TRASH,[HANDS],SPRING,[];
  PUT AWAY,ITEMS,[HANDS],SPRING,[];
  TAKE OUT,TRASH,[HANDS],SPRING,[];
 ];
 DUST DOWN,CEILING,[FEATHER DUSTER],SPRING,[];
 DUST DOWN,CORNERS OF WALLS,[FEATHER DUSTER],SPRING,[];
 DUST AND CLEAN,ART,[LIGHTLY WET CLEAN CLOTH],SPRING,[];
 DUST AND CLEAN,PHOTOGRAPHS,[LIGHTLY WET CLEAN CLOTH],SPRING,[];
 DUST AND CLEAN,CEILING FAN,[MURPHYS OIL SOAP],SPRING,[];
 TAKE DOWN,DRAPERIES,[HANDS],SPRING,[];
 TAKE DOWN,CURTAINS,[HANDS],SPRING,[];
 TAKE DOWN,BLINDS,[HANDS],SPRING,[];
 CLEAN,OVEN,[OVEN CLEANER],SPRING,[];
 CLEAN,REFRIGERATOR,[DAMP CLOTH],SPRING,[];
 CLEAN,STOVE,[DAMP CLOTH],SPRING,[];
 WIPE DOWN AND CLEAN,TOASTER,[DAMP CLOTH],SPRING,[];
 WIPE DOWN AND CLEAN,BLENDER,[DAMP CLOTH],SPRING,[];
 WIPE DOWN AND CLEAN,OTHER SMALL APPLIANCES,[DAMP CLOTH],SPRING,[];
 RUN,DISHWASHER,[VINEGAR],SPRING,[];
 WASH DOWN,COUNTERTOPS,[WINDEX],SPRING,[];
 WIPE DOWN AND CLEAN,DRAWERS,[DAMP CLOTH],SPRING,[];
```

```
 WASH DOWN,SINK,[SPONGE],SPRING,[];
 SWEEP,FLOORS,[BROOM],SPRING,[];
 MOP,FLOORS,[TOWEL],SPRING,[];
];
</task>
</tasks>
</lessonplan>
```

Lesson Plan 6

<lessonplan>
<text>
Task: Clean the bathroom in the spring

Steps:
Follow the 15 Minute Bathroom Cleanup
Dust down the ceiling and corners
Dust the vents and fans.
Take down draperies, curtains, blinds, etc, to wash or have cleaned according to the directions
Scrub the shower and tub
Scrub down the toilet
Wash the inside and outside of cabinets
Wash down the sink and fixtures
Shake out bathroom rugs
Sweep and mop the floors
Empty and wash out the trash can

Answers:
Follow the 15 Minute Bathroom Cleanup with no tools
Dust down the ceiling and corners with a feather duster
Dust the vents and fans with a gentle cleanser mixed with water and a cleaning cloth
Take down draperies, curtains, and blinds with your hands
Scrub the shower and tub with a commercial cleaner
Scrub down the toilet with a commercial cleaner
Wash the inside and outside of cabinets with a damp sponge
Wash down the sink and fixtures with a damp sponge
Shake out bathroom rugs with your hands
Sweep the floors with a broom
Mop the floors with a towel
Empty the trash can with your hands
Wash out the trash can with windex and a sponge

How To:
Follow the 15 Minute Bathroom Cleanup:
- Grab and put dirty clothing in the hamper with your hands
- Grab and put trash in the trash can with your hands
- Wipe down the sink and tub with a disinfectant wipe
- Scrub out inside of toilet with toilet brush and toilet cleaner
- Using a disinfectant wipe, wipe down the outside of the toilet
- Using glass cleaner, wipe down the mirror
- Pickup and put away items with your hands
- Briefly sweep the floor with a Swiffer
Dust down the ceiling - I'll show you
Dust down the corners is similar to dust down the ceiling
Dust the vents - I'll show you
Dust fans is similar to dust vents
Take down draperies - I'll show you
Take down curtains is similar to take down draperies
Take down blinds is similar to take down draperies
Scrub the shower - I'll show you

```
Scrub the tub is similar to scrub the shower
Scrub down the toilet is similar to scrub the shower
Wash the cabinets - I'll show you
Wash down the sink is like wash the cabinets
Wash down the fixtures is like wash the cabinets
Shake out bathroom rugs - I'll show you
Sweep the floors - I'll show you
Mop the floors - I'll show you
Empty the trash can - I'll show you
Wash out the trash can - I'll show you
</text>
<tasks>
<room>
BATHROOM
</room>
<task>
CLEAN,BATHROOM,[],SPRING,[
 CLEANUP,BATHROOM,[],SPRING,[
  GRAB AND PUT,DIRTY CLOTHING,[HAMPER,HANDS],SPRING,[];
  GRAB AND PUT,TRASH,[TRASH CAN,HANDS],SPRING,[];
  WIPE DOWN,SINK,[DISINFECTANT WIPE],SPRING,[];
  WIPE DOWN,TUB,[DISINFECTANT WIPE],SPRING,[];
  SCRUB OUT,INSIDE OF TOILET,[TOILET BRUSH,TOILET
CLEANER],SPRING,[];
  WIPE DOWN,OUTSIDE OF TOILET,[DISINFECTANT WIPE],SPRING,[];
  WIPE DOWN,MIRROR,[GLASS CLEANER],SPRING,[];
  PICKUP AND PUT AWAY,ITEMS,[HANDS],SPRING,[];
  SWEEP,FLOOR,[SWIFFER],SPRING,[];
 ];
 DUST DOWN,CEILING,[FEATHER DUSTER],SPRING,[];
 DUST DOWN,CORNERS,[FEATHER DUSTER],SPRING,[];
 DUST,VENTS,[GENTLE CLEANSER,CLEANING CLOTH],SPRING,[];
 DUST,FANS,[GENTLE CLEANSER,CLEANING CLOTH],SPRING,[];
 TAKE DOWN,DRAPERIES,[HANDS],SPRING,[];
 TAKE DOWN,CURTAINS,[HANDS],SPRING,[];
 TAKE DOWN,BLINDS,[HANDS],SPRING,[];
 SCRUB,SHOWER,[COMMERCIAL CLEANER],SPRING,[];
 SCRUB,TUB,[COMMERCIAL CLEANER],SPRING,[];
 SCRUB DOWN,TOILET,[COMMERCIAL CLEANER],SPRING,[];
 WASH,CABINETS,[DAMP SPONGE],SPRING,[];
 WASH DOWN,SINK,[DAMP SPONGE],SPRING,[];
 WASH DOWN,FIXTURES,[DAMP SPONGE],SPRING,[];
 SHAKE OUT,RUGS,[HANDS],SPRING,[];
 SWEEP,FLOORS,[BROOM],SPRING,[];
 MOP,FLOORS,[TOWEL],SPRING,[];
 EMPTY,TRASH CAN,[HANDS],SPRING,[];
 WASH OUT,TRASH CAN,[WINDEX,SPONGE],SPRING,[];
];
</task>
</tasks>
</lessonplan>
```

Lesson Plan 7

<lessonplan>
<text>
Task: Clean the living room in the spring

Steps:
Begin with a 15 Minute Living Room Cleanup
Dust down the ceiling and corners of walls
Dust and clean all art and photographs along the wall
Dust and clean the ceiling fan
Take down draperies, curtains, and blinds to wash or have cleaned according to the manufacturer's directions
Dust and clean out the couches and chairs
Dust down and clean all accent lamps and knickknacks
Dust down the books and the shelves.
Dust down accent tables and the entertainment center
Clean the carpets and rugs
Take the time to clean the doormats inside and outside your doorways

Answers:
Begin with a 15 Minute Living Room Cleanup with your hands
Dust down the ceiling and corners with a feather duster
Dust and clean art with a lightly wet clean cloth
Dust and clean photographs with a lightly wet clean cloth
Dust and clean the ceiling fan with Murphys Oil Soap
Take down draperies with your hands
Take down curtains with your hands
Take down blinds with your hands
Dust and clean out the couches and chairs with a dust rag, vacuum cleaner, and upholstery shampoo machine
Dust down and clean all accent lamps and knickknacks with a duster
Dust down the books and the shelves with a dust rag
Dust down accent tables and the entertainment center with a dust rag
Clean the carpets and rugs with a vacuum and rug shampoo machine
Clean the doormats with a vacuum cleaner

How To:
Begin with a 15 Minute Living Room Cleanup:
- Put away all items with your hands
- Brush off couch with soft brush
- Fluff couch pillows with your hands
- Dust down the coffee table with a duster
- Arrange magazines and books with your hands
- Vacuum floor with a sweeper
Dust down the ceiling - I'll show you
Dust down the corners is similar to dust down the ceiling
Dust and clean art - I'll show you
Dust and clean photographs is similar to dust and clean art
Dust and clean the ceiling fan - I'll show you
Take down draperies - I'll show you
Take down curtains is similar to take down the draperies
Take down blinds is similar to take down the draperies
Dust and clean out the couches - I'll show you

```
Dust and clean out the chairs is similar to dust and clean out the
couches
Dust down and clean all accent lamps - I'll show you
Dust down and clean all knickknacks is similar to dust down and
clean all accent lamps
Dust down the books - I'll show you
Dust down the shelves is similar to dust down the books
Dust down accent tables is similar to dust down the books
Dust down the entertainment center is similar to dust down the books
Clean the carpets - I'll show you
Clean the rugs is similar to clean the carpets
Clean the doormats - I'll show you
</text>
<tasks>
<room>
LIVING ROOM
</room>
<task>
CLEAN,LIVING ROOM,[],SPRING,[
 CLEANUP,LIVING ROOM,[],SPRING,[
  PUT AWAY,ITEMS,[HANDS],SPRING,[];
  BRUSH OFF,COUCH,[SOFT BRUSH],SPRING,[];
  FLUFF,COUCH PILLOW,[HANDS],SPRING,[];
  DUST DOWN,COFFEE TABLE,[DUSTER],SPRING,[];
  ARRANGE,MAGAZINES,[HANDS],SPRING,[];
  ARRANGE,BOOKS,[HANDS],SPRING,[];
  VACUUM,FLOOR,[SWEEPER],SPRING,[];
 ];
 DUST DOWN,CEILING,[FEATHER DUSTER],SPRING,[];
 DUST DOWN,CORNERS,[FEATHER DUSTER],SPRING,[];
 DUST AND CLEAN,ART,[LIGHTLY WET CLEAN CLOTH],SPRING,[];
 DUST AND CLEAN,PHOTOGRAPHS,[LIGHTLY WET CLEAN CLOTH],SPRING,[];
 DUST AND CLEAN,CEILING FAN,[MURPHYS OIL SOAP],SPRING,[];
 TAKE DOWN,DRAPERIES,[HANDS],SPRING,[];
 TAKE DOWN,CURTAINS,[HANDS],SPRING,[];
 TAKE DOWN,BLINDS,[HANDS],SPRING,[];
 DUST AND CLEAN OUT,COUCHES,[DUST RAG,VACUUM CLEANER,UPHOLSTERY
SHAMPOO MACHINE],SPRING,[];
 DUST AND CLEAN OUT,CHAIRS,[DUST RAG,VACUUM CLEANER,UPHOLSTERY
SHAMPOO MACHINE],SPRING,[];
 DUST DOWN AND CLEAN,ACCENT LAMPS,[DUSTER],SPRING,[];
 DUST DOWN AND CLEAN,KNICKKNACKS,[DUSTER],SPRING,[];
 DUST DOWN,BOOKS,[DUST RAG],SPRING,[];
 DUST DOWN,SHELVES,[DUST RAG],SPRING,[];
 DUST DOWN,ACCENT TABLES,[DUST RAG],SPRING,[];
 DUST DOWN,ENTERTAINMENT CENTER,[DUST RAG],SPRING,[];
 CLEAN,CARPETS,[VACUUM,RUG SHAMPOO MACHINE],SPRING,[];
 CLEAN,RUGS,[VACUUM,RUG SHAMPOO MACHINE],SPRING,[];
 CLEAN,DOORMATS,[VACUUM CLEANER],SPRING,[];
];
</task>
</tasks>
</lessonplan>
```

Lesson Plan 8

```
<lessonplan>
<text>
Task: Clean the bedroom in the spring

Steps:
Start with the 15 Minute Bedroom Cleanup
Dust down the ceiling and corners.
Clean the ceiling fan.
Take down draperies and curtains to wash or have cleaned according
to the directions.
Wash all bedding
Dust down and clean all accent lamps and knickknacks.
Dust down all dressers, chests, and nightstands.
Clean any mirrors.
Clean flooring.

Answers:
Start with the 15 Minute Bedroom Cleanup
Dust down the ceiling and corners with a feather duster
Clean the ceiling fan with Murphys Oil Soap
Take down draperies and curtains with your hands
Wash all bedding in the washing machine
Dust down and clean all accent lamps and knickknacks with a duster
Dust down all dressers, chests, and nightstands with a duster
Clean any mirrors with windex and newspapers
Clean flooring with vacuum cleaner

How To:
Start with the 15 Minute Bedroom Cleanup:
- Grab all dirty clothing and put it in a hamper with your hands
- Grab all clean clothes and refold or rehang with your hands
- Grab all trash and put in the trashcan with your hands
- Make the bed with your hands
- Pickup all the misplaced items with your hands
- Straighten surfaces with your hands
- Vacuum the floor with a sweeper
Dust down the ceiling - I'll show you
Dust down the corners is similar to dust down the ceiling
Clean the ceiling fan - I'll show you
Take down draperies - I'll show you
Take down curtains is similar to take down the draperies
Wash all bedding - I'll show you
Dust down and clean all accent lamps - I'll show you
Dust down and clean all knickknacks is similar to dust down and
clean all accent lamps
Dust down all dressers - I'll show you
Dust down all chests is similar to dust down dressers
Dust down all nightstands is similar to dust down dressers
Clean any mirrors - I'll show you
Clean flooring with vacuum cleaner - I'll show you
</text>
<tasks>
```

```
<room>
BEDROOM
</room>
<task>
CLEAN,BEDROOM,[],SPRING,[
 CLEANUP,BEDROOM,[],SPRING,[
  GRAB,DIRTY CLOTHING,[HANDS],SPRING,[];
  PUT,DIRTY CLOTHING,[HAMPER],SPRING,[];
  GRAB,CLEAN CLOTHES,[HANDS],SPRING,[];
  REFOLD OR REHANG,CLEAN CLOTHES,[HANDS],SPRING,[];
  GRAB,TRASH,[HANDS],SPRING,[];
  PUT,TRASH,[TRASHCAN],SPRING,[];
  MAKE,BED,[HANDS],SPRING,[];
  PICKUP,MISPLACED ITEMS,[HANDS],SPRING,[];
  STRAIGHTEN,SURFACES,[HANDS],SPRING,[];
  VACUUM,FLOOR,[SWEEPER],SPRING,[];
 ];
 DUST DOWN,CEILING,[FEATHER DUSTER],SPRING,[];
 DUST DOWN,CORNERS,[FEATHER DUSTER],SPRING,[];
 CLEAN,CEILING FAN,[MURPHYS OIL SOAP],SPRING,[];
 TAKE DOWN,DRAPERIES,[HANDS],SPRING,[];
 TAKE DOWN,CURTAINS,[HANDS],SPRING,[];
 WASH,BEDDING,[WASHING MACHINE],SPRING,[];
 DUST DOWN AND CLEAN,ACCENT LAMPS,[DUSTER],SPRING,[];
 DUST DOWN AND CLEAN,KNICKKNACKS,[DUSTER],SPRING,[];
 DUST DOWN,DRESSERS,[DUSTER],SPRING,[];
 DUST DOWN,CHESTS,[DUSTER],SPRING,[];
 DUST DOWN,NIGHTSTANDS,[DUSTER],SPRING,[];
 CLEAN,MIRRORS,[WINDEX,NEWSPAPERS],SPRING,[];
 CLEAN,FLOORING,[VACUUM CLEANER],SPRING,[];
];
</task>
</tasks>
</lessonplan>
```

Lesson Plan 9

```
<lessonplan>
<text>
Task: Clean the dining room in the spring

Steps:
Dust down the ceiling and corners.
Dust and clean all wall art.
Dust and clean the Ceiling Fan.
Take down draperies and curtains to wash or have cleaned according
to the directions.
Wash down the dining table, chairs, and any other furniture
thoroughly.
Clean the carpets and rugs.

Answers:
Dust down the ceiling and corners with a feather duster
Dust and clean art with a lightly wet clean cloth
Dust and clean the ceiling fan with Murphys Oil Soap
Take down draperies and curtains with your hands
Wash down the dining table, chairs, and any other furniture with no
tools
Clean the carpets and rugs with a vacuum and rug shampoo machine

How To:
Dust down the ceiling - I'll show you
Dust down the corners is similar to dust down the ceiling
Dust and clean art - I'll show you
Dust and clean the ceiling fan - I'll show you
Take down draperies - I'll show you
Take down curtains is similar to take down the draperies
Wash down the dining table:
- Clean wood with damp cloth
- Oil wood with furniture oil
Wash down chairs:
- Clean wood with damp cloth
- Oil wood with furniture oil
- Spot clean upholstery with spot cleaner
Wash down other furniture is similar to wash down chairs
Clean the carpets - I'll show you
Clean the rugs is similar to clean the carpets
</text>
<tasks>
<room>
DINING ROOM
</room>
<task>
CLEAN,DINING ROOM,[],SPRING,[
 DUST DOWN,CEILING,[FEATHER DUSTER],SPRING,[];
 DUST DOWN,CORNERS,[FEATHER DUSTER],SPRING,[];
 DUST AND CLEAN,ART,[LIGHTLY WET CLEAN CLOTH],SPRING,[];
 DUST AND CLEAN,CEILING FAN,[MURPHYS OIL SOAP],SPRING,[];
 TAKE DOWN,DRAPERIES,[HANDS],SPRING,[];
```

```
 TAKE DOWN,CURTAINS,[HANDS],SPRING,[];
 WASH DOWN,DINING TABLE,[],SPRING,[
  CLEAN,WOOD,[DAMP CLOTH],SPRING,[];
  OIL,WOOD,[FURNITURE OIL],SPRING,[];
 ];
 WASH DOWN,CHAIRS,[],SPRING,[
  CLEAN,WOOD,[DAMP CLOTH],SPRING,[];
  OIL,WOOD,[FURNITURE OIL],SPRING,[];
  SPOT CLEAN,UPHOLSTERY,[SPOT CLEANER],SPRING,[];
 ];
 WASH DOWN,OTHER FURNITURE,[],SPRING,[
  CLEAN,WOOD,[DAMP CLOTH],SPRING,[];
  OIL,WOOD,[FURNITURE OIL],SPRING,[];
  SPOT CLEAN,UPHOLSTERY,[SPOT CLEANER],SPRING,[];
 ];
 CLEAN,CARPETS,[VACUUM,RUG SHAMPOO MACHINE],SPRING,[];
 CLEAN,RUGS,[VACUUM,RUG SHAMPOO MACHINE],SPRING,[];
];
</task>
</tasks>
</lessonplan>
```

Lesson Plan 10

```
<lessonplan>
<text>
Task: Clean the house every day

Steps:
Make beds
Do a load of laundry
Wash dishes
Take out trash
Pick up and put away any clothes or items lying around

Answers:
Make beds with your hands
Do a load of laundry in the washing machine
Wash dishes with the dishwasher
Take out trash with your hands
Pick up clothes with your hands and put them in the hamper
Pick up items lying around with your hands and put them away

How To:
Make beds - I'll show you
Do a load of laundry - I'll show you
Wash dishes - I'll show you
Take out trash - I'll show you
Pick up clothes - I'll show you
Put clothes in hamper - I'll show you
Pick up items lying around is similar to pick up clothes
Put items lying around away - I'll show you
</text>
<tasks>
<room>
HOUSE
</room>
<task>
CLEAN,HOUSE,[],EVERY DAY,[
 MAKE,BEDS,[HANDS],EVERY DAY,[];
 DO,LAUNDRY,[WASHING MACHINE],EVERY DAY,[];
 WASH,DISHES,[DISHWASHER],EVERY DAY,[];
 TAKE OUT,TRASH,[HANDS],EVERY DAY,[];
 PICK UP,CLOTHES,[HANDS],EVERY DAY,[];
 PUT,CLOTHES,[HAMPER],EVERY DAY,[];
 PICK UP,ITEMS LYING AROUND,[HANDS],EVERY DAY,[];
 PUT,ITEMS LYING AROUND,[AWAY],EVERY DAY,[];
];
</task>
</tasks>
</lessonplan>
```

Appendix II – Base Grammar Rules

The grammars are context free rules that specify the word patterns corresponding to the slot [95]. The syntax for a grammar for a slot is:

```
# optional comment
[token_name]
        (<pattern a>)
        (<pattern b>)
<Macro1>
        (<rewrite rule for Macro1>)
<Macro2>
        (<rewrite rule for Macro2)
;
```

The slot name is enclosed in square brackets. After this follow a set of re-write patterns, one per line each enclosed in parentheses, with leading white space on the line. After the basic re-write patterns follow the non-terminal rewrites, which have the same format. Notation used in pattern specification:

- Lower case strings are terminals.
- Upper case strings are macros.
- Names enclosed in [] are non-terminals (calls to other slot rules).
- Regular Expressions:
  *item indicates 0 or 1 repetitions of the item
  + indicates 1 or more repetitions
  +* indicates 0 or more repetitions (equivalent to a Kleene star)
- #include <filename> reads file at that point.

A macro has rewrite rules specified later in the same grammar rule. These cause a text substitution of all of the expansions for the macro, but do not cause a non-terminal slot to appear in the parse. Macros allow for a simpler expression of the grammar without inserting unwanted slots in the parse.

```
# Base Grammar for HRI
# Created June 25, 2007 by David O. Johnson
# Contains everything, but action, object, tool, & condition tokens
#     which are contained in task.gra

# Yes/No answer
[answer]
      ([yes])
      ([no])
      ([unknown])
;

[yes]
```

```
      (yes)
      (yeh)
      (right)
;

[no]
      (no)
      (nah)
      (wrong)
;

[unknown]
      (dont know)
      (not sure)
;

[greeting]
      (hello)
;

[operator]
      (same as)
      (similar to)
      (like)
;

[quit]
      (bye)
      (forget it)
      (stop)
      (quit)
      (start over)
;

[requestRepeat]
      (huh)
;

[connector]
      ([after])
      ([before])
      ([next])
;

[after]
      (after)
;

[before]
      (before)
;

[next]
      (next)
```

```
      (then)
;

[tools]
      ([tool])
      (+[tool] and [tool])
      (+[no_tool])
;

[no_tool]
      (no tools)
      (none)
;

[anaph]
      ([singular_anaphor])
      ([plural_anaphor])
;

[singular_anaphor]
      (THAT *one)
THAT
      (that)
      (this)
      (THE [_other])
THE
      (the)
      (any)
      (an)
;

[_other]
      (other)
;

[plural_anaphor]
      (everything)
      (those)
;

[show]
      (SHOW *YOU)
      (WATCH *ME)
SHOW
      (show)
      (demonstrate)
YOU
      (you)
WATCH
      (watch)
      (observe)
      (look *at)
ME
      (me)
```

;

Appendix III – Task Grammar Rules

```
# Created by GrammarParser.java on 5/20/08 3:09 PM

[action]
      (fluff)
      (scrub out)
      (loading)
      (wash)
      (grab and put)
      (mop)
      (swoosh)
      (wipe out)
      (shake out)
      (take down)
      (sweep)
      (dust and clean)
      (do)
      (arrange)
      (spot clean)
      (clean)
      (straighten out)
      (squeegee)
      (scrub down)
      (wipe down and clean)
      (dust down and clean)
      (wipe)
      (put away)
      (cleanup)
      (load)
      (make)
      (grab)
      (straighten)
      (oil)
      (dust)
      (fold or hang)
      (refold or rehang)
      (empty)
      (dust and clean out)
      (take out)
      (pick up)
      (pickup and put away)
      (run)
      (vacuum)
      (scrape off)
      (spray)
      (dust down)
      (throw out)
      (scrub)
      (wipe down)
      (wash down)
```

```
      (wash out)
      (brush off)
      (pickup)
      (fold)
      (put)
;

[object]
      (trash can)
      (stove top)
      (shower door)
      (faucet)
      (corners)
      (oven)
      (books)
      (jewelry)
      (dining room)
      (ceiling)
      (photographs)
      (fans)
      (curtain liner)
      (bedding)
      (countertops)
      (mirrors)
      (house)
      (ceiling fan)
      (knickknacks)
      (accent lamps)
      (inside of toilet)
      (foyer)
      (outside of toilet)
      (trash)
      (chests)
      (clean clothes)
      (fixtures)
      (dishwasher)
      (sink)
      (other furniture)
      (couch)
      (wood)
      (curtains)
      (cabinets)
      (corners of walls)
      (tub)
      (bedroom)
      (cushions)
      (night table surface)
      (magazines)
      (counters)
      (beds)
      (nightstands)
      (toilet bowl)
      (misplaced items)
      (vents)
```

```
      (clothes)
      (couches)
      (entertainment center)
      (dining table)
      (tabletops)
      (toilet seat)
      (couch pillow)
      (dishes)
      (shower)
      (blender)
      (refrigerator)
      (other small appliances)
      (toilet)
      (carpets)
      (cds)
      (upholstery)
      (bathroom)
      (crumbs and dust bunnies)
      (living room)
      (rim)
      (blinds)
      (laundry)
      (throws)
      (items)
      (art)
      (dirty clothing)
      (floors)
      (floor)
      (flooring)
      (rugs)
      (stove)
      (draperies)
      (family room)
      (toaster)
      (chairs)
      (newspapers)
      (drawers)
      (kitchen)
      (coffee table)
      (clothing)
      (mirror)
      (shelves)
      (accent tables)
      (doormats)
      (bed)
      (videos)
      (dressers)
      (surfaces)
      (items lying around)
;

[tool]
      (dust rag)
      (sponge)
```

```
      (sweeper)
      (trash can)
      (away)
      (gentle cleanser)
      (duster)
      (oven cleaner)
      (commercial cleaner)
      (towel)
      (broom)
      (trashcan)
      (vinegar)
      (disinfectant wipe)
      (glass cleaner)
      (brush)
      (hamper)
      (dishwasher)
      (cleaning cloth)
      (toilet cleaner)
      (feather duster)
      (washing machine)
      (murphys oil soap)
      (shower mist)
      (newspapers)
      (damp cloth)
      (vacuum)
      (swiffer)
      (damp sponge)
      (pledge)
      (soft brush)
      (dry rag)
      (spot cleaner)
      (windex)
      (toilet brush)
      (rug shampoo machine)
      (hands)
      (upholstery shampoo machine)
      (lightly wet clean cloth)
      (furniture oil)
      (handheld vacuum)
      (vacuum cleaner)
;

[condition]
      (spring)
      (after every use)
      (every day)
      (daily)
;
```

Appendix IV – Experimental Measurements

Experiment No.	Lesson Plan	Grounding	Semantic Inputs	Tasks in Lesson Plan	Tasks Learned Correctly	Clarifying Questions
1	1	Optimistic	Speech only	6	3	22
2	1	Cautious	Speech only	6	3	28
3	1	Pessimistic	Speech only	6	3	39
4	2	Optimistic	Speech only	10	5	19
5	2	Cautious	Speech only	10	7	64
6	2	Pessimistic	Speech only	10	10	90
7	3	Optimistic	Speech only	5	3	24
8	3	Cautious	Speech only	5	5	29
9	3	Pessimistic	Speech only	5	5	50
10	4	Optimistic	Speech only	12	4	39
11	4	Cautious	Speech only	12	5	64
12	4	Pessimistic	Speech only	12	5	91
13	5	Optimistic	Speech only	27	19	117
14	5	Cautious	Speech only	27	19	172
15	5	Pessimistic	Speech only	27	20	208
16	6	Optimistic	Speech only	29	16	114
17	6	Cautious	Speech only	29	21	205
18	6	Pessimistic	Speech only	29	17	229
19	7	Optimistic	Speech only	28	25	123
20	7	Cautious	Speech only	28	25	175
21	7	Pessimistic	Speech only	28	25	224
22	8	Optimistic	Speech only	25	19	110
23	8	Cautious	Speech only	25	19	153
24	8	Pessimistic	Speech only	25	19	187
25	9	Optimistic	Speech only	15	7	60
26	9	Cautious	Speech only	15	8	84
27	9	Pessimistic	Speech only	15	7	103
28	10	Optimistic	Speech only	9	5	42
29	10	Cautious	Speech only	9	6	52
30	10	Pessimistic	Speech only	9	6	59
31	1	Optimistic	Speech + RW	6	0	17
32	1	Cautious	Speech + RW	6	2	51
33	1	Pessimistic	Speech + RW	6	5	57
34	2	Optimistic	Speech + RW	10	8	32
35	2	Cautious	Speech + RW	10	9	53
36	2	Pessimistic	Speech + RW	10	10	85

Experiment No.	Lesson Plan	Grounding	Semantic Inputs	Tasks in Lesson Plan	Tasks Learned Correctly	Clarifying Questions
37	3	Optimistic	Speech + RW	5	3	20
38	3	Cautious	Speech + RW	5	4	33
39	3	Pessimistic	Speech + RW	5	5	46
40	4	Optimistic	Speech + RW	12	3	30
41	4	Cautious	Speech + RW	12	6	68
42	4	Pessimistic	Speech + RW	12	6	88
43	5	Optimistic	Speech + RW	27	17	85
44	5	Cautious	Speech + RW	27	24	167
45	5	Pessimistic	Speech + RW	27	25	194
46	6	Optimistic	Speech + RW	29	17	110
47	6	Cautious	Speech + RW	29	21	208
48	6	Pessimistic	Speech + RW	29	25	256
49	7	Optimistic	Speech + RW	28	22	92
50	7	Cautious	Speech + RW	28	26	166
51	7	Pessimistic	Speech + RW	28	26	213
52	8	Optimistic	Speech + RW	25	20	86
53	8	Cautious	Speech + RW	25	19	126
54	8	Pessimistic	Speech + RW	25	19	154
55	9	Optimistic	Speech + RW	15	8	48
56	9	Cautious	Speech + RW	15	7	75
57	9	Pessimistic	Speech + RW	15	9	87
58	10	Optimistic	Speech + RW	9	6	43
59	10	Cautious	Speech + RW	9	7	81
60	10	Pessimistic	Speech + RW	9	7	88
61	1	Optimistic	Speech + History	6	3	16
62	1	Cautious	Speech + History	6	4	55
63	1	Pessimistic	Speech + History	6	5	65
64	2	Optimistic	Speech + History	10	7	31
65	2	Cautious	Speech + History	10	10	67
66	2	Pessimistic	Speech + History	10	10	65
67	3	Optimistic	Speech + History	5	4	14
68	3	Cautious	Speech + History	5	5	30
69	3	Pessimistic	Speech + History	5	5	39
70	4	Optimistic	Speech + History	12	7	35
71	4	Cautious	Speech + History	12	7	92
72	4	Pessimistic	Speech + History	12	6	79
73	5	Optimistic	Speech + History	27	22	75
74	5	Cautious	Speech + History	27	23	203
75	5	Pessimistic	Speech + History	27	23	186
76	6	Optimistic	Speech + History	29	22	83

Experiment No.	Lesson Plan	Grounding	Semantic Inputs	Tasks in Lesson Plan	Tasks Learned Correctly	Clarifying Questions
77	6	Cautious	Speech + History	29	21	203
78	6	Pessimistic	Speech + History	29	24	218
79	7	Optimistic	Speech + History	28	24	74
80	7	Cautious	Speech + History	28	23	183
81	7	Pessimistic	Speech + History	28	26	185
82	8	Optimistic	Speech + History	25	19	64
83	8	Cautious	Speech + History	25	19	150
84	8	Pessimistic	Speech + History	25	21	192
85	9	Optimistic	Speech + History	15	7	37
86	9	Cautious	Speech + History	15	10	153
87	9	Pessimistic	Speech + History	15	11	149
88	10	Optimistic	Speech + History	9	7	29
89	10	Cautious	Speech + History	9	9	79
90	10	Pessimistic	Speech + History	9	9	94
91	1	Optimistic	Speech + Pointing	6	5	28
92	1	Cautious	Speech + Pointing	6	5	40
93	1	Pessimistic	Speech + Pointing	6	6	66
94	2	Optimistic	Speech + Pointing	10	10	38
95	2	Cautious	Speech + Pointing	10	9	68
96	2	Pessimistic	Speech + Pointing	10	10	83
97	3	Optimistic	Speech + Pointing	5	4	24
98	3	Cautious	Speech + Pointing	5	5	29
99	3	Pessimistic	Speech + Pointing	5	5	42
100	4	Optimistic	Speech + Pointing	12	7	46
101	4	Cautious	Speech + Pointing	12	7	80
102	4	Pessimistic	Speech + Pointing	12	7	70
103	5	Optimistic	Speech + Pointing	27	23	113
104	5	Cautious	Speech + Pointing	27	25	173
105	5	Pessimistic	Speech + Pointing	27	27	224
106	6	Optimistic	Speech + Pointing	29	21	138
107	6	Cautious	Speech + Pointing	29	23	188
108	6	Pessimistic	Speech + Pointing	29	23	238
109	7	Optimistic	Speech + Pointing	28	26	116
110	7	Cautious	Speech + Pointing	28	25	165
111	7	Pessimistic	Speech + Pointing	28	26	224
112	8	Optimistic	Speech + Pointing	25	21	104
113	8	Cautious	Speech + Pointing	25	21	147
114	8	Pessimistic	Speech + Pointing	25	21	194
115	9	Optimistic	Speech + Pointing	15	11	76
116	9	Cautious	Speech + Pointing	15	13	95

Experiment No.	Lesson Plan	Grounding	Semantic Inputs	Tasks in Lesson Plan	Tasks Learned Correctly	Clarifying Questions
117	9	Pessimistic	Speech + Pointing	15	13	132
118	10	Optimistic	Speech + Pointing	9	8	44
119	10	Cautious	Speech + Pointing	9	9	60
120	10	Pessimistic	Speech + Pointing	9	9	72
121	1	Optimistic	Speech + FOV	6	4	25
122	1	Cautious	Speech + FOV	6	6	42
123	1	Pessimistic	Speech + FOV	6	6	54
124	2	Optimistic	Speech + FOV	10	9	42
125	2	Cautious	Speech + FOV	10	10	61
126	2	Pessimistic	Speech + FOV	10	10	83
127	3	Optimistic	Speech + FOV	5	2	29
128	3	Cautious	Speech + FOV	5	5	31
129	3	Pessimistic	Speech + FOV	5	5	44
130	4	Optimistic	Speech + FOV	12	7	42
131	4	Cautious	Speech + FOV	12	7	61
132	4	Pessimistic	Speech + FOV	12	7	85
133	5	Optimistic	Speech + FOV	27	25	113
134	5	Cautious	Speech + FOV	27	25	174
135	5	Pessimistic	Speech + FOV	27	27	237
136	6	Optimistic	Speech + FOV	29	21	135
137	6	Cautious	Speech + FOV	29	22	189
138	6	Pessimistic	Speech + FOV	29	25	240
139	7	Optimistic	Speech + FOV	28	26	107
140	7	Cautious	Speech + FOV	28	26	167
141	7	Pessimistic	Speech + FOV	28	26	223
142	8	Optimistic	Speech + FOV	25	20	105
143	8	Cautious	Speech + FOV	25	20	142
144	8	Pessimistic	Speech + FOV	25	21	196
145	9	Optimistic	Speech + FOV	15	13	63
146	9	Cautious	Speech + FOV	15	13	110
147	9	Pessimistic	Speech + FOV	15	15	165
148	10	Optimistic	Speech + FOV	9	8	44
149	10	Cautious	Speech + FOV	9	9	65
150	10	Pessimistic	Speech + FOV	9	9	81
151	1	Optimistic	Speech + Nodding	6	5	32
152	1	Cautious	Speech + Nodding	6	6	36
153	1	Pessimistic	Speech + Nodding	6	6	45
154	2	Optimistic	Speech + Nodding	10	9	48
155	2	Cautious	Speech + Nodding	10	10	62
156	2	Pessimistic	Speech + Nodding	10	8	103

Experiment No.	Lesson Plan	Grounding	Semantic Inputs	Tasks in Lesson Plan	Tasks Learned Correctly	Clarifying Questions
157	3	Optimistic	Speech + Nodding	5	4	28
158	3	Cautious	Speech + Nodding	5	5	28
159	3	Pessimistic	Speech + Nodding	5	5	51
160	4	Optimistic	Speech + Nodding	12	7	43
161	4	Cautious	Speech + Nodding	12	7	58
162	4	Pessimistic	Speech + Nodding	12	7	68
163	5	Optimistic	Speech + Nodding	27	24	113
164	5	Cautious	Speech + Nodding	27	25	179
165	5	Pessimistic	Speech + Nodding	27	25	216
166	6	Optimistic	Speech + Nodding	29	22	148
167	6	Cautious	Speech + Nodding	29	23	211
168	6	Pessimistic	Speech + Nodding	29	22	259
169	7	Optimistic	Speech + Nodding	28	25	110
170	7	Cautious	Speech + Nodding	28	27	173
171	7	Pessimistic	Speech + Nodding	28	28	225
172	8	Optimistic	Speech + Nodding	25	21	116
173	8	Cautious	Speech + Nodding	25	21	139
174	8	Pessimistic	Speech + Nodding	25	20	176
175	9	Optimistic	Speech + Nodding	15	9	62
176	9	Cautious	Speech + Nodding	15	13	111
177	9	Pessimistic	Speech + Nodding	15	15	151
178	10	Optimistic	Speech + Nodding	9	7	45
179	10	Cautious	Speech + Nodding	9	8	73
180	10	Pessimistic	Speech + Nodding	9	9	69
181	1	Optimistic	All	6	0	31
182	1	Cautious	All	6	3	54
183	1	Pessimistic	All	6	3	61
184	2	Optimistic	All	10	4	36
185	2	Cautious	All	10	7	98
186	2	Pessimistic	All	10	8	57
187	3	Optimistic	All	5	3	15
188	3	Cautious	All	5	3	60
189	3	Pessimistic	All	5	5	36
190	4	Optimistic	All	12	6	35
191	4	Cautious	All	12	7	72
192	4	Pessimistic	All	12	7	103
193	5	Optimistic	All	27	20	78
194	5	Cautious	All	27	27	172
195	5	Pessimistic	All	27	25	207
196	6	Optimistic	All	29	26	106

Experiment No.	Lesson Plan	Grounding	Semantic Inputs	Tasks in Lesson Plan	Tasks Learned Correctly	Clarifying Questions
197	6	Cautious	All	29	29	268
198	6	Pessimistic	All	29	29	239
199	7	Optimistic	All	28	23	88
200	7	Cautious	All	28	26	190
201	7	Pessimistic	All	28	28	205
202	8	Optimistic	All	25	19	76
203	8	Cautious	All	25	20	186
204	8	Pessimistic	All	25	21	209
205	9	Optimistic	All	15	10	45
206	9	Cautious	All	15	13	144
207	9	Pessimistic	All	15	13	132
208	10	Optimistic	All	9	6	35
209	10	Cautious	All	9	6	131
210	10	Pessimistic	All	9	9	115

Appendix V – Confusion Matrices

Unbalanced Training Set and One Output Node (h = 5 – 50)

		Classified As		
		Best	Not Best	Unknown
Input	Best	0	28	176
	Not Best	0	4070	604
Classified Correctly = 83%				

Training for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	0	17	168
	Not Best	0	2529	440
Classified Correctly = 80%				

Strict Test for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	49	136	0
	Not Best	3	2966	0
Classified Correctly = 96%				

Fuzzy Test for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	38	31	135
	Not Best	3	4081	590
Classified Correctly = 84%				

Training for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	38	18	129
	Not Best	1	2540	428
Classified Correctly = 82%				

Strict Test for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	51	134	0
	Not Best	6	2963	0
Classified Correctly = 96%				

Fuzzy Test for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	5	25	174
	Not Best	0	4129	545
Classified Correctly = 85%				

Training for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	10	19	156
	Not Best	0	2538	431
Classified Correctly = 81%				

Strict Test for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	54	131	0
	Not Best	18	2951	0
Classified Correctly = 95%				

Fuzzy Test for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	36	30	138
	Not Best	3	4410	261
Classified Correctly = 91%				

Training for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	42	18	125
	Not Best	3	2536	430
Classified Correctly = 82%				

Strict Test for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	140	45	0
	Not Best	5	2964	0
Classified Correctly = 98%				

Fuzzy Test for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	6	31	167
	Not Best	2	4237	435
Classified Correctly = 87%				

Training for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	10	19	156
	Not Best	0	2529	440
Classified Correctly = 81%				

Strict Test for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	132	53	0
	Not Best	11	2958	0
Classified Correctly = 98%				

Fuzzy Test for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	18	28	158
	Not Best	0	4174	500
Classified Correctly = 86%				

Training for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	15	19	151
	Not Best	3	2533	433
Classified Correctly = 81%				

Strict Test for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	135	50	0
	Not Best	9	2960	0
Classified Correctly = 98%				

Fuzzy Test for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	11	37	156
	Not Best	0	4509	165
Classified Correctly = 93%				

Training for h = 35

		Classified As		
		Best	Not Best	Unknown
Input	Best	6	21	158
	Not Best	0	2551	418
Classified Correctly = 81%				

Strict Test for h = 35

		Classified As		
		Best	Not Best	Unknown
Input	Best	136	49	0
	Not Best	5	2964	0
Classified Correctly = 98%				

Fuzzy Test for h = 35

		Classified As		
		Best	Not Best	Unknown
Input	Best	12	2	190
	Not Best	0	68	4606
Classified Correctly = 2%				

Training for h = 40

		Classified As		
		Best	Not Best	Unknown
Input	Best	2	2	181
	Not Best	0	43	2926
Classified Correctly = 1%				

Strict Test for h = 40

		Classified As		
		Best	Not Best	Unknown
Input	Best	135	50	0
	Not Best	9	2960	0
Classified Correctly = 98%				

Fuzzy Test for h = 40

		Classified As		
		Best	Not Best	Unknown
Input	Best	23	66	115
	Not Best	0	4656	18
Classified Correctly = 96%				

Training for h = 45

		Classified As		
		Best	Not Best	Unknown
Input	Best	5	41	139
	Not Best	0	2953	16
Classified Correctly = 94%				

Strict Test for h = 45

		Classified As		
		Best	Not Best	Unknown
Input	Best	49	136	0
	Not Best	5	2964	0
Classified Correctly = 96%				

Fuzzy Test for h = 45

		Classified As		
		Best	Not Best	Unknown
Input	Best	42	64	98
	Not Best	1	4644	29
Classified Correctly = 96%				

Training for h = 50

		Classified As		
		Best	Not Best	Unknown
Input	Best	47	41	97
	Not Best	0	2942	27
Classified Correctly = 95%				

Strict Test for h = 50

		Classified As		
		Best	Not Best	Unknown
Input	Best	52	133	0
	Not Best	10	2959	0
Classified Correctly = 95%				

Fuzzy Test for h = 50

Unbalanced Training Set and One Output Node (h = 55 – 100)

		Classified As		
		Best	Not Best	Unknown
Input	Best	7	66	131
	Not Best	0	4647	27
Classified Correctly = 95%				

Training for h = 55

		Classified As		
		Best	Not Best	Unknown
Input	Best	8	43	134
	Not Best	0	2940	29
Classified Correctly = 93%				

Strict Test for h = 55

		Classified As		
		Best	Not Best	Unknown
Input	Best	49	136	0
	Not Best	6	2963	0
Classified Correctly = 95%				

Fuzzy Test for h = 55

		Classified As		
		Best	Not Best	Unknown
Input	Best	108	1	95
	Not Best	4	7	4663
Classified Correctly = 2%				

Training for h = 60

		Classified As		
		Best	Not Best	Unknown
Input	Best	128	2	55
	Not Best	5	5	2959
Classified Correctly = 4%				

Strict Test for h = 60

		Classified As		
		Best	Not Best	Unknown
Input	Best	177	8	0
	Not Best	2581	388	0
Classified Correctly = 18%				

Fuzzy Test for h = 60

		Classified As		
		Best	Not Best	Unknown
Input	Best	34	3	167
	Not Best	4	47	4623
Classified Correctly = 2%				

Training for h = 65

		Classified As		
		Best	Not Best	Unknown
Input	Best	36	2	147
	Not Best	3	37	2929
Classified Correctly = 2%				

Strict Test for h = 65

		Classified As		
		Best	Not Best	Unknown
Input	Best	172	13	0
	Not Best	2570	399	0
Classified Correctly = 18%				

Fuzzy Test for h = 65

		Classified As		
		Best	Not Best	Unknown
Input	Best	9	78	117
	Not Best	0	4648	26
Classified Correctly = 95%				

Training for h = 70

		Classified As		
		Best	Not Best	Unknown
Input	Best	12	50	123
	Not Best	0	2952	17
Classified Correctly = 94%				

Strict Test for h = 70

		Classified As		
		Best	Not Best	Unknown
Input	Best	51	134	0
	Not Best	5	2964	0
Classified Correctly = 96%				

Fuzzy Test for h = 70

		Classified As		
		Best	Not Best	Unknown
Input	Best	15	89	100
	Not Best	3	4669	2
Classified Correctly = 96%				

Training for h = 75

		Classified As		
		Best	Not Best	Unknown
Input	Best	30	52	103
	Not Best	1	2962	6
Classified Correctly = 95%				

Strict Test for h = 75

		Classified As		
		Best	Not Best	Unknown
Input	Best	49	136	0
	Not Best	4	2965	0
Classified Correctly = 96%				

Fuzzy Test for h = 75

		Classified As		
		Best	Not Best	Unknown
Input	Best	16	64	124
	Not Best	0	4629	45
Classified Correctly = 95%				

Training for h = 80

		Classified As		
		Best	Not Best	Unknown
Input	Best	13	43	129
	Not Best	0	2927	42
Classified Correctly = 93%				

Strict Test for h = 80

		Classified As		
		Best	Not Best	Unknown
Input	Best	129	56	0
	Not Best	7	2962	0
Classified Correctly = 98%				

Fuzzy Test for h = 80

		Classified As		
		Best	Not Best	Unknown
Input	Best	111	0	93
	Not Best	10	6	4658
Classified Correctly = 2%				

Training for h = 85

		Classified As		
		Best	Not Best	Unknown
Input	Best	129	0	56
	Not Best	13	3	2953
Classified Correctly = 4%				

Strict Test for h = 85

		Classified As		
		Best	Not Best	Unknown
Input	Best	185	0	0
	Not Best	2949	20	0
Classified Correctly = 6%				

Fuzzy Test for h = 85

		Classified As		
		Best	Not Best	Unknown
Input	Best	19	83	102
	Not Best	8	4660	6
Classified Correctly = 96%				

Training for h = 90

		Classified As		
		Best	Not Best	Unknown
Input	Best	19	51	115
	Not Best	4	2962	3
Classified Correctly = 95%				

Strict Test for h = 90

		Classified As		
		Best	Not Best	Unknown
Input	Best	50	135	0
	Not Best	7	2962	0
Classified Correctly = 95%				

Fuzzy Test for h = 90

		Classified As		
		Best	Not Best	Unknown
Input	Best	11	38	155
	Not Best	0	4496	178
Classified Correctly = 92%				

Training for h = 95

		Classified As		
		Best	Not Best	Unknown
Input	Best	16	19	150
	Not Best	2	2564	403
Classified Correctly = 82%				

Strict Test for h = 95

		Classified As		
		Best	Not Best	Unknown
Input	Best	141	44	0
	Not Best	6	2963	0
Classified Correctly = 98%				

Fuzzy Test for h = 95

		Classified As		
		Best	Not Best	Unknown
Input	Best	116	2	86
	Not Best	3	43	4628
Classified Correctly = 3%				

Training for h = 100

		Classified As		
		Best	Not Best	Unknown
Input	Best	133	1	51
	Not Best	4	35	2930
Classified Correctly = 5%				

Strict Test for h = 100

		Classified As		
		Best	Not Best	Unknown
Input	Best	161	24	0
	Not Best	655	2314	0
Classified Correctly = 78%				

Fuzzy Test for h = 100

Small Balanced Training Set and One Output Node (h = 5 – 35)

		Classified As		
		Best	Not Best	Unknown
Input	Best	73	15	116
	Not Best	3	145	53
Classified Correctly = 54%				

Training for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	63	12	110
	Not Best	73	2157	739
Classified Correctly = 70%				

Strict Test for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	169	16	0
	Not Best	448	2521	0
Classified Correctly = 85%				

Fuzzy Test for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	69	10	125
	Not Best	2	119	80
Classified Correctly = 46%				

Training for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	63	10	112
	Not Best	29	1961	979
Classified Correctly = 64%				

Strict Test for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	168	17	0
	Not Best	440	2529	0
Classified Correctly = 86%				

Fuzzy Test for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	137	9	58
	Not Best	3	112	86
Classified Correctly = 61%				

Training for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	141	11	33
	Not Best	41	1984	944
Classified Correctly = 67%				

Strict Test for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	169	16	0
	Not Best	264	2705	0
Classified Correctly = 91%				

Fuzzy Test for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	147	8	49
	Not Best	3	146	52
Classified Correctly = 72%				

Training for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	146	13	26
	Not Best	77	2253	639
Classified Correctly = 76%				

Strict Test for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	169	16	0
	Not Best	262	2707	0
Classified Correctly = 91%				

Fuzzy Test for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	131	5	68
	Not Best	1	150	50
Classified Correctly = 69%				

Training for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	140	4	41
	Not Best	48	2296	625
Classified Correctly = 77%				

Strict Test for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	169	16	0
	Not Best	270	2699	0
Classified Correctly = 91%				

Fuzzy Test for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	132	4	68
	Not Best	1	147	53
Classified Correctly = 69%				

Training for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	138	6	41
	Not Best	56	2248	665
Classified Correctly = 76%				

Strict Test for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	169	16	0
	Not Best	282	2687	0
Classified Correctly = 91%				

Fuzzy Test for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	147	5	52
	Not Best	3	145	53
Classified Correctly = 72%				

Training for h = 35

		Classified As		
		Best	Not Best	Unknown
Input	Best	147	3	35
	Not Best	75	2255	639
Classified Correctly = 76%				

Strict Test for h = 35

		Classified As		
		Best	Not Best	Unknown
Input	Best	169	16	0
	Not Best	341	2628	0
Classified Correctly = 89%				

Fuzzy Test for h = 35

Large Balanced Training Set and One Output Node (h = 5 – 35)

		Classified As		
		Best	Not Best	Unknown
Input	Best	1078	160	3437
	Not Best	16	832	3826
Classified Correctly = 20%				

Training for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	50	1	134
	Not Best	7	275	2687
Classified Correctly = 10%				

Strict Test for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	170	15	0
	Not Best	474	2495	0
Classified Correctly = 84%				

Fuzzy Test for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	1147	298	3230
	Not Best	20	3292	1362
Classified Correctly = 47%				

Training for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	53	13	119
	Not Best	13	2074	882
Classified Correctly = 67%				

Strict Test for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	170	15	0
	Not Best	496	2473	0
Classified Correctly = 84%				

Fuzzy Test for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	1354	321	3000
	Not Best	61	3393	1220
Classified Correctly = 51%				

Training for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	58	11	116
	Not Best	37	2143	789
Classified Correctly = 70%				

Strict Test for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	170	15	0
	Not Best	455	2514	0
Classified Correctly = 85%				

Fuzzy Test for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	1101	276	3298
	Not Best	15	2916	1743
Classified Correctly = 43%				

Training for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	52	10	123
	Not Best	9	1899	1061
Classified Correctly = 62%				

Strict Test for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	169	16	0
	Not Best	435	2534	0
Classified Correctly = 86%				

Fuzzy Test for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	1377	161	3137
	Not Best	54	2630	1990
Classified Correctly = 43%				

Training for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	60	10	115
	Not Best	40	1896	1033
Classified Correctly = 62%				

Strict Test for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	170	15	0
	Not Best	454	2515	0
Classified Correctly = 85%				

Fuzzy Test for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	1033	161	3481
	Not Best	24	2687	1963
Classified Correctly = 40%				

Training for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	52	10	123
	Not Best	15	1916	1038
Classified Correctly = 62%				

Strict Test for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	168	17	0
	Not Best	466	2503	0
Classified Correctly = 85%				

Fuzzy Test for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	1332	161	3182
	Not Best	74	2658	1942
Classified Correctly = 43%				

Training for h = 35

		Classified As		
		Best	Not Best	Unknown
Input	Best	56	10	119
	Not Best	48	1892	1029
Classified Correctly = 62%				

Strict Test for h = 35

		Classified As		
		Best	Not Best	Unknown
Input	Best	170	15	0
	Not Best	466	2503	0
Classified Correctly = 85%				

Fuzzy Test for h = 35

Unbalanced Training Set and Two Output Nodes (h = 5 – 35)

		Classified As		
		Best	Not Best	Unknown
Input	Best	0	26	178
	Not Best	0	4049	625
Classified Correctly = 83%				

Training for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	0	15	170
	Not Best	0	2503	466
Classified Correctly = 79%				

Strict Test for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	14	171	0
	Not Best	4	2965	0
Classified Correctly = 94%				

Fuzzy Test for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	0	24	180
	Not Best	0	3960	714
Classified Correctly = 81%				

Training for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	0	17	168
	Not Best	0	2478	491
Classified Correctly = 79%				

Strict Test for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	45	140	0
	Not Best	3	2966	0
Classified Correctly = 95%				

Fuzzy Test for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	0	31	173
	Not Best	0	4045	629
Classified Correctly = 83%				

Training for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	0	18	167
	Not Best	0	2502	467
Classified Correctly = 79%				

Strict Test for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	127	58	0
	Not Best	4	2965	0
Classified Correctly = 98%				

Fuzzy Test for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	2	26	176
	Not Best	0	4111	563
Classified Correctly = 84%				

Training for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	0	17	168
	Not Best	0	2513	456
Classified Correctly = 80%				

Strict Test for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	45	140	0
	Not Best	2	2967	0
Classified Correctly = 95%				

Fuzzy Test for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	36	22	146
	Not Best	0	4034	640
Classified Correctly = 83%				

Training for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	42	15	128
	Not Best	2	2502	465
Classified Correctly = 81%				

Strict Test for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	134	51	0
	Not Best	16	2953	0
Classified Correctly = 98%				

Fuzzy Test for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	12	22	170
	Not Best	0	4053	621
Classified Correctly = 83%				

Training for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	16	15	154
	Not Best	0	2514	455
Classified Correctly = 80%				

Strict Test for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	53	132	0
	Not Best	5	2964	0
Classified Correctly = 96%				

Fuzzy Test for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	21	26	157
	Not Best	0	4084	590
Classified Correctly = 84%				

Training for h = 35

		Classified As		
		Best	Not Best	Unknown
Input	Best	5	17	163
	Not Best	0	2523	446
Classified Correctly = 80%				

Strict Test for h = 35

		Classified As		
		Best	Not Best	Unknown
Input	Best	52	133	0
	Not Best	5	2964	0
Classified Correctly = 96%				

Fuzzy Test for h = 35

Small Balanced Training Set and Two Output Nodes (h = 5 – 35)

		Classified As		
		Best	Not Best	Unknown
Input	Best	50	0	154
	Not Best	0	0	201
Classified Correctly = 12%				

Training for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	56	0	129
	Not Best	18	0	2951
Classified Correctly = 2%				

Strict Test for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	170	15	0
	Not Best	446	2523	0
Classified Correctly = 85%				

Fuzzy Test for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	56	0	148
	Not Best	1	4	196
Classified Correctly = 15%				

Training for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	53	0	132
	Not Best	12	28	2929
Classified Correctly = 3%				

Strict Test for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	169	16	0
	Not Best	436	2533	0
Classified Correctly = 86%				

Fuzzy Test for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	59	9	136
	Not Best	2	115	84
Classified Correctly = 43%				

Training for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	54	11	120
	Not Best	43	1931	995
Classified Correctly = 63%				

Strict Test for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	168	17	0
	Not Best	431	2538	0
Classified Correctly = 86%				

Fuzzy Test for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	69	7	128
	Not Best	0	113	88
Classified Correctly = 45%				

Training for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	61	10	114
	Not Best	15	1944	1010
Classified Correctly = 64%				

Strict Test for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	168	17	0
	Not Best	265	2704	0
Classified Correctly = 91%				

Fuzzy Test for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	126	5	73
	Not Best	0	112	89
Classified Correctly = 59%				

Training for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	132	2	51
	Not Best	9	1904	1056
Classified Correctly = 65%				

Strict Test for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	169	16	0
	Not Best	263	2706	0
Classified Correctly = 91%				

Fuzzy Test for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	59	7	138
	Not Best	0	112	89
Classified Correctly = 42%				

Training for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	53	11	121
	Not Best	9	1926	1034
Classified Correctly = 63%				

Strict Test for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	167	18	0
	Not Best	250	2719	0
Classified Correctly = 92%				

Fuzzy Test for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	133	7	64
	Not Best	0	113	88
Classified Correctly = 61%				

Training for h = 35

		Classified As		
		Best	Not Best	Unknown
Input	Best	139	10	36
	Not Best	19	1951	999
Classified Correctly = 66%				

Strict Test for h = 35

		Classified As		
		Best	Not Best	Unknown
Input	Best	169	16	0
	Not Best	254	2715	0
Classified Correctly = 91%				

Fuzzy Test for h = 35

Large Balanced Training Set and Two Output Nodes (h = 5 – 35)

		Classified As		
		Best	Not Best	Unknown
Input	Best	1146	0	3529
	Not Best	111	0	4563
Classified Correctly = 12%				

Training for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	45	0	140
	Not Best	58	0	2911
Classified Correctly = 1%				

Strict Test for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	145	40	0
	Not Best	308	2661	0
Classified Correctly = 89%				

Fuzzy Test for h = 5

		Classified As		
		Best	Not Best	Unknown
Input	Best	964	161	3550
	Not Best	24	2499	2151
Classified Correctly = 37%				

Training for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	41	10	134
	Not Best	9	1792	1168
Classified Correctly = 58%				

Strict Test for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	170	15	0
	Not Best	506	2463	0
Classified Correctly = 83%				

Fuzzy Test for h = 10

		Classified As		
		Best	Not Best	Unknown
Input	Best	1262	0	3413
	Not Best	9	0	4665
Classified Correctly = 13%				

Training for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	53	0	132
	Not Best	5	0	2964
Classified Correctly = 2%				

Strict Test for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	168	17	0
	Not Best	434	2535	0
Classified Correctly = 86%				

Fuzzy Test for h = 15

		Classified As		
		Best	Not Best	Unknown
Input	Best	1171	161	3343
	Not Best	50	2591	2033
Classified Correctly = 40%				

Training for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	53	10	122
	Not Best	38	1870	1061
Classified Correctly = 61%				

Strict Test for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	168	17	0
	Not Best	450	2519	0
Classified Correctly = 85%				

Fuzzy Test for h = 20

		Classified As		
		Best	Not Best	Unknown
Input	Best	871	115	3689
	Not Best	6	406	4262
Classified Correctly = 14%				

Training for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	23	0	162
	Not Best	27	97	2845
Classified Correctly = 4%				

Strict Test for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	170	15	0
	Not Best	485	2484	0
Classified Correctly = 84%				

Fuzzy Test for h = 25

		Classified As		
		Best	Not Best	Unknown
Input	Best	1240	161	3274
	Not Best	10	2482	2182
Classified Correctly = 40%				

Training for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	55	10	120
	Not Best	3	1784	1182
Classified Correctly = 58%				

Strict Test for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	170	15	0
	Not Best	446	2523	0
Classified Correctly = 85%				

Fuzzy Test for h = 30

		Classified As		
		Best	Not Best	Unknown
Input	Best	1194	161	3320
	Not Best	4	2675	1995
Classified Correctly = 41%				

Training for h = 35

		Classified As		
		Best	Not Best	Unknown
Input	Best	53	10	122
	Not Best	8	1901	1060
Classified Correctly = 62%				

Strict Test for h = 35

		Classified As		
		Best	Not Best	Unknown
Input	Best	169	16	0
	Not Best	438	2531	0
Classified Correctly = 86%				

Fuzzy Test for h = 35