Interactive Multi-Modal Robot Programming

Soshi Iba

CMU-RI-TR-04-50

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Robotics

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

May 3, 2004

Thesis Committee
Pradeep Khosla (chair)
Chris Paredis
Chris Atkeson
Katsushi Ikeuchi

Abstract

As robots enter the human environment and come in contact with inexperienced users, they need to be able to interact with users in a multi-modal fashion—keyboard and mouse are no longer acceptable as the only input modalities. Humans should be able to communicate with robots using methods as similar as possible to the concise, rich, and diverse means they use to communicate with one another.

This thesis is an investigation of how one can improve user's ability to control and program a mobile robot. The goal is a comprehensive multi-modal human-machine interface that allows non-experts to compose robot programs conveniently. Two key characteristics of this novel programming approach are that the system infer the user's intent to support interaction, and that the user can provide feedback interactively through intuitive interface, at any time. The framework takes a three-step approach to the problem: multi-modal recognition, intention interpretation, and prioritized task execution. The multi-modal recognition module translates hand gestures and spontaneous speech into a structured symbolic data stream without abstracting away the user's intent. The intention interpretation module selects the appropriate primitives based on the user input, current state, and robot sensor data. Finally, the prioritized task execution module selects and executes primitives based on current state, sensor input, and the task given by the previous step. Depending on the mode of operation, the system can provide interactive robot control and composition of robot programs.

The framework is demonstrated by interactively controlling and programming a vacuum-cleaning robot. The demonstrations are used to exemplify the interactive programming and the plan recognition aspect of the research. The key contributions of this thesis are the introduction and implementation of the novel programming approach. It is expected to improve significantly the state-of-the-art in robot programming and interactive personal robotics.

Acknowledgements

I wish to thank my co-advisors Chris Paredis and Pradeep Khosla for their tireless guidance, encouragement, patience, and support. Chris has been my big brother figure – guiding me on every aspect of research and providing feedback on most of this work, in addition to being a mentor and a friend. Pradeep has been my father figure – encouraging me throughout the years of my doctoral education, and providing invaluable advice not necessarily limited to the scope of this work.

I also want to thank the other members of my thesis committee, Chris Atkeson and Katsushi Ikeuchi for their feedback and kind help. I am grateful to Chris for his frequent encouragements and comments. Katsu has been my mentor for a long time in both my personal and academic development.

I would like to thank the members of the Advanced Mechatronics Lab, which has been an extraordinary environment to work in. The people in the lab remind me of how much I have yet to learn to become a skilled roboticist. I would also like to thank many members of the Robotics Institute who have helped me reach this milestone, including some of my classmates and friends over the years.

Lastly, I would like to thank my family back home for their support and encouragement. Most of all, thanks to my wife Risa, for her love and support. It would not have been possible to survive through the final few semesters without her care.

Table of Contents

List of Figures

List of Tables

Chapter 1.

Introduction

1.1. Motivation

An important aspect of a successful robotic system is the human-machine interaction. As robots enter the human environment and come in contact with inexperienced users, they need to be able to interact with users through a novice friendly interface and ease the burden of knowledge transfer from the user to the robot. In terms of human-machine interface, interaction should be done in a multi-modal fashion—keyboard and mouse are no longer acceptable as the only input modalities. Humans should be able to communicate with robots using methods as similar as possible to the concise, rich, and diverse means they use to communicate with one another. In terms of human-machine knowledge transfer, an expert in the task who is not necessarily a robot programmer may need to rely on a robot programming expert to convey his knowledge to the system. Humans should be able to transfer knowledge without relying on a robot programming expert.

The goal of this work is to create a Programming by Interaction (PBI) system that enables novice users to control and program a robot interactively through an intuitive interface. Empirical user study is conducted to investigate the contribution of the components to a superior robot programming experience and performance provided by the PBI system. The key elements behind this novice-friendly system are intuitive interfaces based on speech and hand gesture recognition, the system's intention awareness that models, recognizes and makes suggestions based on the user's intention, and interaction capabilities that allow the user to take over the control of the robot at any given time. The PBI system is similar to the WYSIWYG (*what you see is what you get*) interface introduced in the human-computer interaction domain. Instead of off-line robot programming, the PBI system lets the user see on-line, what to expect from the program execution and how to make adjustments.

1.2. Goals

The goal of this work is to create a Programming by Interaction (PBI) system that enables novice users to control and program a robot interactively through an intuitive interface. We also verify the system's advantage in programming experience and performance through empirical user study on mobile vacuum cleaning robot.

The intuitive interface based on hand gesture and spontaneous speech recognition provides simple means to convey symbolic and parametric data in three-dimensional space. A teach pendant, connected to a robot controller used to direct and program a robot, provides interface for accurate but cumbersome parameter specification, and is not novice-friendly. The use of graphical user interface (GUI) to control a robot through keyboard and mouse for both on-line and off-line robot programming has been widely used, but the GUI does not always reflect the accurate state of the environment, and besides, it is two-dimensional in nature. Direct interaction with robots based on hand gestures are suitable for three-dimensional space where it can be used to specify location, size, velocity, etc. almost like a multi-purpose mouse. Spontaneous speech can provide symbolic commands to the system.

The system with intention awareness models, recognizes and makes suggestions based on the user's intention. The user's intent is captured in the form of a sequential robot program, and the flexibility given to the user through real-time interaction and the framework's intuitive interface allows the captured intent to be closer to what the user really expects from the robot. Sequential robot programs are converted to statistical models so that partial inputs from the user can be used by the system in future to recognize the robot program that the user may want to execute. The suggestion is made through a graphical display, where a simulated robot executes the task so that the user can choose to accept or decline the offer.

Interaction capabilities give a sense of assurance to the users and help them in dealing with disparity between real and modeled environment, which may have been caused by loosely calibrated position sensors, by including a human in the control loop. Users are able to initiate a programming phase through voice commands and move the robot to any desired location. The sequence of commands turns into a sequential robot

program. The user can then initiate an execution phase and execute the program while taking control at any given time.

The research questions addressed in this work are the following:

- How can one make robot programming easier?
  - "Do multi-modal user interface improve user's ability to program a robot?"
  - "Does program suggestion based on intention recognition help robot programming?"
- How can one simplify the interaction with mobile robots?
  - "Does direct interaction improve user's ability to control a robot?"

Through answering these questions, I am making contributions to the areas of human-robot interaction and robot programming.

1.3. Contributions

The contributions I am making through this work are in the area of human-robot interaction and robot programming. The main contributions are the following:

- Introduction of Programming by Interaction (PBI) paradigm as a method to enable human-robot knowledge transfer (Iba et al. 2002).
- Design and implementation of the PBI framework (Iba et al. 2002).
- Development of the algorithm which enables program suggestion by the system based on intention awareness (Iba et al. 2003).
- Empirical user study, which investigates the benefits of the PBI framework in the domain of mobile vacuum cleaning robot control and programming.

These contributions facilitate a novice-friendly robot programming system.

1.4. Thesis Organization

This thesis is an investigation of how one can improve user's ability to control and program a mobile robot. The organization of the thesis corresponds to the overall framework of the system as described on Figure 1. The thesis is organized as follows:

In Chapter 2, I present a review of the works in the areas related to this thesis. Operating and controlling robots using multi-modal interface, such as eye gaze, hand-gestures, spontaneous speech, haptics, etc. have been an area of interest to many researchers. In particular, works on hand-gesture recognition and their use for robot control are discussed in detail. Discussion of the works in the area of robot programming follows, in order to provide a sense of what was done in the past and where the field is trying to go. The discussion is provided in both industrial and personal robot domains. Related to robot programming are the works on intention modeling and interpretation, since robot programming can be thought of as a form of transferring knowledge and intent of users to robots. Previous works on intention interpretation and their use for robotics are discussed.

In Chapter 3, I provide a description of the system design and overall framework. The framework is roughly divided into three modules: multi-modal recognition, intention interpretation, and prioritized task execution, each roughly corresponds to providing intuitive interface, robot programming and suggestion, and interactive control capability of the overall system. The actual system setup is described, which includes the graphical user interface (3.4) used for robot control, iconic programming and programming by interaction demonstration, the vacuum cleaning mobile robot (3.3), and the input devices used to

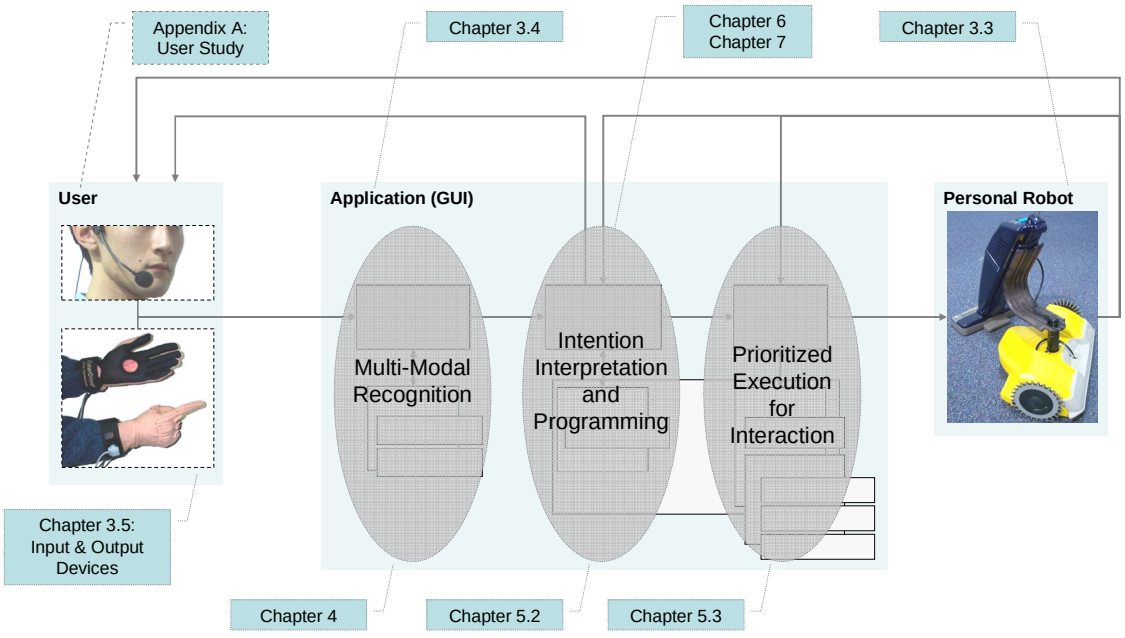


Figure 1: Correspondence between the Framework and Chapters

collect joint curvatures and positions of the user's hands (3.5).

In Chapter 4, I present the method and implementation of the multi-modal recognition module of the PBI framework. The module consists of two different recognition modes: hand-gesture recognition and spontaneous speech recognition. The hand gesture recognition takes temporal data stream from data gloves and inductive position sensors to recognize spontaneous hand gestures in real time using statistical modeling and recognition techniques. Gesture vocabularies are selected analytically, whereas training, recognition, and adaptation of the models are based on the stochastic technique. The spontaneous speech recognition is implemented on the public domain large volume speech recognition engine. Temporal stream of results from hand gesture recognition and spontaneous speech recognition is combined to generate a semantically correct interpretation to control and program the robot

In Chapter 5, I describe the preemptive execution feature of the interactive multi-modal robot programming framework. The preemptive execution is crucial to providing the user a real-time interaction to control and program a robot on-the-fly. Tasks are prioritized according to the pre-defined rule, and sequential robot actions in the tasks are executed and sometimes overtaken based on their priorities.

In Chapter 6, I present the method and the implementation of the framework on modeling and inferring user's intention from a temporal stream of robot actions. In the framework, user's intention is captured in a form of robot program. The flexibility given to the user through real-time interaction and the framework's intuitive interface allows the captured intent to be closer to the user's true intent. The user's intention is classified into two categories: task level intention and primitive level intention. I describe modeling, updating, and recognition methods for both categories.

In Chapter 7, I describe two demonstrations conducted to verify the interactive multi-modal robot programming and execution framework. Two sequential programming scenarios are demonstrated: point-to-point navigation and area coverage, to clearly illustrate the usefulness of multi-modal interaction

In Chapter 8, I conclude the thesis by going over the overall system design of the multi-modal interactive robot programming framework, and the implication it has on

novice friendly human-robot interaction and knowledge transfer in a form of robot programming.

In Appendix A, I describe the procedure and results of the user study conducted to investigate the benefits of the interactive multi-modal robot programming paradigm. The user study was conducted to compare the performance of the system built on the new paradigm against iconic programming system. Users are asked to perform vacuum cleaning tasks in the laboratory environment using the same vacuum cleaning robot and the performance was measured in time, user satisfaction, and the amount of trash collected. This study is added as an appendix since it is a test of the part of the system and it does not necessarily reflect the performance of an entire system, although it can be used to implicate the performance.

In Appendix B, I present the support documents submitted to the Institutional Review Board that were required to conduct the user study described in Appendix A.

In Appendix C, I present the derivation of a sample merging procedure used to adapt program models to incoming observations, which is described in Chapter 6.

1.5. Summary

The motivation behind this work is the creation and investigation of the novice friendly human-robot knowledge transfer paradigm. Programming by Interaction (PBI) paradigm is introduced, which is realized by a combination of intuitive interface, intention aware robot programming, and preemptive human-robot interaction. The contributions of this thesis are described as: (1) the introduction and implementation of PBI paradigm as a method to enable human-robot knowledge transfer (2) the development of the algorithm enabling the system to be aware of user's intention (3) the evaluation of the system through empirical user study.

Chapter 2.

Related Works

I describe the research related to this thesis. The area of human-robot interaction is a rich and diverse field of study. In order to understand the work of controlling and programming robots through a multi-modal interface, this chapter is divided into subsections each describing the research field associated with the Interactive Multi-Modal Robot Programming system.

2.1. Introduction

Interactive Multi-Modal Robot Programming system created in this work is a realization of Programming by Interaction paradigm that enables novice friendly robot control and knowledge transfer through interactive robot programming. This work is related to numerous research fields such as multi-modal robot control, robot programming, and intention recognition.

2.2. Multi-Modal Robot Control

The area of robot control refers to the problem of efficiently conveying control signals to the robot system. Every robot system must have a device through which its user can control the behavior of the robot. The control signal can be in various forms ranging from low-level joint motor torque to high-level symbolic skill representations. For both mobile robots and industrial manipulators, the basic level of control is in the joint space, where the user input often comes from a teach-pendant or a joystick. At the higher level of abstraction, the control specifications are symbolic and come from either a graphical user interface or a natural user interface such as eye gaze tracking, finger pointing, or natural language interpretation.

Multi-modal Interface: From the perspective of multi-modal interfaces, (*e.g.* gestures, speech) the interaction between the user and the robot systems has many advantages over conventional interaction modes, such as teach-pendants or joysticks.

Hand gestures have an advantage in specifying geometric objects and spatial (three-dimensional) data (Quek 1994), and are more intuitive for conveying information to robots that exist in the three-dimensional world (Skubic et al. 2002). The advantage is even more obvious when interacting with a team of robots, where complicated maneuvers and grouping commands can be executed by gesturing a set of points, a region of interest, or a group formation (Perzanowski et al. 2002). Hand gestures are convenient for specifying parametric and 3D information, but not for symbolic gestures. For symbolic information and commands, speech input is a natural choice. In comparison to the GUI used in personal computers, hand gesture can be a superset of a mouse, and speech can be a superset of a keyboard.

Hand gesture recognition is a popular field due to its broad applicability. Many successful gesture recognition methods are derived from algorithms in natural language recognition. They are roughly divided into three approaches: template-based, stochastic, and neural net based approaches. Nishimura and Oka (Nishimura et al. 1998) used template based continuous dynamic time warping (DTW) for spotting continuous visual gestures. The mobile robot interaction system by Kuno et al. (Kuno et al. 2000) also used a gesture-spotting strategy based on DTW. Starner (Starner and Pentland 1995) applied Hidden Markov Models (HMM; often used to model doubly stochastic processes) to visual hand recognition of dynamic American Sign Language (ASL). Lee and Xu (Lee and Xu. 1996) used a similar HMM based method to recognize static ASL alphabets with a data glove as an input device. Kortenkamp et al. (Kortenkamp et al. 1996) developed a model-based method which models different parts of the body as a set of proximity spaces and defines pose gestures by examining the angles between the links that connect these proximity spaces. Waldherr et al. (Waldherr et al. 2000) combined a neural net approach for static pose gestures with a temporal template matching approach for motion gestures. Yang et al. (Yang et al. 2002) adopted TDNN (Time Difference Neural Network) based approach to recognize dynamic hand gestures using motion trajectory extracted from segmented temporal images. They all differ in their assumptions, implementations (vision vs. magnetic

spatial sensor, controlled lighting/background condition vs. mobile robot's on-board camera), and capabilities (pose vs. motion gesture, recognition rate), and it is important to keep in mind that their advantages and disadvantages are task dependent.

Several researches have implemented a variety of natural interface to control mobile robots. The *GestureDriver* and *HapticDriver* systems by Fong (Fong 2001) provide a teleoperation interface through symbolic hand gestures and force feedback through a haptic device. Other mobile robot interactions systems are capable of receiving symbolic gesture commands through an on-board camera (Boehme et al. 1997; Kortenkamp et al. 1996; Waldherr et al. 2000). Kuno et al. (Kuno et al. 2000) have developed a wheelchair robot controlled by detecting hand gestures with a camera. This system is capable of dealing with unknown gestures by considering all periodic hand motions as potential gestures. Another example is Matsumoto's wheelchair robot (Matsumoto and Zelinsky 2000), which can detect the user's gaze and facial direction to navigate.

To move a step closer to the human-human interaction, researchers are currently exploring multi-modal interaction scenarios. The advantage of working with multi-modal input mainly lies in its redundancy. For example, the system developed by Perzanowski et al. (Perzanowski et al. 2001) combines natural language and hand gestures to interpret both complete and fragmental commands. The multi-modal interface system by Ghidary et al. (Ghidary et al. 2001) makes use of speech, posture, and object recognition to navigate a mobile robot to an object of interest. Human-robot interaction can become more intuitive as the level of flexibility in the human interface increases. However, to achieve a higher level of human-robot interaction, the human interface and robot programming modules must work together.

A multi-modal interface combines multiple input modalities such as natural speech, pen-based input, hand gestures, facial gestures, eye gaze, body language, or tactile input. In the past, before robust multi-modal approaches were available, skeptics believed that a multi-modal interface incorporating two error-prone recognition technologies would compound errors and yield even greater unreliability. However, recent data shows that fusing two or more information sources can effectively reduce recognition uncertainty, thereby improving robustness (Oviatt 2000).

2.3. Robot Programming

Over years, we have seen evolution in robot programming language and methods. There is a difference in trends between industrial and mobile robots, but overall, they are leading to more user-friendly and intuitive methods.

Industrial Robotics: Since the early years of robotics, industrial robots have been programmed on text-based programming language. Gruver et al. (Gruver et al. 1984) provides the list and the general overview of the industrial and research robot programming languages available. More sophisticated robot programming language, such as ABB's Rapid (Rapid 1994), is in common use but they require specialized knowledge of the language and the programs have limited portability (Figure 2). Up to date, teach pendants, connected to a robot controller used to direct and program a robot, have been the most common mode of interaction, although there have been attempts to provide interactive

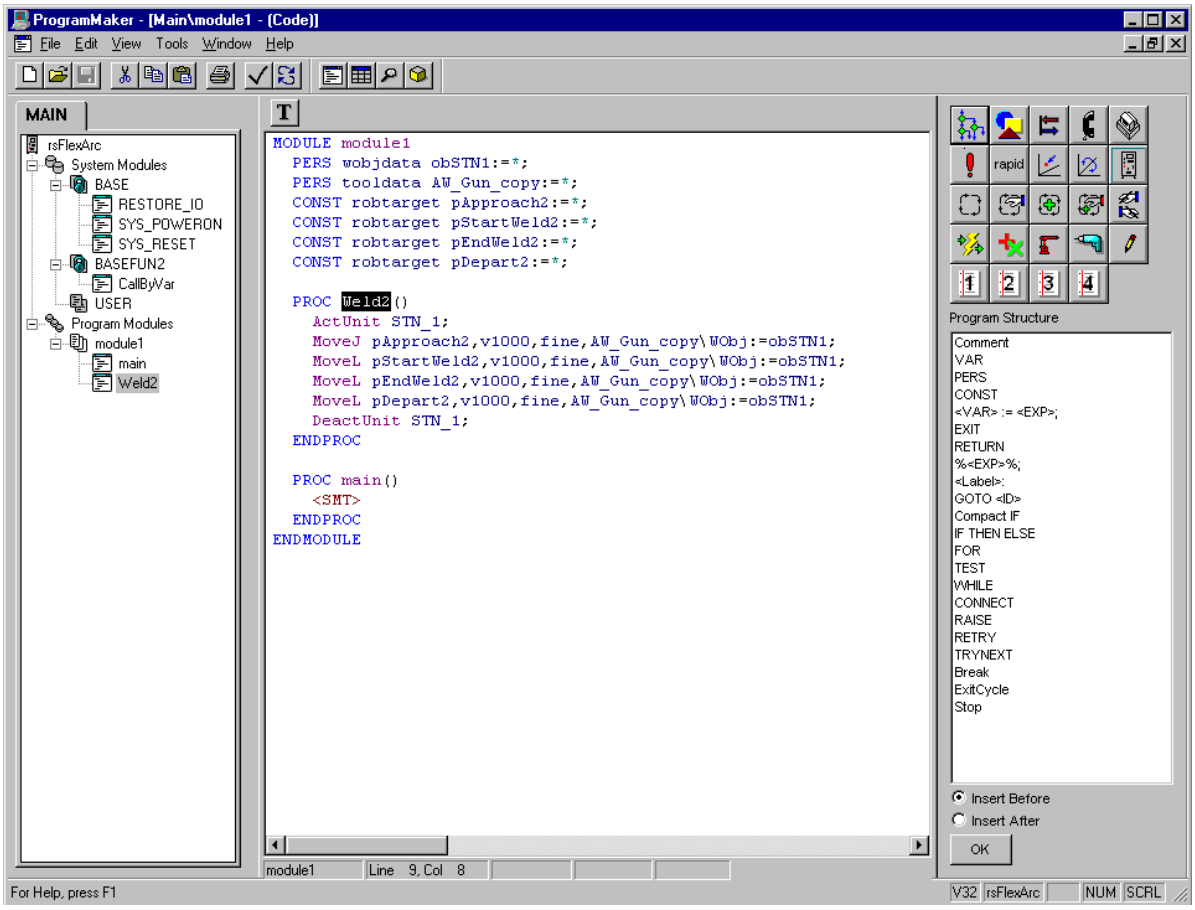


Figure 2: Example of a Text Based Robot Programming (ABB RAPID)

programming framework on text based interface (Jayaraman and Deisenroth 1987). As software capabilities improved, the ability to do off-line programming proved to be a significant step forward. Interfaces to manipulator systems made further progress with the introduction of user friendly programming paradigms for sensor-based manipulation (Morrow and Khosla 1997). The current state-of-the-art in manipulator interaction is based on iconic programming (Gertz and Khosla 1994) and/or programming by human demonstration (Ikeuchi and Suehiro 1994). The goal of these paradigms is to translate the burden of programming manipulator systems from robot experts to task experts. Task experts have extensive knowledge and experience with respect to the task, but may only have limited expertise in robotics. To enable these novice users to interact with the robot, the interface needs to be intuitive and have the ability to interpret the vague specifications of the user. An example of such a system is the gesture-based programming interface developed by Voyles and Khosla (Voyles et al. 1997). The robot system observes the operator unobtrusively while she is demonstrating the task. The observations can be based on vision, range sensing, data gloves, or tactile sensing.

Personal Robotics: Due to the growing field of personal robotics, we encounter more robots than ever before. Examples of such robots include pet robots (Fujita 2001), tour-guiding robots (Thrun et al. 1999), entertainment robots (Ishida 2003), intelligent wheelchairs (Rao et al. 2002), and mobile vacuuming robots (Musser 2003; Schofield 1999). Traditionally, mobile robots are controlled via a joystick or mouse, but increasingly, voice or gestures are included as input modalities (Boehme et al. 1997; Rogalla et al. 2002). We explore the task of interactively controlling and programming a vacuum-cleaning robot called Cye (Batavia and Nourbakhsh 2000) in this thesis. This task requires both interactive multi-modal control and a certain degree of autonomy. To accommodate novice users, the programming framework utilizes multi-modal interaction (hand gestures and voice commands) and encompasses preemptive interaction during both programming and execution.

Gesture Based Programming: In addition to using gesture-based interaction for direct control of robots, it can also be used for robot programming. Position and path-based applications such as arc welding and machine loading typically employ walk-through or lead-through teaching (Todd 1986). For walk-through teaching the user specifies

intermediate points with a teach pendant. For lead-through teaching, the user performs the required motions manually while holding some device (the manipulator itself, or a replica) to record the path. While these forms of teaching are useful for non-contact applications; other methods are needed for applications that involve contact. Kang and Ikeuchi's (Kang and Ikeuchi 1997) learning-from-observation system models human behavior as transitions of contact states, by observing a human demonstration. The system is able to model high-level task specifications but not the sensor feedback during contact. Voyles et al. (Voyles et al. 1999) proposed a gesture-based programming paradigm where the system is assumed to have a set of basic skills (also referred to as *a priori* control policies (Kortenkamp et al. 2001), or sensori-motor primitives (Morrow and Khosla 1997)) from which the system can compose programs. The system observes human demonstration through gesture recognition and interpretation agents, and selects correct skills based on the votes from the agents. A similar skill-based approach is used in the telerobotics system by Onda et al. (Onda et al. 2000) that combines geometric modeling, teaching by demonstration in a virtual environment, and execution based on manipulation skills. Programming based on observed human demonstration is called by several names, including Gesture Based Programming, Programming by Demonstration, or Learning by Observation. An overview and classification of these systems can be found in (Dillmann et al. 1999).

To achieve robot interaction at elevated conceptual levels, robot programs can be composed from primitive behaviors. Such composition of skills can either be prepared in advance or learned from observation. Asada's human-robot interaction system uses Petri-nets to model the interaction between the robot and the human, but a plan has to be prepared in advance by a programmer (Mascaro and Asada 1998). Kimura and Ikeuchi (Kimura et al. 1999) model human-robot cooperation tasks by observing both parties and placing pre- and post-conditions into a stack to compose a program. For its humanoid application, Kawamura's DBAM architecture (Kawamura et al. 2000) captures similar pre- and post- conditions into a look-up table.

Programming by Interaction: The multi-modal interactive programming framework has several distinct advantages over conventional methods. From the robot programming perspective, on-line interaction adds a new flavor to the robot programming problem.

- It enables novice users to program robots,
- It enables interactive composition of primitives to create robot programs,
- It enables task model adaptation through continuous interaction.

To some degree, other paradigms such as iconic programming (Gertz et al. 1993; Nagchaudhuri et al. 2002), and programming by demonstration (Ikeuchi and Suehiro 1994) succeed in shifting the burden of robot programming from robot experts to task experts. However, due to the current lack of understanding of intention interpretation and of the robotic task itself, such off-line programming methods are very fragile. The task expert may demonstrate the task to the robot, but the task expert has no idea how the robot has interpreted his skill, or whether the robot has a sufficient set of actions to perform the demonstrated task. In contrast, our framework allows the task expert to "coach" the robot and to make adjustments on-line as it performs the new task. One of the examples of the interactive programming system is the interactive training system by Natakani et al. (Nakatani et al. 2003) that learns controller parameters for a biped robot by utilizing human knowledge and evaluation. The human who is in the control loop of the training system may observe the robot's performance and give subjective evaluation at any point of time. The system may adjust its controller parameters based on the evaluation, and the process repeats itself until the human is satisfied with the performance. As seen on Nakatani's system, Programming by Interaction is useful when qualitative evaluation of the system performance is difficult. Interactive Robot Programming system for service robots by Friedrich et al. (Friedrich et al. 1999) is a Programming by Demonstration paradigm with capabilities to supervise and influence the process of program generation after the initial demonstration. It is very close to what my work achieves, with a difference in where the interaction takes place. The interaction in the system by Friedrich et al. takes place in the simulator, whereas my system provides interaction in the real environment.

2.4. Intention Interpretation

The most challenging aspect of interactive robot programming is to interpret the intent of the users, rather than simply mimic their actions. Intent is the purpose or goal the user has in mind. User input can be vague, inaccurate, and often contradicting. An intention aware system can be used to reduce unnecessary and often redundant instructions by being

aware of what the user really wants. Classical example is Yared and Sheridan's system (Yared and Sheridan 1991) that used generalized symbolic planning approach to infer intent from task directed robot manipulation program. Yamada et al. (Yamada et al. 1999) created a system using Hidden Markov Models to model manipulator's trajectory controlled by a human operator to infer the operator's intended path in order to reduce the load felt by the operator. Dixon (Dixon 2004) developed a learning by observation system based on the combination of Sequenced Linear Dynamical Systems and Continuous Density Hidden Markov Models to learn motor skills by observing users demonstrating a task. The system is capable of inferring what the user would have done in different conditions. Intention interpretation can be thought of as a search for the mapping from the user input and robot sensory data to the correct set of robot actions. To accomplish such interpretation, the user needs an intuitive mode of interaction with the robot, while letting the system collect additional data leading to the correct intention interpretations. The term intent is often loosely defined since it is very task dependent. In our framework, intention refers to a sequential robot program that the user would like to execute or modify, and the system needs to determine from inputs given by the user if such a robot program exists in the system's database. In other words, the user's intent is captured in the form of a sequential robot program, and the flexibility given to the user through real-time interaction and the framework's intuitive interface allows the captured intent to be closer to the user's true intent. Previous work on intention-aware systems such as (Agah and Tanie 1996; Voyles et al. 1997) lacks this flexibility, and our system is more robust by being aware of a user's intent and incorporating real-time alterations based on this information.

2.5. Summary

The area of human-robot interaction is a rich and diverse field of study. Interactive Multi-Modal Robot Programming is a combination of multiple research fields such as multi-modal robot control, robot programming, and intention recognition.

Chapter 3.

System Design

I present the overall system design of the multi-modal interactive robot programming framework. The framework takes a three-step approach to the problem: multi-modal recognition, intention interpretation, and prioritized task execution, each implemented as a separate module. The modules are implemented on top of the commercial iconic control and programming application. The framework provides additional capabilities such as interactive control and programming through intuitive interfaces to the application. The vacuum cleaning mobile robot and the input devices of the system are also described in this chapter.

3.1. Introduction

The Multi-Modal Robot Programming approach described in the thesis offers an intuitive interface for the user and the ability to provide interactive feedback to coach the robot throughout the programming process. The approach addresses shortcomings apparent in previous approaches, which are an unfriendly user interface preventing a novice user from using a service robot, and an inability to teach and program a robot on-the-fly. As input modalities, we support hand gestures and spontaneous speech. We selected hand gestures as a modality to convey parametric information such as speed, angles or positions, and spontaneous speech is selected as a modality to convey symbolic information such as names, confirmations, or program statements. The selection is made based on intuitiveness of the modality.

In order to make a comparison between the current state of the art in mobile robot programming system and the interactive multi-modal robot programming system, it is desirable to add new features on an existing robot programming system.

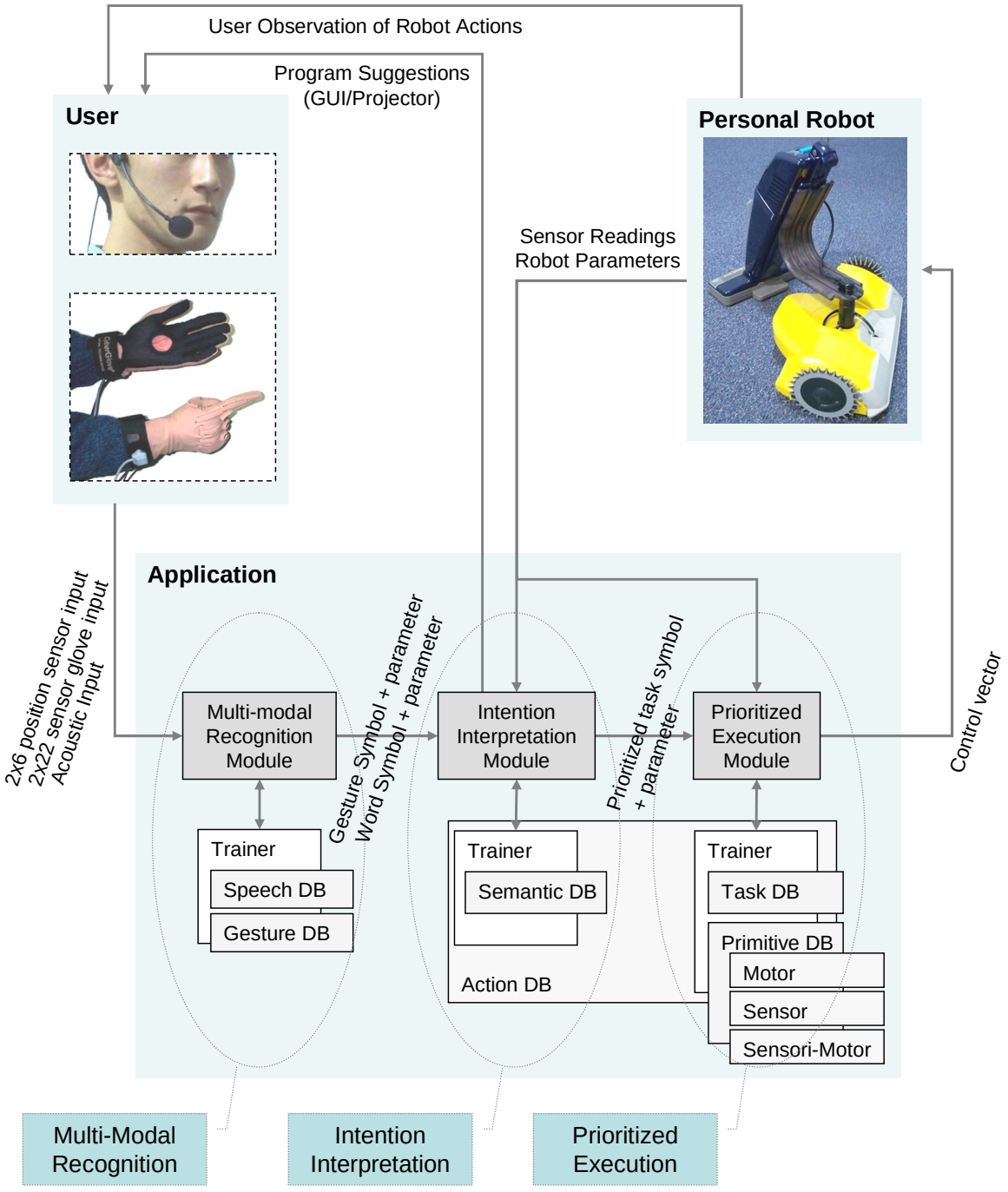


Figure 3: Framework for the Interactive Multi-Modal Programming system

3.2. Overall Framework

The framework is composed of three functional modules as illustrated in Figure 3. The first module (multi-modal recognition) translates hand gestures and spontaneous speech into a structured symbolic data stream without abstracting away the user's intent. The second module (intention interpretation) selects the appropriate set of primitives based on the user input, current state, and robot sensor data. Finally, the third module (prioritized task execution) selects and executes primitives based on the current state, sensor inputs, and the task given by the previous step. Each module includes two modes of operation: a learning and an execution mode. Depending on the mode of operation, the overall system can provide interactive robot control, adjustment of primitives, or composition of robot programs.

There are three main reasons for implementing the system in a modular fashion as described in Figure 3. First, the implementation follows a functional decomposition of the problem: recognition, interpretation, and execution. Second, in a modular architecture, one can easily replace the implementations of individual modules. For example, if we were to program an industrial manipulator instead of a vacuum cleaning mobile robot, the task execution module can be replaced by another implementation. Finally, because the first and

Module	*Input*	*Function (Execution and/or Learning mode)*	*Output*
Multi-modal Recognition	CyberGlove-R Polhemus-R CyberGlove-L Polhemus-L Acoustic (8bit-16KHz)	• Translate incoming audio and gesture signals into a structured stream of word and gesture unit symbols with appropriate parameters. (*E*) • Reinforce models during recognition (exec. & learn)	Gesture Symbol-R + param. Gesture Symbol-L + param. Word Symbol + param
Intention Interpretation	Gesture Symbol-R + param Gesture Symbol-L + param Word Symbol + param Robot Data Robot Position Robot Velocity Sensor Readings Knowledge of its current state	• Select the appropriate primitives based on the user input, current state, and robot sensor data. (*E*) • Prioritization of tasks, according to the database (*E*) • Adapt task model used for selections using the most current observations (exec. and learn)	Task symbol + priority + param
Prioritized Task Execution	Robot Status Sensor Readings Task symbol + priority + param	• Arbitrate and execute primitives based on current state, sensor input, and the prioritized task given by the previous module. (*E*) • Generate a robot program (task) by configuring primitives. (*E* & *L*)	Control vector

Table 1: Functional Summary

last module can be implemented as slight modification of existing software and hardware products, a modular implementation allows us to work independently on the intention interpretation module, which is the focus of this research. Table 1summarizes the functions offered by each of the three modules in the framework. The three modules work synchronously in a continuous flow of data for providing intuitive multi-modal interaction and programming of robots.

3.3. Vacuum Cleaning Robot

There are increasing number of vacuum cleaning robots being manufactured and sold to household consumers (Musser 2003). Depending on the price range, these robots can navigate, avoid obstacles, localize themselves and cover the area autonomously. Cye (The Robot in Figure 3) is a two-wheeled mobile vacuum cleaning robot (Batavia and Nourbakhsh 2000). The user can program Cye off-line using iconic programming and communicates with the host computer through wireless serial channel. The size is 10”×16” (L×W), which is close to the width of the head of a small vacuum cleaner. Sensing capability of the robot is limited to odometry and the current sensor on the motors, so the robot detects obstacles from collision, and keeps its global position through deduced reckoning, making occasional calibration using its home position necessary. The robot carries a vacuum cleaner which can be turned on and off from the robot.

3.4. User Interface Application

Map-N-Zap is an open source iconic programming framework provided by Probotics, Inc (Figure 4). It is capable of controlling Cye mobile vacuum cleaning robot, and simultaneously creating a probability grid map of the environment using deduced reckoning. Cye detects obstacles from collision. The user is able to create robot programs using the iconic programming interface, and execute programs. In order to build the interactive multi-modal robot programming system, I added several features to Map-N-Zap, which include:

- Hand gesture and spontaneous speech recognition interface
- Capability to preemptively interrupt a set of instructions
- Capability to create and adjust the program on the fly
- Simulated robot for users to control and execute a program in the virtual environment
- Capability to suggest a program that the user may want to execute, based on the sequence of robot trajectory. A program suggestion is presented to the user by executing the program on a simulated robot.

3.5. Input and Output Device

Two 22-sensor CyberGloves (CyberGlove 1998), and Polhemus 6DOF position sensors track hand joint measurements and positions of wrists in global reference frame. CyberGlove and Polhemus 6DOF position sensors generate data stream at 60Hz. A Polhemus 6DOF positioning system consists of the transmitter and the receiver pair. The

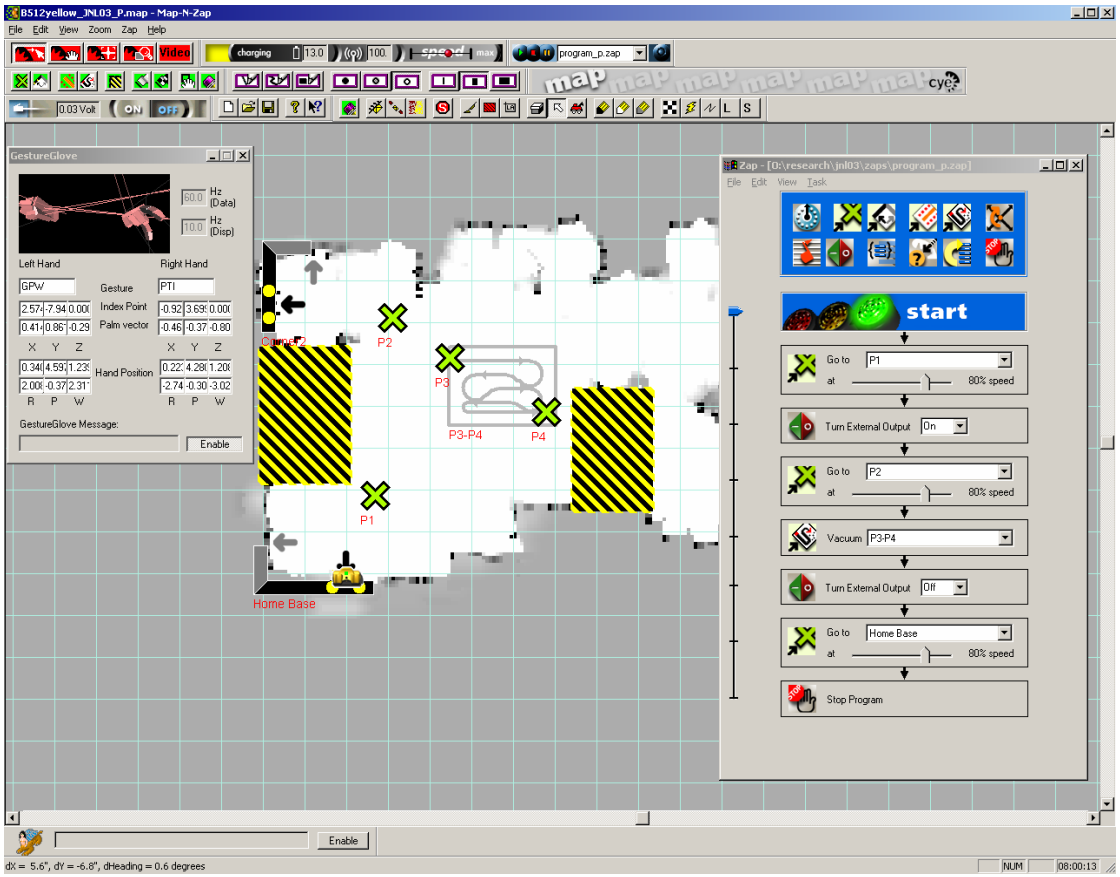


Figure 4: Map-N-Zap Screen Shot

receiver placed on the user's hand detects the magnetic fields emitted by the fixed transmitter and inductively tracks the position and orientation.

LCD projector is used to display the graphical user interface on the wall (Figure 5), so that the user can examine the current situation easily and receive program suggestion from the system regardless of whether the user is operating on mouse or multi-modal interface.

3.6. Summary

The framework is composed of three functional modules: multi-modal recognition, intention interpretation, and prioritized task execution. CyberGloves and Polhemus 6DOF position sensors provide intuitive input modality to the user, and the rest of the modules are built on top of Map-N-Zap, an open-source robot control and iconic programming application. The first module (multi-modal recognition) recognizes gesture and spontaneous speech from CyberGloves and microphone. The second module (intention interpretation) interprets multi-modal recognition result to a task. The third module (prioritized task execution) executes the task and sends control vector to Cye. The user may confirm the

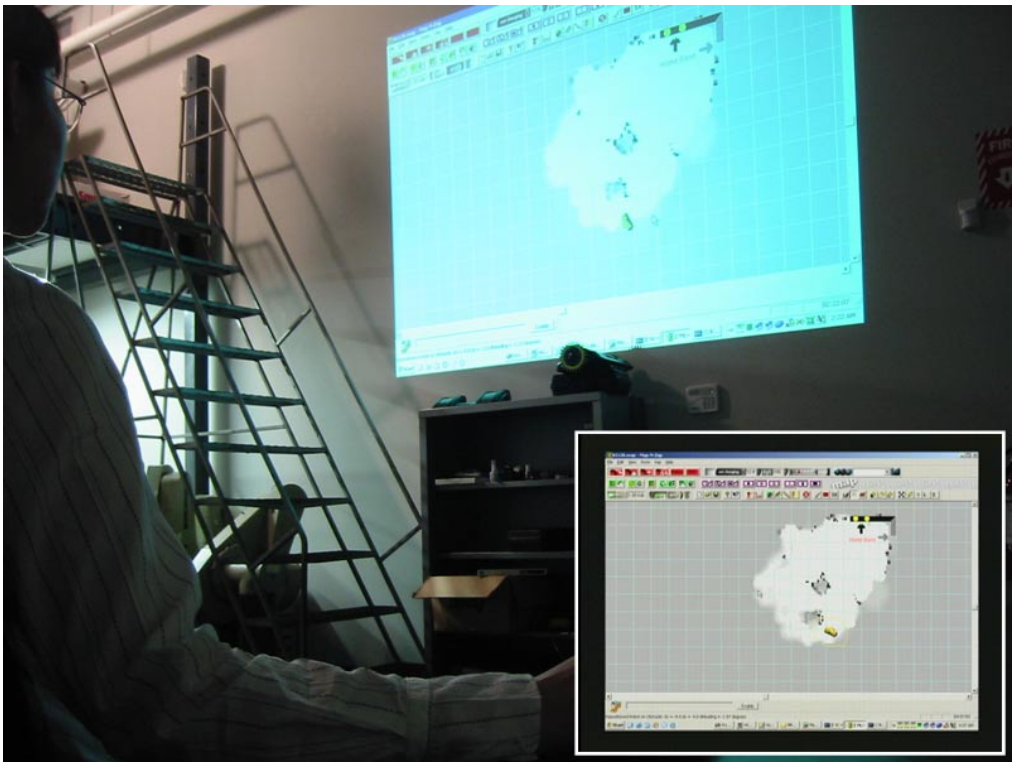


Figure 5: GUI projected on the wall

execution either by observing Cye, or from the graphical user interface which is projected on the wall.

Chapter 4.

Multi-Modal Recognition

I present the implementation of the multi-modal recognition module of the interactive multi-modal robot programming framework. Two key requirements of the multi-modal recognition module are recognition speed and accuracy necessary for real-time human-robot interaction. The module consists of two different recognition modes: hand gesture recognition and spontaneous speech recognition. The hand gesture recognition takes temporal data stream from data gloves and inductive position sensors to recognize spontaneous hand gestures in real-time using statistical modeling and recognition techniques. Gesture vocabularies are selected analytically. Training, recognition, and adaptation of the models are based on the stochastic technique. The spontaneous speech recognition is implemented on the public domain large volume speech recognition engine. Temporal stream of results from hand gesture recognition and spontaneous speech recognition is combined to generate a semantically correct interpretation to control and program the robot.

4.1. Introduction

Humans communicate with and influence fellow humans in a wide variety of ways, primarily through verbal, tactile, and motion cues. They take advantage of the presence of these multiple "modes" by resolving the ambiguity in one mode with information from another (Oviatt 2000). Redundancy and cross-cueing of senses in each person provide for robust interaction, and it is desirable that human-robot interaction should be carried out in a similar way. Indeed, multi modal interfaces between human and robotics systems are beginning to receive attention in the literature (Perzanowski et al. 2001).

The primary motivation for multi-modality is that no single mode provides a highly competent human-robot interface. Verbal cues are most appropriate when either party needs to convey symbolic information with an unambiguous context, such as "stop", "I cannot

move", "hold on to the beam", etc. Motion cues are an essential supplement when deictic elements are involved, as in the verbal commands "go *there*", "move *this* way", and "pick up the object of *this* size". Deictics are linguistic strings containing demonstrative words such as "this', "that", and "those", referring to objects in the real world. They are also known as determiners or "pointer words", so called because they function by "pointing" to something in the context. Such instructions are ambiguous without the gestures that are inherently more suitable to express position and geometry.

In the context of interactive multi-modal robot programming framework, the multi-modal recognition module is essential to providing an intuitive interface to users interacting directly with a mobile robot. Users have a choice to interact directly with a robot through hand gestures and spontaneous speech instead of indirect interaction through a graphical user interface (GUI) using mouse and keyboard (Figure 6). Having direct multi-modal interface is desirable to human-robot interaction for two reasons: First, it is often true that the representation of the environment on a GUI differs from the state of the real environment. In such case, it is necessary for the user to give commands to a robot in the real environment directly instead of giving commands on a GUI and constantly check what the robot is doing in the real environment. Secondly, the interface based on hand gestures combined with spontaneous speech is more natural and capable of conveying spatial information than conventional interface on a GUI.

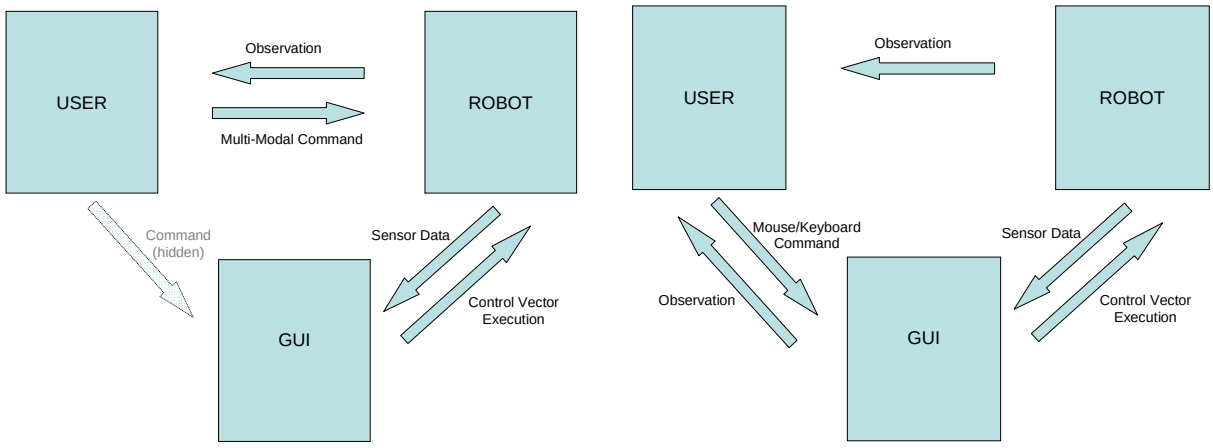


Figure 6: Direct Multi-Modal Interaction vs. Indirect GUI Based Interaction

In this chapter, I describe the method and implementation of the hand gesture recognition and spontaneous speech recognition, and how their results are combined as a multi-modal recognition result. I first describe what is involved in hand gesture recognition: what hands are capable of doing, how to select the set of hand gesture vocabulary, how to train and model the vocabulary, how to recognize the hand gestures, how to extract parameters from the hand gestures, and how the hand gesture recognition is implemented in the module. Then I describe the vocabulary selection and the implementation of spontaneous speech recognition. Since the spontaneous speech recognition is implemented using the commercially available large vocabulary continuous speech recognition engine, exact detail of the recognition method is unavailable. Finally, I describe how recognition results from hand gesture and spontaneous speech are integrated into a multi-modal recognition result.

4.2. Hand Gesture Recognition

Human hand is a diverse tool that can be used for manipulation, communication, emotional expression, noise generation, etc. Various aspects of grasping by human hand is described in (Shastri and Iberall 1990). (Kamakura 1989) also provides diverse set of grasps from occupational therapist's viewpoint, where she describes sets of dynamic hand actions as combinations of primitive finger motions.

In this thesis, I concentrate on the communication aspect of a human hand, in particular the use of hand gestures for spatial communication. A hand gesture is a meaningful part of the hand motion that can be used to express both symbolic and parametric information. We selected hand gestures as a modality to convey parametric information such as speed, angles or positions in three-dimensional space where human-robot interactions take place.

Key requirements for our framework are speed, accuracy, and adaptability. Speed is required for the parameters to be extracted and transferred to the robot that is moving in real-time. Accuracy is required in two ways: recognition accuracy, and parameter extraction accuracy. When it comes to the comparison of two programming paradigms, interactive multi-modal programming framework against conventional iconic programming framework, it is crucial that shortcomings in gesture recognition and parameter extraction

from hands do not become deciding factors. Adaptability is required so that the system can be evaluated on multiple users.

A number of researchers have studied the interaction based on hand gestures as an alternative form of interface for human computer interaction (Costanzo et al. 2003; Freeman et al. 1998; Lee and Xu. 1996; Pavlovic et al. 1997; Quek 1994; Starner and Pentland 1995; Yang and Ahuja 2001). They all vary in input device (vision, sensor glove, or color coded glove), types of gestures (static, or dynamic), vocabularies (pre-defined sign language, or task dependent gestures), speed (from 0.5 fps to a real-time), and segmentation (capable of spotting, or requires short pause in data).

We chose high-speed glove based interface capable of spotting dynamic gestures from task dependent gesture vocabularies. Gesture recognition module is implemented using a Hidden Markov Model, a stochastic method in which on-line model adaptation and reinforcement are very common. Each features and the implementation is discussed in detail in the following sections.

4.2.1. Gesture Recognition Module: Implementation

The function of the multi-modal recognition module (the first block in Figure 3, page 23) is to translate hand gestures and spontaneous speech into a structured symbolic data stream without abstracting away the user's intent. The symbols could be gestures, words, or both. Abstraction of intent can be avoided by ensuring that the robot can cover the entire configuration space by using the multi-modal interface, since intention is defined as a set of goal-directed robot actions. The flexibility given to the user through real-time interaction and the framework's intuitive interface allows the captured intent to be closer to the user's true intent. We consider two sub-functions. First, the module needs to translate incoming audio and gesture signals into a structured stream of word and gesture unit symbols with appropriate parameters. Second, the module needs to be able to adapt to new users by reinforcing recognition models using new incoming data during recognition.

The recognition module generates a parameterized output stream. Examples of such parameters are the direction and velocity of the hand for a *waiving* gesture, or the designated x-y coordinates on the floor for a *pointing* gesture. The types of input modalities discussed throughout the paper are human voice and hand gestures parameterized by two

22-sensor CyberGlove. Other modalities can replace or be added to the current recognition module.

The second function of the module is to adapt to the data from new users by reinforcing symbols during recognition. Online adaptation of the recognition model to the data from new users contributes to a better recognition rate than that achievable without adaptation. The multi-modal recognition module is implemented using a Hidden Markov Model, a stochastic method in which on-line model adaptation and reinforcement are very common.

Gesture recognition module is implemented using the Hidden Markov Toolkit (*HTK*) (Young et al. 2000) that has been customized to recognize gestures at 60Hz. Using HTK, which was primarily developed for speech recognition research, we were able to treat hand gestures as words, and a sequence of hand gestures as a sentence. HTK offers versatile tools and capability to build HMMs for recognition purposes. Features that are useful for constructing a fast and adaptable gesture spotter include the following:

1) a capability to specify a grammatical network.
2) a tool to train model parameters from unlabeled continuous training data where only the sequence of the gestures are known (no boundary information)

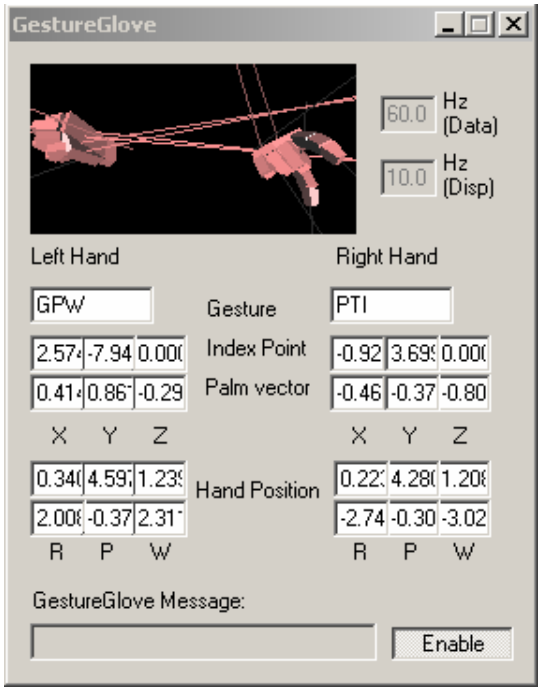


Figure 7: Implementation of the Gesture Recognition Module

3) a tool to adapt HMM to new users from small amount of new training data.
4) a tool to recognize HMMs in real time, with additional capability to adapt HMMs using newly generated transcripts.

The first feature is useful for constructing a spotting recognizer (Discussed in Section 4.2.4) so that the system recognizes spontaneous gestures without having to place a short pause between meaningful gestures. The second feature is helpful because unlike speech, there is no corpus (a large set of bootstrap data) for gestures, and it would be impossible to train dynamic gestures if one needs to time-stamp training sequences by hand. The second feature allows designer of gesture recognition system to simply record gestures in known sequence and have the system go through embedded training using multiple HMMs and training data. The third feature is useful when HMMs trained on particular user needs to be adapted to new users, which is very common in speech and gesture application scenario. The fourth feature allows the recognizer to come up with the most likely HMM from given observation data. It also allows the model to adapt to new users on-the-fly. Figure 7 shows the current implementation of the gesture recognition module.

4.2.2. Vocabulary

A hand gesture is a rich and diverse mode of communication, and the vocabulary used in hand gestures vary depending on the context and cultural background (Bremmer and Roodenburg 1992). In the context of spatial interaction, Quek (Quek 1994) developed a taxonomy of gestures which describe spatial relationships, structure and motion (Figure 8). They are divided into four sets of gestures: locative (*point*), orientational (*rotate, roll, pitch*), relative spatial (*large, small, left, right, up, down, farther, nearer*), and spatial pantomime (*track*). This vocabulary is to be applied in applications that require three-dimensional spatial input. For my purpose of interacting with two dimensional mobile robot, I have selected the following gestures: *point, stop, left, right, farther*, *nearer*, and *rotate*. These gestures are generalized into the following gestures: *point, grasp, waive,* and *turn*. Pointing gestures are particularly useful in conveying item of interest (Conway and Cohen 1998) and many researchers are interested in using it for interaction purpose (Hexmoor and Yang 2000).

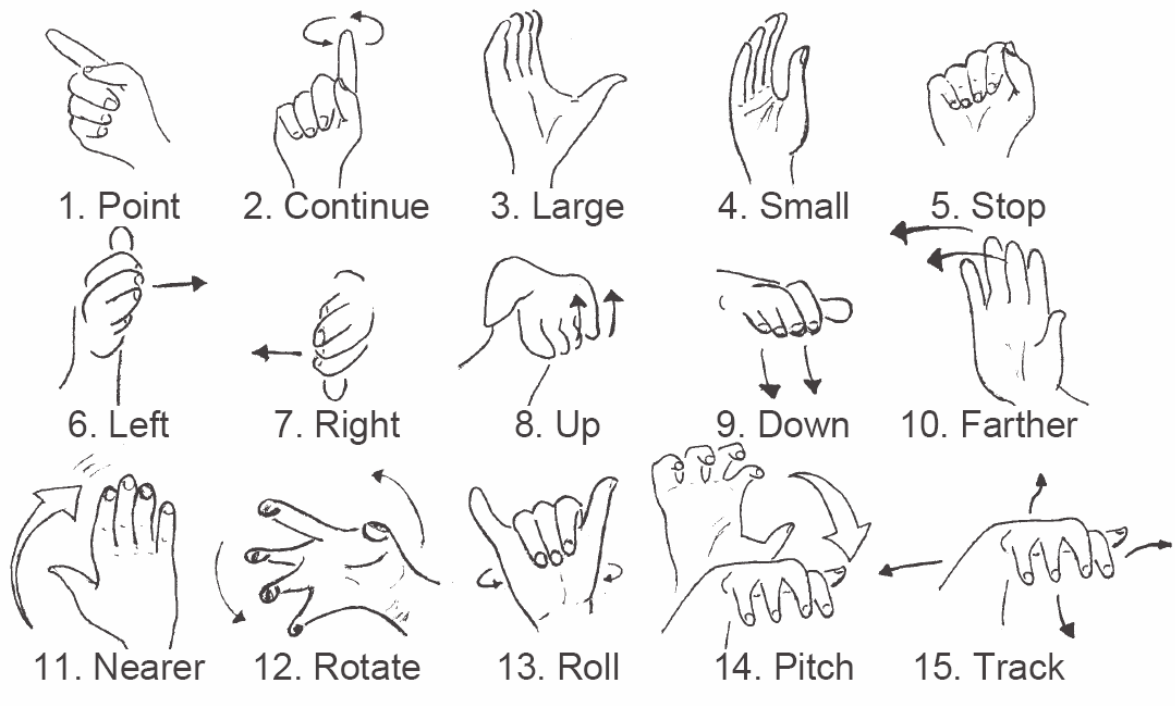


Figure 8: Gesture Vocabulary (Quek 1994)

Quek's Gesture	*Selected Gesture*	*Gesture Symbols*	*Phonemes*
Point	Point (index)	PTI	pti
	Point (index+thumb)	PTL	ptl
	Point (index+middle+thumb)	PTX	ptx
Stop	Grasp (Power)	GPW	gpw
	Grasp (Precision)	GPC	gpc
Left	Waive (forward)	WVF	wvf1+wvf2+sp
Right	Waive (backward)	WVB	wvb1+wvb2+sp
Farther			
Nearer			
Rotate	Turn (clockwise)	TNO	tno1+tno2
	Turn (c-clockwise)	TNI	tni1+tni2
-	Open	OPN	opn+sp
-	Garbage	GBG	gbg

Table 2: Gesture Vocabulary

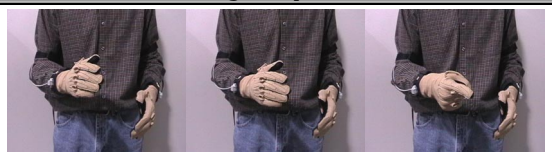
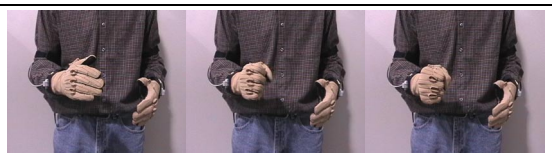
Gesture Symbols	*Image Sequence*
PTI	
PTL	
PTX	
GPW	
GPC	
WVF	
WVB	
TNI	
TNO	
OPN	

Table 3: Gesture Vocabulary in Images Sequence

The set of vocabulary used in the multi-modal robot programming interface is described in Table 2 and Table 3. For additional diversity, I have added different kinds of pointing (index finger with thumb up, and index and middle finger with thumb up), grasping (power grasp, and precision grasp), waiving (waiving in palm direction, and waiving in opposite direction), and turn (turn clockwise, and turn anti-clockwise). Gesture symbols refer to the actual symbols that the gesture recognition module generates, and the phonemes refer to the building blocks of the gestures.

4.2.3. Training and Adaptation

Prior to recognizing gestures, three training subjects each spent two hours to record a total of ~4000 executions of gesture sentences (a total of ~19200 gestures) to train the basic gesture models. The large number of training sets was necessary due to variability in gesture execution and the stochastic nature of HMMs. The training data comes with no boundary information, but comes with the sequence of labels. The trainer applies embedded Baum-Welch algorithm (Young et al. 2000) which produces the best HMM parameter estimate out of all training set using provided label sequences. In addition, tri-phone model is constructed to capture both inter-word transitions as well as intra-word transitions.

To adapt HMMs to new users, the Maximum Likelihood Linear Regression technique in the Hidden Markov Model toolkit is used to estimate a set of linear transformations for the mean and variance parameters of a Gaussian mixture HMM system that reduces the mismatch between the current model set and the adaptation data. The same technique is used in both supervised and unsupervised mode. The supervised mode uses adaptation data of the new user from known label sequence. The unsupervised mode uses an estimated label sequence from recognition results to adapt model parameters.

For the gesture recognition module, new user is asked to perform few gesture sentences to provide supervised adaptation data. Then HMMs adapts to the new user accordingly by generating model parameter transforms that further reduce modeling errors on given adaptation data.

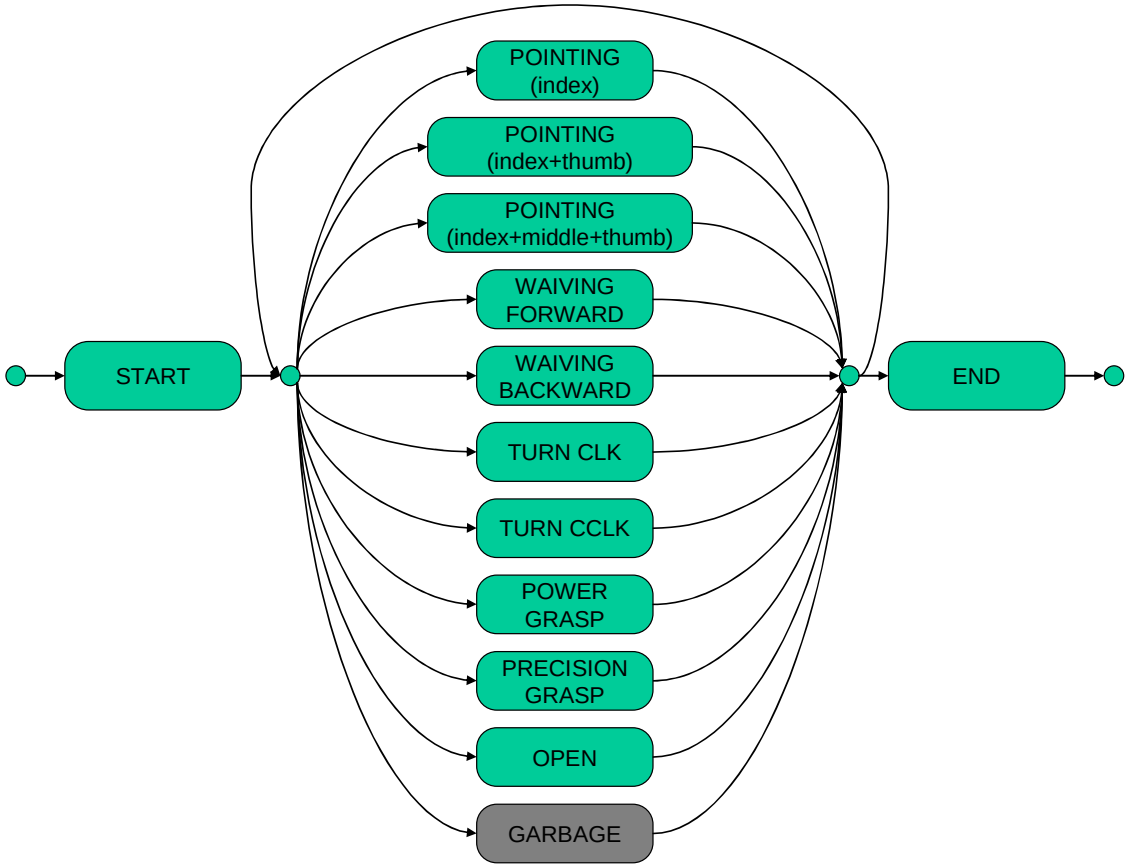


Figure 9: Gesture Spotting Network

4.2.4. Spotting

We purposely use the term ‘spotting' in addition to 'recognition' to emphasize the importance of spotting a gesture from a sequence of data containing both gestures and non-gestures. It is essential for a gesture-based robot control system to avoid sending inadvertent control signals to the robot. In order to spot a gesture, a garbage model is prepared in advance. A garbage model is trained just like any other words, but the model represents non-words, so that training data is basically a sequence of random hand motion.

Given a garbage model, recognition takes place in the gesture grammar network described in Figure 9. It describes word transition and it is obvious that the recognizer will either spot the gesture, or fall into a garbage.

The gesture recognition module takes 22-sensor CyberGlove data, and uses their delta to produce 44-dimensional vector, which runs through a Token Passing Algorithm (a

different implementation of Viterbi algorithm) to recognize the most likely gesture at the time. The average recognition accuracy on new user after adaptation is ~92%.

4.2.5. Parameter Extraction

Not only that the hand gesture needs to be recognized, it is as important to extract parameters such as vectors coming out of finger tips, direction of the palm, etc. For gesture-based robot control, accuracy of the parameter decides usability of the system. Polhemus 6DOF position sensor provides positions of wrists for both hands. The system finds necessary information by using coordinate transforms with CyberGlove joint angles.

There are two sources of error in extracted parameter: bad positioning of a Polhemus source transmitter, and failure in CyberGlove calibration. The origin in global reference frame is where the robot, Cye, charges its battery. It is necessary to make the best effort to come up with the right distance from the home position to the Polhemus source transmitter in order to get a good positioning accuracy of hands. The same goes to the glove calibration. The current strategy is to have the user lay his hand flat on the table, so that the difference in offset is removed from the model. Although extracted parameters are plagued by dual accumulating errors, the user will be able to control the robot well up to the certain distance since the user is in the loop to compensate for the errors.

4.3. Speech Vocabulary and Recognition

In our implementation, spontaneous speech is translated into words using SPHINX-II, an off-the-shelf speech recognition package (Huang et al. 1993). The current system works only with a basic set of words and gestures and does not include interactive learning

Vocabulary Type	*Candidate Symbols*
Motion	Go, Move, Goto, Stop, Turn, Forward, Backward, Right, Left, Sweep, Vacuum, On, Off,
Deictics	This, That, There
Attributes	Yellow, Green, Black
Numbers	One, Two, Three, etc.
Names	Robot, Cye
Programming Commands	Program, Execute, Complete

Table 4: Speech Vocabulary

of new gestures. Table 4 lists some of the initial words that such a basic vocabulary could include. For example, the user is able to point at a certain position on the floor using a hand gesture coupled with the “Go There” command to the “Yellow Robot” via voice.

4.4. Combining Gesture and Speech Recognition

Results from gesture and speech recognition are streamed into the next module “intention interpretation” which combines the results, extracted parameters, and the robot state to decide that robot task to execute. In general, gesture input need to come together with deictic terms from speech recognition.

4.5. Summary

This chapter described the implementation of the multi-modal recognition module. Gesture vocabulary was selected from the set of 3D spatial interaction gestures and the set is reduced to match the need of 2D interaction required for mobile vacuum robot control. Gesture recognition module is implemented using the Hidden Markov Toolkit, which provides number of necessary features to perform training, adaptation, and spotting recognition. Training and adaptation strategy ensures that the new user can get their gestures recognized. Grammatical network with garbage model is used for gesture spotting. Parameter extraction is a serious issue when working with loosely calibrated sensors, but reasonable effort will lead to a good gesture-based robot control system. Speech recognition uses off-the-shelf recognition engine (*Sphinx-II*), and their vocabulary is selected based on task.

Chapter 5.

Interactive Robot Control

In this chapter, I describe how the interactive multi-modal robot programming framework achieves tight interaction.

5.1. Introduction

Providing interactive control to a mobile robot is a key to creating a novice-friendly human-robot interaction system. The intention interpretation module of the framework interprets a stream of results from the multi-modal recognition module into a stream of task requests for the robot or the system. Then task requests are separated into smaller unit of actions called primitives and fed into the preemptive arbitration module, which sends control commands to the robot based on the control policy of the primitive. The preemptive execution is crucial to providing the user a real-time interaction to control and program a robot on-the-fly. In this chapter, I describe how the two modules operate to provide interactive control and on-line programming to the overall framework.

5.2. Intention Interpretation Module

The intention interpretation module (The second block in Figure 3) has three functions. The first is to recognize and select the appropriate task based on the current context. The second is to attach priorities to the task to handle multiple task requests. The third is to adapt the task representation used for task recognition and selection to the most current observation.

The problem of intention interpretation can be considered as a mapping problem from the stream of user inputs, the current state of the system, and the robot sensor data, to the correct robot task. The user input is an incoming stream of structured symbolic data (with parameters) from the multi-modal recognition module. The robot sensor data is an abstracted version of the robot's sensor stream. For a mobile robot, the robot sensor data

could include range sensor data, distance to the closest obstacle, the robot's global position and current velocity. For manipulators, the robot sensor data includes the end-effector's position and velocity in the joint space or Cartesian space, and contact data if force, torque, or tactile sensors are available.

The output of the intention interpretation module is a task symbol representing a configuration of robot primitives. The usage and definition of the terms *primitive* and *task* are discussed in the next section. In short, a primitive is an encapsulation of a low level robot behavior; that is, a policy $\pi(\boldsymbol{x},t,\alpha)=\boldsymbol{u}$ that maps the state $\boldsymbol{x}$ of a system and its environment into an appropriate action $\boldsymbol{u}$ at a particular time t, with additional parameters α. The task is a robot program composed of various primitives. The semantics database (Table 5) is implemented as a look-up table of candidate task symbols and their priorities from

Input Symbol ("voice" and 'gesture')	*Candidate Task*	*Priority*
"Stop" or two 'Closed' fists	Stop()	High
"Go" + "This", "That", *etc* + 'Point'	Goto(*P*)	Medium
"Go" or "Move" + Direction ("Right", "Forward", "Left", "Back")	Move(*v*)	Medium
'Waive' or "Go" + 'Waive'	Move(*v*)	Medium
"Go Home"	GoHome()	Medium
"Vacuum" + "On" or "Off"	Vacuum(On/Off)	Medium
"Turn" or "Turn" + direction ("Right", "Left")	Turn(ω)	Medium
"Cover Area" + two 'Point's	AreaCoverage(P_1,P_2)	Medium
"Program" *p*	Program a task *p*	Low
"Complete" *p*	End of program	Low
"Execute Program" *p*	Execute a task *p*	Low

Table 5: Semantics Database

Primitive	*Parameter*	*Action*
Goto	*Position P*	Move to the position, w/path-planning
Vacuum	*On/Off*	Toggle the state of the vacuum cleaner
AreaCoverage	Rectangular Area (P_0, P_1)	Traverse the area specified
GoHome	*N/A*	Move the robot back to the home position
Move	*Velocity(v)*	Apply additional velocity *v* to the robot
Turn	*Angular Velocity(ω)*	Apply additional angular velocity ω to the robot
Stop	*N/A*	Stops the robot motion

Table 6: Primitives Database

input symbols from the multi-modal recognition module. The database is implemented as an XML lookup table (Figure 10). Initially, the semantics database contains tasks that are composed of single primitives. The primitives (Table 6) in our vacuum-cleaning robot scenario are single-purpose controllers. Priorities in the semantics database are assigned in a way such that critical and smaller tasks receive higher priorities. For example, critical tasks such as Stop() receive highest priority, single primitive tasks such as Goto(P), Move(v), etc. receive medium priorities, and finally tasks composed of multiple primitives and those associated with program composition receive low priorities. Tasks with identical priorities are arbitrated on a first-come first-serve basis (Section 5.3).

Instead of merely mapping the sequence of multi-modal recognition results to the set of actions using the semantics database (Table 5), the intention aware system should suggest which task (set of primitives: Table 6) the user may want to execute based on an incomplete sequence of primitives executed by the user. This recognition ability is similar to the auto-completion ability in a text editing program (Stallman 1984). It is especially

```
- <GRAMMAR LANGID="409">
  + <DEFINE>
    <!-- top level rules -->
  + <RULE NAME="PID_CmdRobotMotion" ID="PID_CmdRobotMotion" TOPLEVEL="ACTIVE">
  + <RULE NAME="PID_CmdConfirmation" ID="PID_CmdConfirmation" TOPLEVEL="ACTIVE">
  + <RULE NAME="PID_CmdProgManip" ID="PID_CmdProgManip" TOPLEVEL="ACTIVE">
  + <RULE NAME="PID_CmdReference" ID="PID_CmdReference" TOPLEVEL="ACTIVE">
    <!-- Turn commands -->
  + <RULE NAME="PID_CmdTurn" ID="PID_CmdTurn">
    <!-- vacuum commands -->
  + <RULE NAME="PID_CmdVacuum" ID="PID_CmdVacuum">
  + <RULE NAME="PID_Vacuum" ID="PID_Vacuum">
    <!-- stop commands -->
  + <RULE NAME="PID_CmdStop" ID="PID_CmdStop">
    <!-- GO commands -->
  + <RULE NAME="PID_CmdGo" ID="PID_CmdGo">
  - <RULE NAME="PID_CmdGoPoint" ID="PID_CmdGoPoint">
    - <L PROPNAME="PID_VacuumStatus" PROPID="PID_VacuumStatus">
      - <P VAL="0">
          <RULEREF NAME="PID_Goes" />
        </P>
      - <P VAL="1">
          <RULEREF NAME="PID_Vacuum" />
        </P>
      </L>
      <RULEREF NAME="PID_Deictics" />
    </RULE>
  + <RULE NAME="PID_CmdGoArea" ID="PID_CmdGoArea">
  + <RULE NAME="PID_CmdGoDirection" ID="PID_CmdGoDirection">
  + <RULE NAME="PID_Goes">
```

Figure 10. Part of the XML implementation of the Semantic Database.

helpful when there are a large number of programs, and explicitly searching for any particular program may be time-consuming.

In order to perform such recognition in the real world, it is necessary to represent tasks in a probabilistic framework rather than as a discrete sequence of commands such as {*Goto*(P_1), *Vacuum(vacOn), AreaCoverage*(P_2,P_3), *Vacuum(vacOff), GoHome()*}, where the P_i's describe robot positions in terms of (x, y). A Hidden Markov Model (HMM) (Rabiner 1989) provides a way to model the task in a probabilistic framework, where both state transitions and observations can be expressed stochastically. The conversion of a task to the HMMs are described in detail later in Chapter 6. Since no branching or looping is allowed in tasks, each task can be described as a left-right (Bakis) HMM using an observation sequence collected at the time of programming. Tasks represented in HMMs are organized and compared to the current observation sequence to detect which task, if any, the user may want to execute. Other work, such as the human intention recognition by Yamada et al. (Yamada et al. 2002) and the online point-based hand writing recognition by Bahlmann (Bahlmann and Burkhardt 2001), employs similar strategies. However, our method has an advantage since it is capable of disregarding non-task (garbage) sequences through a garbage collector, without prior training of a garbage model. In this work, the garbage sequence refers to a sequence of observations that are not previously modeled. Such a sequence could therefore occur when the user guides the robot to a new position, and thus needs to be disregarded in the system's task suggestion.

A second important function of the intention interpretation module is to prioritize tasks. Not all tasks are of equal importance. For example, the gesture or word that corresponds to an emergency stop has a very high priority, and should be executed even if the robot is already engaged in another task. Similarly, a high-level task, like navigating to a point (x,y), may require assistance from the user to avoid obstacles and dead ends. The task, therefore, has a lower priority than the tasks for interactive user assistance.

A third function of the intention interpretation module is to perform online modification and adjustment, which are essential since it is unreasonable to expect the system to have prior knowledge of every intended task. The system must be capable of adjusting and adding primitives to the program with ease. The system supports these

adjustments by letting the user interrupt the task while it is running, and by registering the interrupts as additional primitives in the task.

5.3. Prioritized Task Execution Module

The prioritized task execution module (the third block in Figure 3) has two functions. The first is to arbitrate and execute primitives based on the current state, the sensor inputs, and the prioritized task given by the previous module. The second is to generate a robot program (task) by configuring primitives.

Before going into the details of each function, we distinguish tasks from primitives based on their level of abstraction. Primitives are encapsulations of low-level robot behaviors and serve as building blocks of high-level behaviors. They consist of motor (M), sensor (S), or sensori-motor (SM) primitives. Motor primitives generate open loop behaviors that do not depend on sensor feedback. For mobile robots, motor primitives include sensor independent acceleration, stop, turn, beep, and directional motions. Sensor primitives provide the system with observable sensor signals, such as the current robot position, range sensor data, and bumper switch data. Sensori-motor primitives generate closed loop behaviors, such as wall following or navigation towards a particular destination. The sensori-motor primitives can be thought of as pre-tailored configurations of motor and sensor primitives.

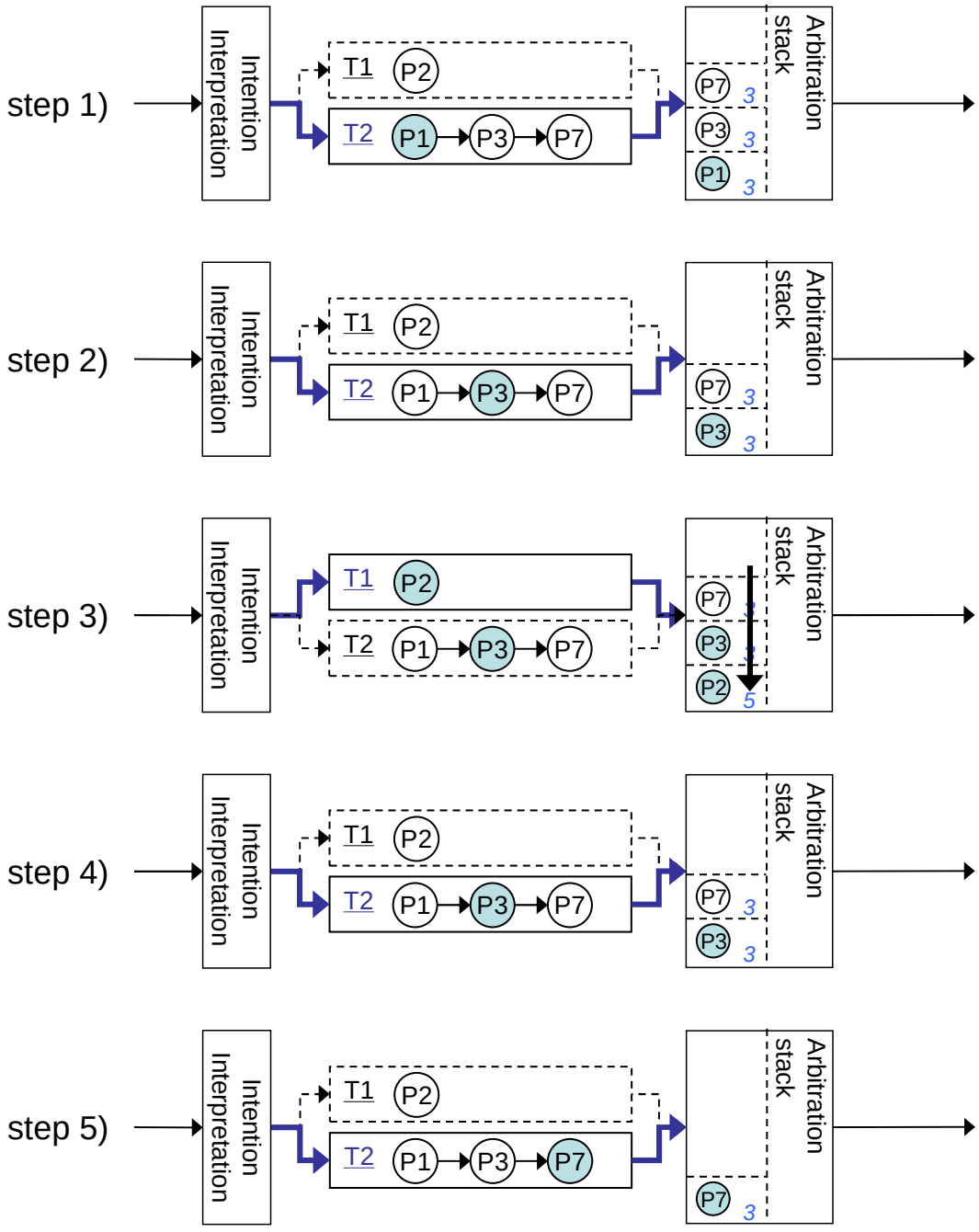


Figure 11: Arbitration Based on Task Priority with T1 (priority=5), T2 (priority=3)

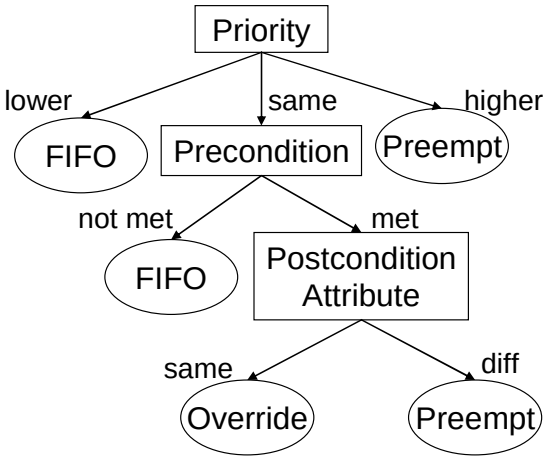


Figure 12. Arbitration Policy Tree

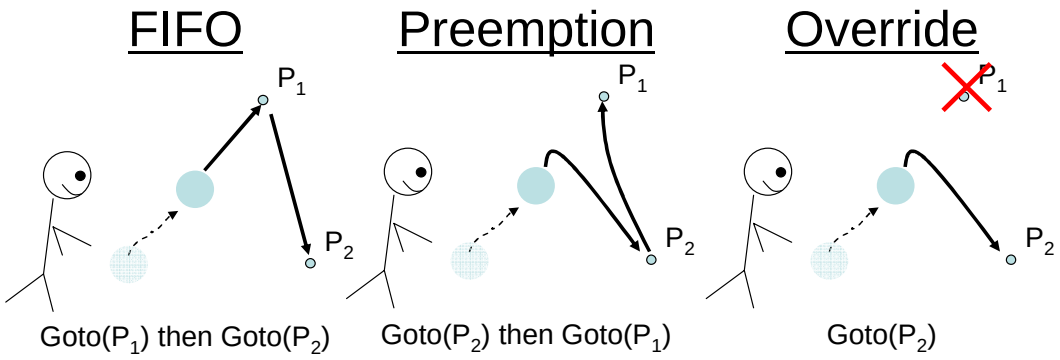


Figure 13. Possible actions for the sequence of instructions: Goto(P_1) followed by Goto(P_2).

A task is a configuration of primitives—either a sequence of primitives, or a single primitive. Tasks are stored in a database in the form of a state buffer (Kimura et al. 1999), Markov chain (Rybski and Voyles 1999) or finite state machine (Morrow and Khosla 1997). Because the intention interpretation module requires access to some of the task data also, it shares the semantic, primitive, and task databases with the task execution module, as is shown in Figure 3.

The first important function of the task execution module is task arbitration. As is explained in the section on intention interpretation, not all tasks are equally important. The scheme can be described as event driven preemption, where the event (a request from the intention interpretation module to execute a task) triggers an active switch from the running task with lower priority to another with higher priority. Preemptive execution is crucial in providing the user real-time interaction to control and program a robot on-the-fly. Tasks are

prioritized according to a pre-defined rule (Table 5), and sequential robot actions in the tasks are executed and sometimes overridden based on the arbitration policy. Figure 11 describes the preemptive execution when task priority is used as a decisive factor. The task T2 with low priority (3) is preempted by the task T1 with medium priority (5) during the execution of primitive P3. The task T2 can be thought of as a program (task) that consists of primitives P1, P3, and P7, while the task T1 is a single primitive task. Priority based arbitration can be used for a simple set of tasks, but it does not provide elaborate arbitration based on a given environment and its current situation. Therefore, arbitration policy tree (Figure 12) is used in the system which allows the user to handle situations such as making an emergency stop or avoiding an obstacle during the execution of other tasks. An illustrative example of task arbitration given both priority and arbitration tree is given in Figure 13. Imagine that the user first points to a particular position, P1, and asks the robot to "go there", then points to a different position, P2, and asks the robot to "go there" while the robot is moving toward P1. There are three potential outcomes to the above sequential instructions: FIFO, Preemption, and Override. The outcome is decided by the arbitration policy tree by using features such as priority, precondition, and post-condition attribute. For the given example with two consecutive "go there" instructions, the outcome is an Override, since they have no preconditions and their priorities (given by the Semantic Database: Table 5) and post-condition attributes (position) are the same. On the other hand, if the second instruction were "go this way" for a given direction, the outcome would be Preemption, since the post-condition attributes are different (position vs. direction).

The second function of the task execution module is to generate a robot program (task) interactively. The basic approach is to take a coaching strategy using a redundant input mode. The user sets the module to a learning mode and executes primitives sequentially; the system remembers the sequence as a task. There is an obvious problems with this approach. The problem is that the robot programs include conditional branching and looping. Forcing the user to remember a special gesture command to indicate branching and looping conditions would make the system counter intuitive. A tool to convey a program structure and an intuitive interface to edit the program are necessary, unless the system can infer such conditions from multiple examples. In the current implementation, iconic programming is used to write non-sequential program structures.

5.4. Summary

I described roles of two modules in the overall framework. The intention interpretation module interprets a stream of multi-modal recognition results into the task requests. The preemptive arbitration module receives the task request and separate them into a smaller set of instructions for the robot. The preemptive execution is implemented using priorities attached to the task request.

Chapter 6.

Intention Awareness

I present the method and the implementation of the framework on modeling and inferring user's intention from a temporal stream of robot actions. In the framework, user's intention is captured in a form of robot program. The flexibility given to the user through real-time interaction and the framework's intuitive interface allows the captured intent to be closer to the user's true intent. I describe modeling, updating, and recognition methods used to provide intention awareness to the system.

6.1. Introduction

Adaptation of models is necessary to reflect changes in the environment and modifications to a program made by the user while interacting with the system. Also, observation sequences collected from subsequent executions of the same task can be combined to improve stochastic parameters used in the HMM representation of the task.

In the remainder of this section, we explain how HMM representations of tasks are constructed and used for intention recognition, and how they can be updated in real-time during execution.

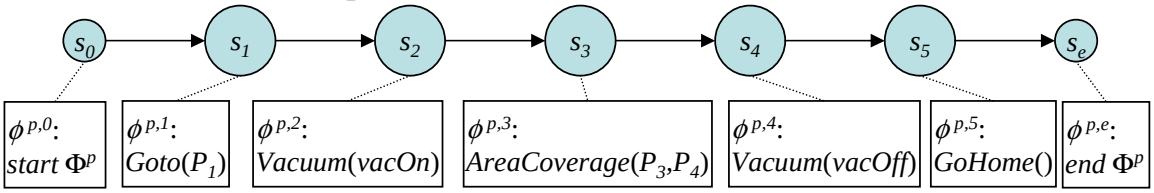
6.1.1. Modeling

The aim in this section is to create λ_p, a Continuous-Density HMM (CDHMM) description of a program p, from Φ^p, a command sequence of the program and to combine the λ_p's into a network CDHMM, λ_{net}, that can be used for real-time program recognition. This process is illustrated in Figure 14 and Figure 15.

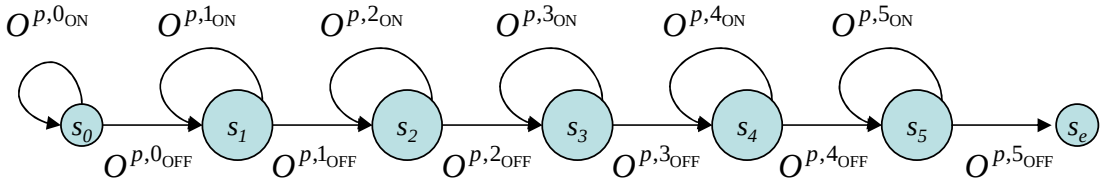
Robot Program Φ^p:

$\Phi^p = \{Goto(P_1), Vacuum(vacOn), AreaCoverage(P_3, P_4), Vacuum(vacOff), GoHome()\}$
$= \{ \phi^{p,1}, \phi^{p,2}, \phi^{p,3}, \phi^{p,4}, \phi^{p,5} \}$

1) Markov Chain Description of Φ^p:



2) Association of Observations Generated by Φ^p with CDHMM:



$O^p = \{O^{p,0}, O^{p,1}, \cdots, O^{p,N_p}\}$

$O^{p,a} = (O^{p,a_{ON}}, O^{p,a_{OFF}}) = (O^p_{\tau^{p,a_{ON}}}, O^p_{\tau^{p,a_{OFF}}})$

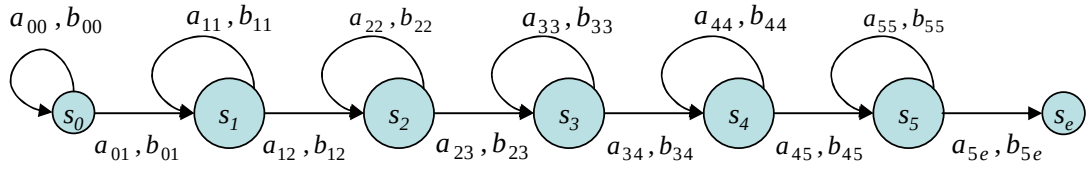
$O^p_t = (x_t, y_t, \theta_t)^p = (x, y, \theta)$ of program Φ^p at time index t

$\tau^{p,a_{ON}} = \{t \in \tau^p$: robot at time t executing an action $\phi^{p,a}\}$

$\tau^{p,a_{OFF}} = \{t \in \tau^p$: robot at time t after execution of an action $\phi^{p,a}$, before executing $\phi^{p,a+1}\}$

$\tau^p = \{$set of all time t during execution of program $\Phi^p\}$

3) Generated CDHMM description λ_p of Φ^p:



$\lambda_p = \{\pi_p, A_p, B_p\}$, where A_p is the set of a's, B_p the set of b's
π_p = initial state probability
a_{ij} = state transition probability from s_i to s_j
b_{ij} = observation probability during transition from s_i to s_j
$= N(\overline{\mu}_{ij}, \Sigma_{ij}) \cdot WN(\mu_{\theta ij}, S_{\theta ij})$

Figure 14: Conversion of a sample program Φ^p to Continuous Density HMM λ_p

HMM Network for Task Recognition:

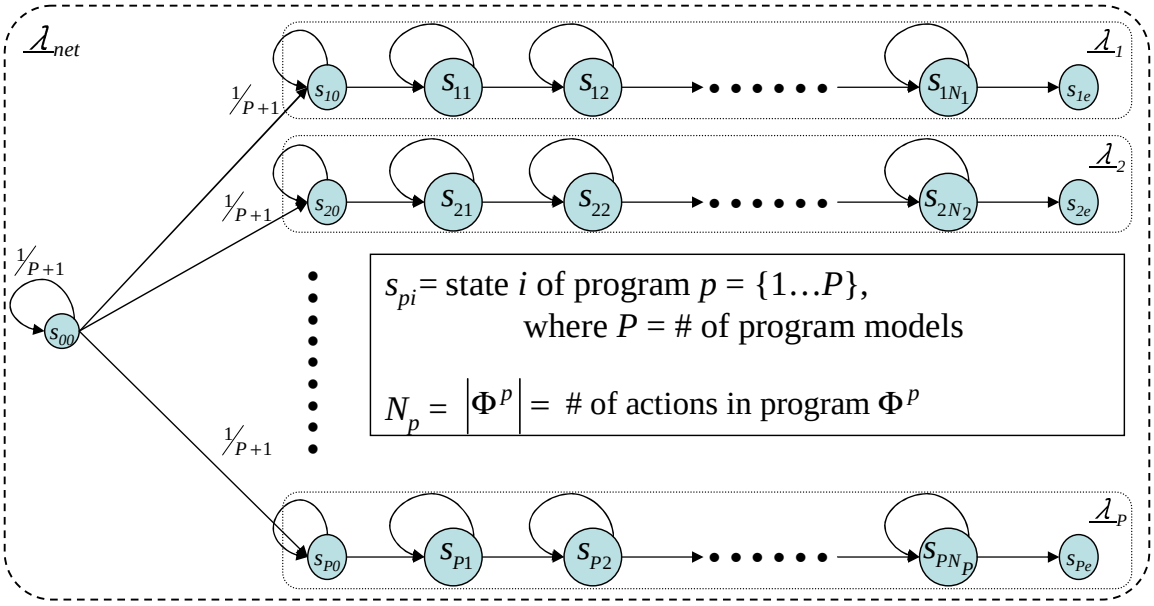


Figure 15: HMM Network with Shared Initial State

A program Φ^p consists of a set of sequential program actions $\phi^{p,0}$, $\phi^{p,1}$,…, $\phi^{p,n}$. When a robot is programmed interactively, the system collects an observation sequence $O^p = \{o_1^p, o_2^p \ldots o_{T_p}^p\}$ where $o_t^p = (x_t, y_t, \theta_t)^p$ correspond to the robot position and orientation at time t for program p. The sequence O^p of the program Φ^p is the collection of all observations $O^{p,a}$ resulting from program actions $\phi^{p,0}$, $\phi^{p,1}$,… $\phi^{p,a}$. The robot program Φ^p is then converted into a CDHMM λ_p through the process illustrated in Figure 14. The program is first converted into a Markov chain (top of Figure 14) whose states correspond to each programmed action. The number of states in the chain is the number of actions in the program plus two (the start and end states). Observations of the robot's position (x, y, θ) are collected during the task execution and are associated with each of the arcs in the state transition diagram. For each arc, the observation sequence is encoded in an observation density function, b_{ij}.

The observation density function, b_{ij}, is an expression for the likelihood of observing a given robot position $o_t^p = (x_t, y_t, \theta_t)^p$ given that the robot is moving from state i

to state *j*. b_{ij} (Equation 1) is modeled as a combination of a normal distribution, $N(\bar{\mu}_{ij}, \Sigma_{ij})$ (Equation 4) and a wrapped normal distribution $WN(\mu_{\theta ij}, S_{\theta ij})$ (Equation 9).

$$b_{ij} = N(\bar{\mu}_{ij}, \Sigma_{ij}) \cdot WN(\mu_{\theta ij}, S_{\theta ij}) \tag{1}$$

$$\bar{\mu}_{ij} = \frac{1}{T_{ij}} \sum_t^{T_{ij}} \bar{x}_t \tag{2}$$

$$\Sigma_{ij} = \frac{1}{T_{ij}} \sum_t^{T_{ij}} [\bar{x}_t - \bar{\mu}_{ij}][\bar{x}_t - \bar{\mu}_{ij}]^T \tag{3}$$

$$N(\bar{\mu}_{ij}, \Sigma_{ij}) = \frac{1}{2\pi\sqrt{|\Sigma_{ij}|}} \exp\left(-\frac{[\bar{x} - \bar{\mu}_{ij}]^T \Sigma_{ij}^{-1} [\bar{x} - \bar{\mu}_{ij}]}{2}\right) \tag{4}$$

$$\mu_{\theta ij} = \arctan\left(\sum_t^{T_{ij}} \sin\theta_t \Big/ \sum_t^{T_{ij}} \cos\theta_t \right) \tag{5}$$

$$S_{\theta ij} = 1 - \frac{\sqrt{\sum_t^{T_{ij}} \cos\theta_t + \sum_t^{T_{ij}} \sin\theta_t}}{T_{ij}} \tag{6}$$

$$\sigma_{\theta ij}^2 = -2\log(1 - S_{\theta ij}) \tag{7}$$

$$WN(\mu_{\theta ij}, S_{\theta ij})(\theta_t) = \frac{1}{\sqrt{2\pi\sigma_{\theta ij}^2}} \sum_{k=-\infty}^{\infty} \exp\left(-\frac{1}{2} \frac{(\theta_t + 2\pi k)^2}{\sigma_{\theta ij}^2} \right) \tag{8}$$

$$\cong \begin{cases} \dfrac{1}{\sqrt{2\pi\sigma_{\theta ij}^2}} \exp\left(-\dfrac{1}{2} \dfrac{\theta_t^2}{\sigma_{\theta ij}^2} \right) & , \text{for } \sigma_{\theta ij}^2 \le \pi \\ \left(1 + 2\sum_{p=1}^{3} e^{-\frac{1}{2}\sigma_{\theta ij}^2 p^2} \cos p\theta_t \right) \Big/ 2\pi & , \text{for } \sigma_{\theta ij}^2 > \pi \end{cases} \tag{9}$$

Observed positions are assumed to be correlated but orientations are assumed to be independent from the positions (*i.e.* $WN(\mu_{\theta ij}, S_{\theta ij})$ is independent from $N(\bar{\mu}_{ij}, \Sigma_{ij})$) since the robot motion tends to be either unidirectional or equally varying throughout the configuration space. We should point out that extra care is needed to calculate the sample mean and variance of orientation data, which are expressed in circular rather than Cartesian coordinates (Mardia 1972). A wrapped normal distribution $WN(\mu_{\theta ij}, S_{\theta ij})$ is approximated in one of two methods (Equation 9) depending on its angular variance, $\sigma_{\theta ij}^2$. Refer to Mardi's text (Mardia 1972) for justification of the approximation.

In addition to the observation probabilities, b_{ij}, the CDHMM for program λ_p is also characterized by state transition probabilities, a_{ij}. The state transition probability a_{ij} is determined by taking the ratio of the number of observations used at recurring and transition arcs. $O^{p,a} = (O^{p,a_{\mathrm{ON}}}, O^{p,a_{\mathrm{OFF}}}) = (O^{p}_{\tau^{p,a_{\mathrm{ON}}}}, O^{p}_{\tau^{p,a_{\mathrm{OFF}}}})$ is an observation sequence for program action $\phi^{p,a}$, where $\tau^{p,a_{\mathrm{ON}}}$ is a set of time index t while action $\phi^{p,a}$ is being executed, and $\tau^{p,a_{\mathrm{OFF}}}$ is a set of time index t after execution of $\phi^{p,a}$ and before execution of $\phi^{p,a+1}$. The state transition probability a_{ij} is defined as follows:

$$a_{ij} = \begin{cases} \dfrac{\left|\tau^{p,a_{\mathrm{ON}}}\right|}{\left|\tau^{p,a_{\mathrm{ON}}}\right| + \left|\tau^{p,a_{\mathrm{OFF}}}\right|} & \text{for } i = j \\ \dfrac{\left|\tau^{p,a_{\mathrm{OFF}}}\right|}{\left|\tau^{p,a_{\mathrm{ON}}}\right| + \left|\tau^{p,a_{\mathrm{OFF}}}\right|} & \text{for } i \neq j \end{cases} \tag{10}$$

After converting all programs Φ^p to λ_p for $p = 1 \ldots P$, where P is the number of program models, the λ_p are combined into one λ_{net} for recognition purposes. Figure 15 describes how λ_{net} is constructed from the CDHMMs λ_1, λ_2 … λ_P. All the transition probabilities from s_{00} ($a_{s_{00}s_{00}}, a_{s_{00}s_{10}}, \cdots a_{s_{00}s_{P0}}$) are assumed to be equal with a value of $1/(P+1)$. Observation probabilities for these arcs $b_{s_{00}s_{00}}, b_{s_{00}s_{10}}, \cdots b_{s_{00}s_{P0}}$ are undetermined at this point. They are assigned dynamically inside the recognition algorithm described in the following section.

The first demonstration was conducted to verify the connections between all three modules and to illustrate the overall operation of the framework with a basic interactive programming example. The framework is implemented using a Cye vacuum cleaning robot (Batavia and Nourbakhsh 2000), two 22-sensor CyberGloves, and a microphone. We modified the graphical user interface provided with the vacuum cleaning robot, to accept hand gestures and speech input, while retaining its original functionality: mapping, iconic programming, and path-planning. As a result, Cye can be controlled via mouse, speech, and hand gestures.

The multi-modal recognition module is implemented using the *Sphinx-II* speech recognition engine (Huang et al. 1993) and the Hidden Markov Toolkit (*HTK*) (Young et al.

2000) that has been customized to recognize gestures at 60Hz with 92% recognition accuracy. A discussion of the gesture recognition methodology is outside the scope of this article; however, the method is similar to the one in (Ogawara et al. 2000), where parameters of Hidden Markov models for each gesture are obtained from known strings of gesture examples. Each gesture consists of gesture phonemes that take into account finger-joint positions, joint velocities, and the hand's Cartesian position and velocity. The vocabulary of gestures is listed in Table 4. The on-line addition of vocabulary is not implemented at this point although the system is capable of adapting model parameters for new users with very few additional training samples, using capabilities offered by HTK.

The intention interpretation module is implemented with a semantic database (Table 5). The semantic database connects inputs such as gesture and speech symbols, the robot's sensor readings, and the current state to the most likely robot task. A task, which can be considered as a robot program, is a set of one or more primitives. Each task has predefined priorities attached to specify the importance of the task over the others in the event of preemption. At this point, the semantic database is fixed and does not support the on-line addition of entries.

The prioritized task execution module ensures that the primitives are executed according to their assigned priorities. The primitives used in the current scenarios are listed in Table 6 in page 42. Primitives such as *GoHome* and *AreaCoverage* provide high-level navigation, whereas primitives such as *Move*, *Turn*, and *Vacuum* give low-level control of the robot. Primitives are executed in order of arrival except when a high-priority task is introduced; such tasks pre-empt the current task and execute immediately.

6.1.2. Update

Online seamless adjustments of the statistics (a_{ij}, b_{ij}) that describe the robot program are essential for keeping the system healthy. For example, an additional obstacle on the path between via-points can change the trajectory of the mobile robot, requiring that the program description be adjusted. Parameter adaptation can be used to improve the CDHMM parameters over multiple executions of the same task. This can be done by first partitioning the observation sequence and merging statistics derived from new samples with the old statistics. n_{add} additional samples with mean vector, μ_{add}, and covariance matrix,

Σ_{add}, can be merged with n_{old} old samples with statistics μ_{old} and Σ_{old} to derive the combined statistics, n_{new}, μ_{new}, Σ_{new} as follows: (Appendix C):

$$n_{new} = n_{add} + n_{old} \tag{11}$$

$$\mu_{new} = (n_{add}\mu_{add} + n_{old}\mu_{old}) / n_{new} \tag{12}$$

$$\Sigma_{new} = (A + B + C + D) \ / \ (n_{new} - 1)$$
$$\begin{cases} A = (n_{add} - 1)\Sigma_{add} \\ B = n_{add}(\mu_{add} - \mu_{new})(\mu_{add} - \mu_{new})^T \\ C = (n_{old} - 1)\Sigma_{old} \\ D = n_{old}(\mu_{old} - \mu_{new})(\mu_{old} - \mu_{new})^T \end{cases} \tag{13}$$

Using the above equations, one can compute the statistics for adapted observation probabilities without having to keep the entire observation history. For implementation purposes, we always set $n_{old} = n_{new}$, so that the effects from old samples will eventually decay with additional adaptation cycles.

6.2. Intention Recognition

During recognition, the current sequence of position observations is evaluated and compared to all the robot program CDHMMs stored in the network λ_{net}. It is necessary not only to detect in real-time which program the user may be interested in, but also to reject observations that are not part of any existing program. The goal here is to find the most likely state q_t at the current time t, given observations up to time t, and the CDHMMs λ_1, λ_2 … λ_P, constructed from P robot programs organized into λ_{net} as described in Figure 15.

Initialization:

Assign a token with value of 1 to the initial shared state s_{00}.

Assign a token with value of 0 to all other states.

For all arcs not originating from state s_{00}, compute and store the value ψ_{ij}.

Algorithm:

for each time t **do**

for each state $i \neq s_{00}$ **do**

Compute and store the Mahalanobis distance between o_t and μ_{ij};

Pass a copy of the token in state i to each connecting state j, multiplying its value by $a_{ij}b_{ij}(o_t)$.

If the new token value underflows to 0, let the value be ε;

end;

Pass a copy of the token in state s_{00} to each connecting state j, multiplying its value by $a_{ij} \bullet \psi_{mn}$. Choose the ψ_{mn} for which the Mahalanobis distance between μ_{mn} and o_t is the smallest;

Discard the original tokens;

for each state i **do**

Find the token in state i with the largest value and discard the others;

end;

Normalize all tokens such that their sum equals 1;

Find the state q_t with the largest token value;

end;

Figure 16: Viterbi Algorithm with Dynamic Garbage Collection

To find the single most likely state q_t out of all states in the shared CDHMM network for the current observation sequence, we use a modified Viterbi Algorithm described in . The Viterbi algorithm, based on the Token Passing paradigm (Young et al. 1989), has been modified by adding dynamic garbage collection, that is, recognizing the state, s_{00}, in which none of the programs is being executed. The modification involves the dynamic computation of the observation probabilities for state s_{00}, as is illustrated in the sequence Figure 17, Figure 18, Figure 19, and Figure 20.

Consider a sample CDHMM network (Figure 17) constructed from a shared garbage state s_{00}, and two member CDHMMs each with only one state, s_{10} and s_{20}. The member CDHMMs have observation density functions $b_{s_{10}s_{10}}$ and $b_{s_{20}s_{20}}$ (Figure 18 and Figure 19). If the observation o_t is close to either $\mu_{s_{10}s_{10}}$ or $\mu_{s_{20}s_{20}}$, the algorithm will consider the observation to indicate that the corresponding state s_{10} or s_{20}, respectively, should be promoted. However, if the observation is far removed from both $\mu_{s_{10}s_{10}}$ and $\mu_{s_{20}s_{20}}$, then the shared garbage state, s_{00}, should be promoted. This is achieved by introducing a

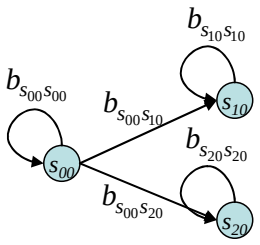


Figure 17: sample CDHMM network

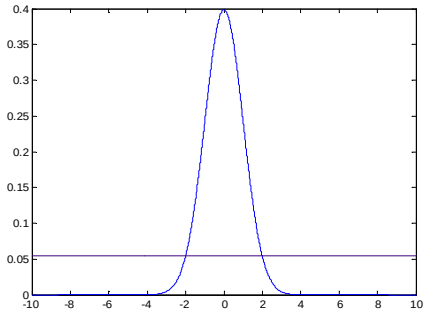


Figure 18: PDF for b_{s10s10}

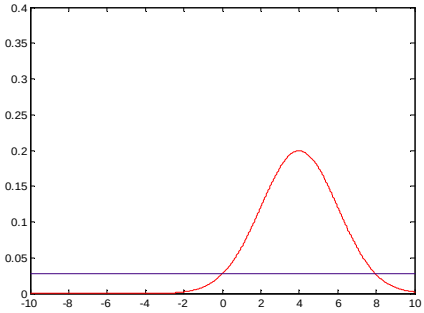


Figure 19: PDFfor b_{s20s20}

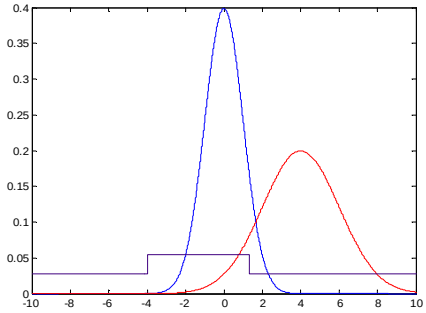


Figure 20: Dynamically generated b_{s00s00}, b_{s00s10}, b_{s00s20}

threshold ψ_{ij} for each b_{ij} below which the garbage state will be given preference. The value of ψ_{ij} is chosen as the value of b_{ij} at a distance of 3σ from the mean, that is:

$$\psi_{ij} = \frac{e^{-3^2/2}}{2\pi\sqrt{|\Sigma_{ij}|}} = \frac{e^{-4.5}}{2\pi\sqrt{|\Sigma_{ij}|}} \tag{14}$$

Since this value is different for every b_{ij}, the algorithm uses the value for the state that is closest to the current observation according to the Mahalanobis metric (Forsyth and Ponce 2003):

$$r_{ij}(t) = \sqrt{[o_t - \mu_{ij}]^T \Sigma_{ij}^{-1} [o_t - \mu_{ij}]} \tag{15}$$

This is illustrated in Figure 20. The lines for $b_{s_{00}s_{00}}$, $b_{s_{00}s_{10}}$, and $b_{s_{00}s_{20}}$ can be thought of as classification boundaries below which the algorithm gives preference to the garbage state.

The advantage of the above algorithm is that, unlike the garbage models used in large vocabulary state-spotting systems, this algorithm requires no previous training or

batch processing of the garbage model. It also requires little additional computation during the recognition phase because the ψ values only need to be computed once, during initialization, and the Mahalanobis distances, used to select the appropriate ψ values, are calculated and stored when the observation density functions are evaluated for a particular

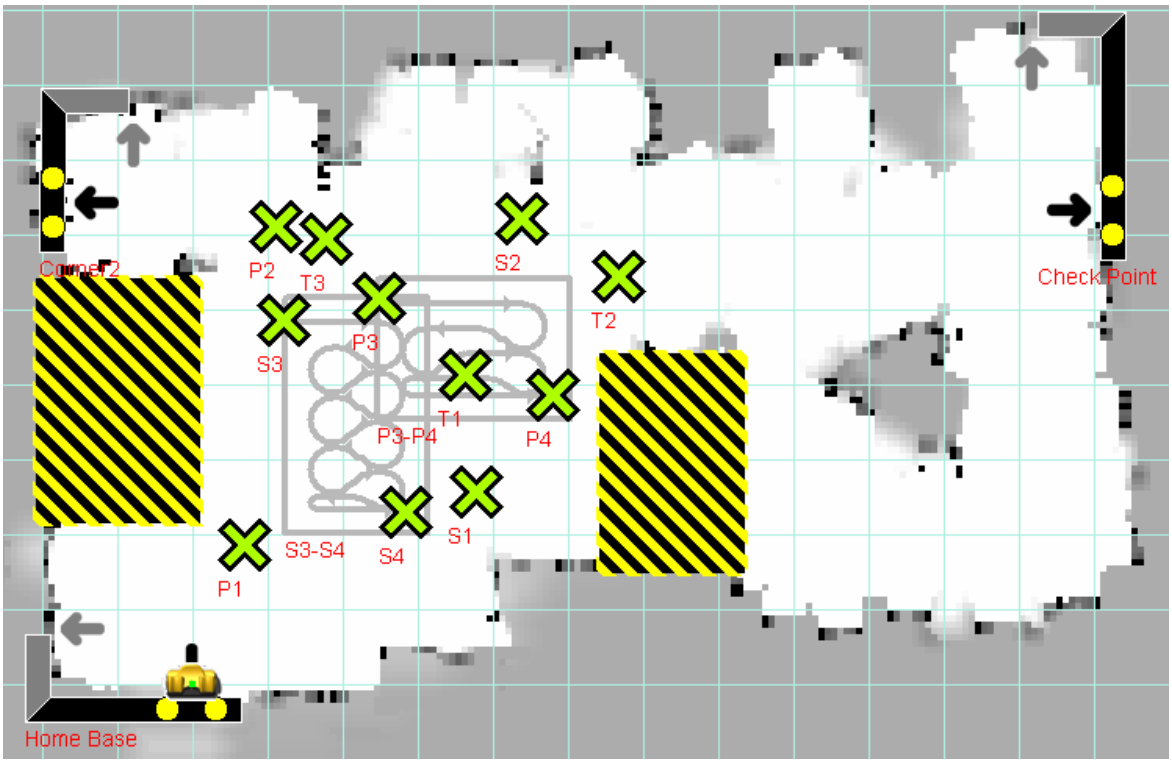


Figure 21: Positions used for the test programs λ_1, λ_2, λ_3

observation.

Since we are only interested in the most likely state, we only need to keep track of the CDHMM trellis of token scores as a starting point for the processing of additional observations. Each token value corresponds to the likelihood of being in the particular state after going through the most likely state sequence. Based on the assumption that the model λ_{net} fully explains all observation sequences, the entire trellis is normalized to 1.0 for every observation. The initial "garbage" state becomes the most likely state if the observation sequence can not be explained by any of the other models $\lambda_1 \ldots \lambda_P$. After finding the current CDHMM node, the system can determine the action that should be taken according to the most probable robot program.

The demonstration was conducted to verify the system's intention awareness. Assume that the database of robot programs contains three test programs:

Φ^1 = {Goto(P_1), Vacuum(vacOn), AreaCoverage(P_2, P_3), Vacuum(vacOff), GoHome()}

Φ^2 = {Vacuum(vacOn), Goto(S_1), Goto(S_2), AreaCoverage(S_3, S_4), GoHome()}

Φ^3 = {Goto(T_1), Goto(T_2), Goto(T_3)}

where P_i, S_i, T_i all represent positions on the map in (x, y), as described on Figure 21

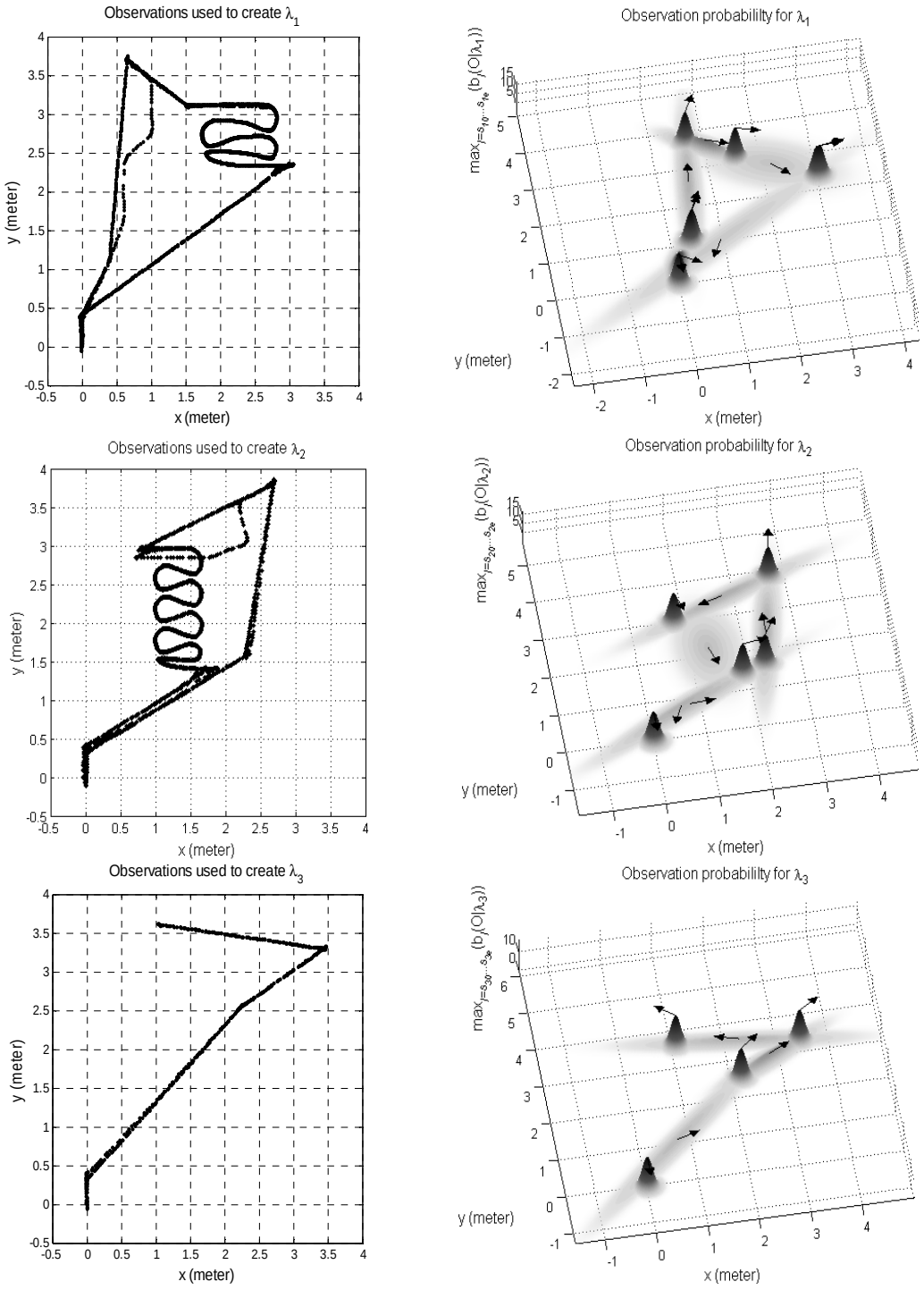


Figure 22: Observations used to construct the programs λ_1, λ_2, λ_3 (left) and the resulting observation probability densities (right)

For each test program, a CDHMM representation was created through the method described in section 6.1.1. The trajectories and the resulting observation probability densities are illustrated in Figure 22. The left column illustrates the data collected for the programs λ_1, λ_2, and λ_3, and the right column illustrates the statistical model constructed from the data. Each program was executed four times while collecting observation sequences with a 5Hz sample frequency. There was variability in the path in λ_1 and λ_2 to test adaptation.

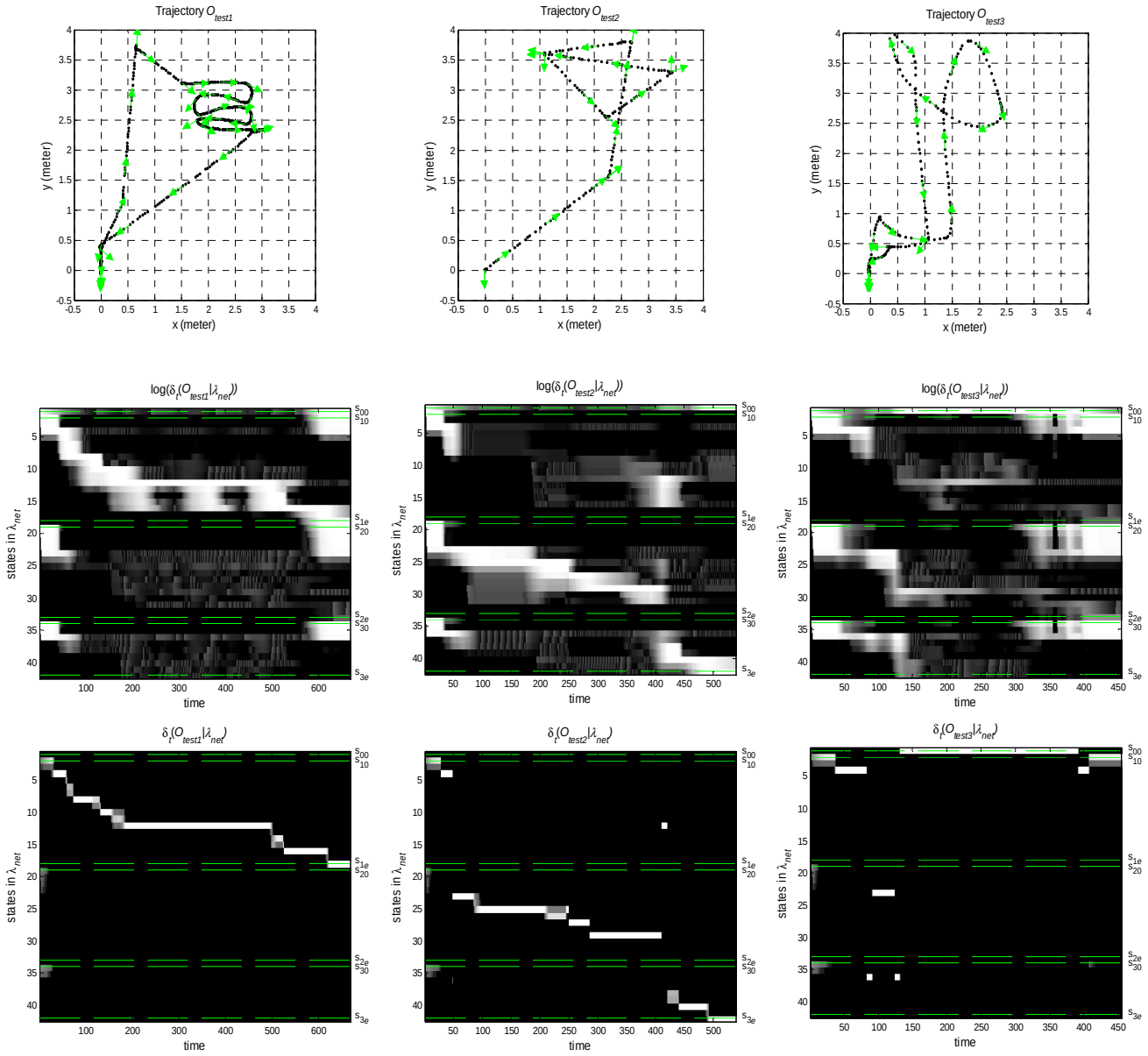


Figure 23: Observation sequences O_{test1}, O_{test2}, O_{test3}, with their corresponding δ_t (i)

Recognition was performed on the constructed CDHMM network λ_{net} using three test observation sequences, illustrated in Figure 23, shows three test sequence in different columns, and each row correspond to the observations sequence, $\log(\delta_t(i))$ to enhance small scores, and $\delta_t(i)$ to show which state i had the best score in each time step t.

The first test sequence is one of the training sequences used for the first program λ_1. Its $\delta_t(i)$ image (3^{rd} row, 1^{st} column) shows that the algorithm follows the states in λ_1 that are between s_{10} and s_{1e}. The second test sequence used Goto() commands to move in the order of {*Home,* S_1, S_2, T_3, T_1, T_2, T_3}. The test result shows what is expected: the recognition algorithm starts by selecting the states in the program λ_2 and jumps to the states in program λ_3. In the third test sequence, the robot follows a random trajectory that does not include any of the three programs in the database. Since this trajectory does not resemble any of the pre-defined programs, the recognition algorithm selects the shared garbage state almost the entire time. This is shown in the right-most graphs in Figure 23 where the top row of the figure is white, indicating that state zero has the highest probability value. The intention recognition procedure is summarized in Figure 24.

6.3. Summary

This chapter demonstrated that the system can determine the most likely high-level goal the user is trying to achieve, given a limited, initial sequence of task primitives. A set of user intentions, expressed as a robot program, was converted to HMM representations, and was used to recognize the most likely action that could be suggested to the user. Furthermore, we suggested a way to incorporate new observations to adapt the statistical model to previously unknown situations.

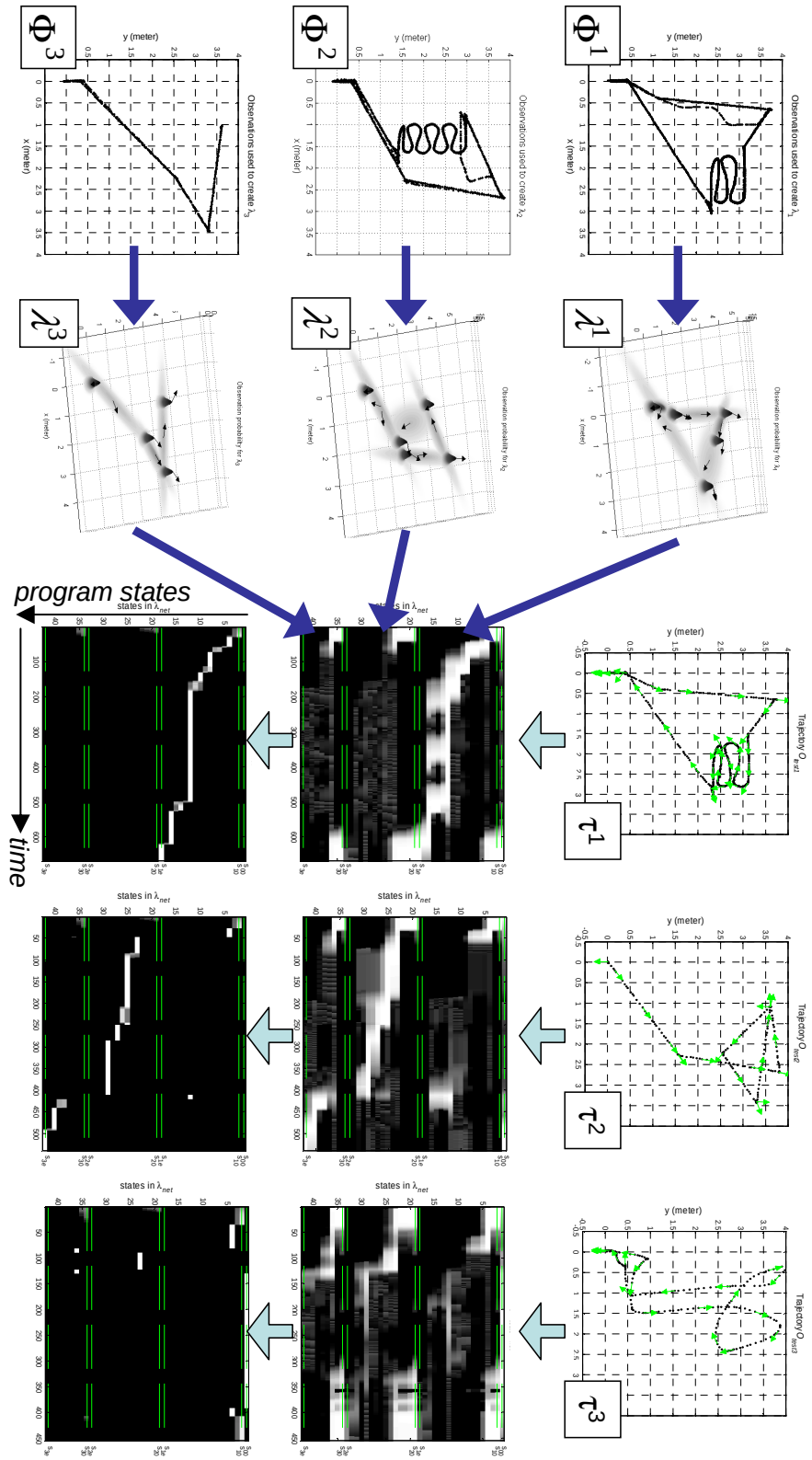


Figure 24: Summary of the intention recognition procedure

Chapter 7.

Interactive Robot Programming Demonstrations

7.1. Introduction

Tasks are prioritized according to the pre-defined rule, and sequential robot actions in the tasks are executed and sometimes overtaken based on their priorities. A task in the context of this framework is a high-level sequential robot program, composed by a sequential set of primitives that provide low-level control policy of a robot. Users are able to program a high-level sequential robot program while interactively controlling a robot. The benefit of the Programming by Interaction system is that the user can see what to expect from the program execution while programming a robot. Users can also modify the program by giving commands during program execution. If an intended task requires non-sequential structure, users may utilize the iconic programming interface on the system. The rest of the chapter describes the demonstration that was conducted to verify the system's task level interactive robot programming capability.

7.2. Demonstration

For the current implementation, we have considered two interactive programming scenarios. The first scenario is to have a user register numerous via-points to which the robot should navigate using its path planning capability. The second scenario is to use a two-handed gesture to specify an area that the robot should vacuum; the robot then vacuums the area using its area coverage primitive. In both scenarios, the robot can accept the user's preemptive speech and hand gesture commands to deal with unforeseen events. Figure 25 and Figure 26 illustrate the sequences of the first and second scenario. Each figure contains a sequence of camera snapshots with the corresponding conceptual illustrations of the framework, and the cropped images of the GUI. Refer to Multimedia

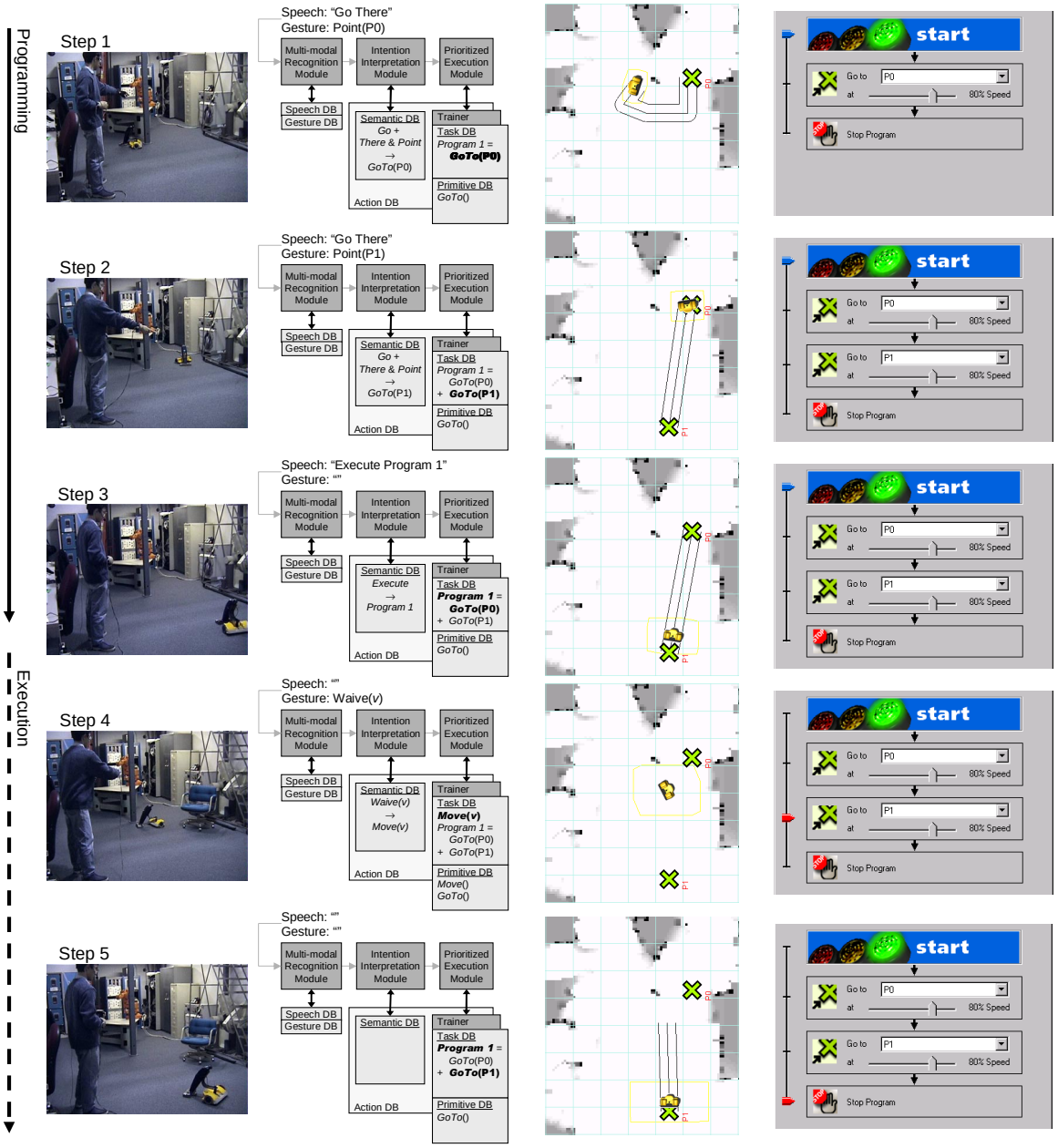


Figure 25: Demonstration Scenario 1

Extension 1 for the video of the programming phase and Extension 2 for the execution phase of the first scenario.

In the first scenario, illustrated in Figure 25, the user first verbally commands that the subsequent actions be stored as "Program One". The user then executes the *Goto* primitive by combining the voice command "Go There" with the gestural command 'Point'

to indicate the destination. In general, deictic terms such as “This”, “That”, and “There” must be accompanied by a referential gesture to specify the corresponding task parameters. For the *Goto* primitive, the Cartesian coordinates are extracted from the intersection between the extension of the index finger and the ground (Iba et al. 1999). In step 2, the user enters another *Goto* primitive but with a different end-position. After having saved these two primitives in “Program One” with the “Complete” command, the user can re-execute the program through with the voice command “Execute Program One”. However, in step 4, when the robot navigates to the second position from the first, it encounters an unknown obstacle. At this point, the user gestures the ‘Waive’ command, which has a higher task priority and can be used to control the robot around the obstacle. When the obstacle has been cleared and the user stops waiving, the robot returns to the execution of “Program One” (Step 5).

In the second scenario, illustrated in Figure 26, the user defines the task “Program Two.” After turning on the vacuum attachment with the voice command “Vacuum On” (step 1), the user issues the *AreaCoverage* command with one two-handed gesture; each hand performs a ‘Point’ gestures to specify the diagonally opposite corners of the area (with the direction aligned along the axes of the GUI). Steps 3 to 5 show the execution of the *AreaCoverage* command. As in the first scenario, at any point can the user re-execute “Program Two”, interrupt the execution, or interactively adjust the execution with higher-priority commands.

7.3. Summary

Set of two demonstration scenarios verified interactive multi-modal programming and execution of two sequential programming scenarios: point-to-point navigation and area coverage, which clearly illustrates the usefulness of multi-modal interaction, including the capability to interrupt commands preemptively.

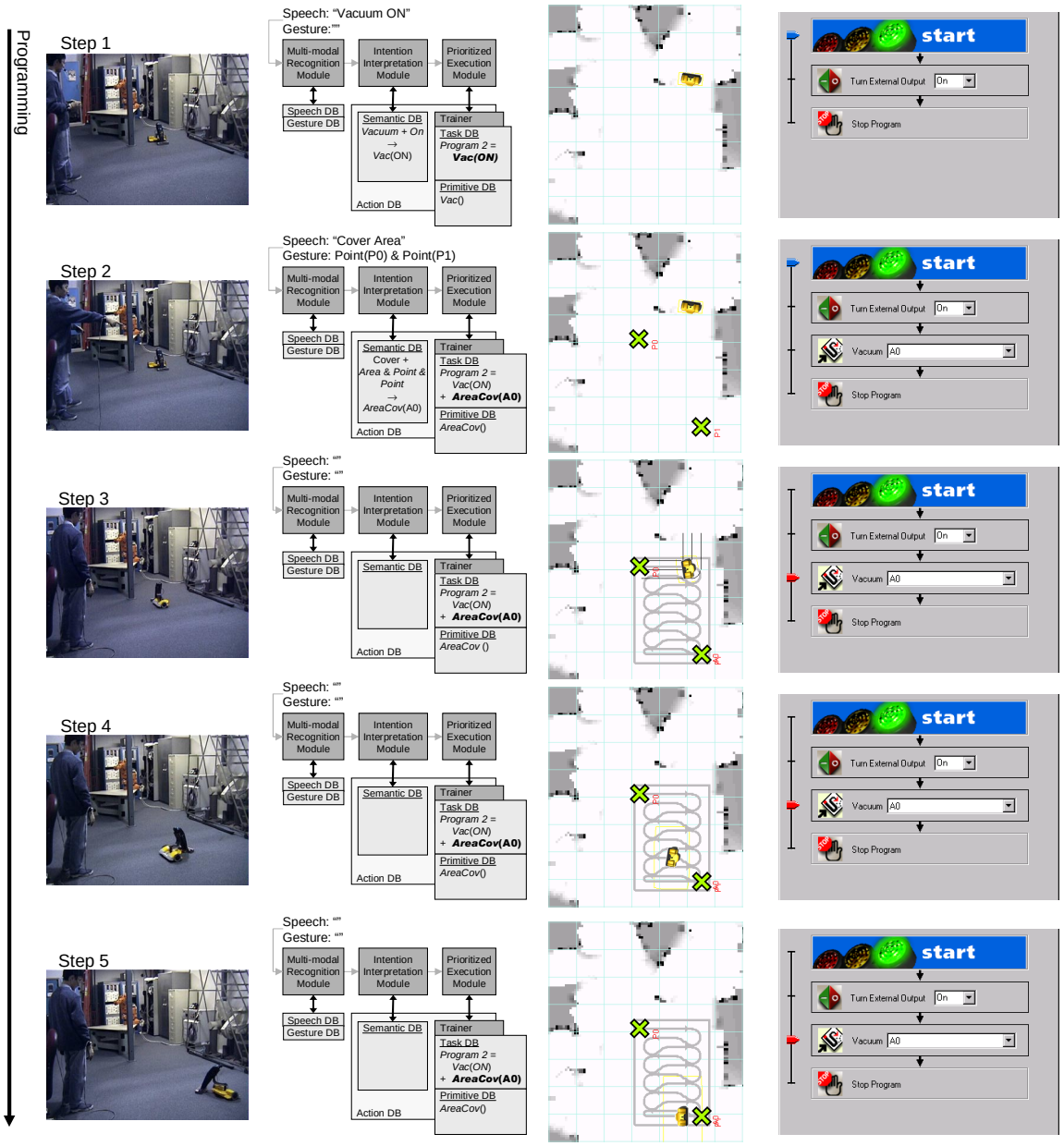


Figure 26: Demonstration Scenario 2

Chapter 8.

Conclusion

I conclude the thesis by going over the overall system design of the multi-modal interactive robot programming framework, the result of the user study, and the implication it has on novice friendly human-robot interaction and knowledge transfer.

8.1. Summary and Discussion

The Interactive Multi-Modal Robot Programming framework described in this thesis is a Programming by Interaction (PBI) system that allows the user to program a robot interactively through an intuitive interface.

The intuitive interface based on hand gesture and spontaneous speech recognition provides simple means to convey symbolic and parametric data in the real environment. Gesture vocabulary was selected from the set of 3D spatial interaction gestures and the set is reduced to match the need of 2D interaction required for mobile vacuum robot control. Gesture recognition module is implemented on top of the Hidden Markov Toolkit, which provides several helpful and necessary features to perform training, adaptation, and spotting recognition. Training and adaptation strategy ensures that the new user can get their gestures recognized. Grammatical network with garbage model is used for gesture spotting.

Error in extracting parameters from the user's hand can be a serious problem. Direction of a vector coming out of an index finger can be significantly different from the user's intended direction if the hand is not well calibrated. During preliminary trial of the user study, one of the users had real hard time controlling Cye using gestures. It turned out that the Polhemus 6DOF position sensor rotated around the user's wrist, and produced a wrong vector. CyberGlove poses a similar problem for the users with large hands, because the fabric from different part of the finger can pull the sensor and produce a similar poor result. The interactive programming framework can deal with poorly localized robot, but it is important to have the user's hand calibrated reasonably well.

Interaction capabilities give a sense of assurance to the users and help them in dealing with disparity between real and modeled environment, by including a human in the control loop. Users are able to initiate a programming phase through voice commands and move the robot to any desired location. The sequence of commands turns into a sequential robot program. The user can then initiate an execution phase and execute the program while taking control at any given time.

The system with intention awareness models recognizes and makes suggestions based on the user's intention. The user's intent is captured in the form of a sequential robot program, and the flexibility given to the user through real-time interaction and the framework's intuitive interface allows the captured intent to be closer to what the user really expects from the robot.

The research questions I addressed at the beginning of the thesis are the following:

- How can one make robot programming easier?
  1) "Do multi-modal user interface improve user's ability to program a robot?"
  2) "Does program suggestion based on intention recognition help robot programming?"
- How can one simplify the interaction with mobile robots?
  3) "Does direct interaction improve user's ability to control a robot?"

Appendix A attempts to answer the last question through comparative experiment between direct gesture based robot control and indirect mouse based robot control. Based on the user study, a gesture based robot control interface performed better than conventional mouse based graphical user interface. The user can keep his/her eyes on the robot in the real environment using gesture based robot control, whereas the user needs to observe graphical user interface and the real environment simultaneously using the graphical user interface. The first two questions remain unanswered. However, the ability to program a robot depends on the ability to manipulate the robot especially for a simple sequential programming task. In order to understand the effects and benefits of a program suggestion, it is necessary to conduct additional user study.

8.2. Dissertation Contributions

This dissertation addresses design, implementation, and evaluation of a novel robot programming paradigm.

- Introduction of Programming by Interaction (PBI) paradigm as a method to enable human-robot knowledge transfer (Iba et al. 2002).
- Design and implementation of the PBI framework (Iba et al. 2002).
- Development of the algorithm which enables program suggestion by the system based on intention awareness (Iba et al. 2003).
- Empirical user study which investigates the benefits of the PBI framework in the domain of mobile vacuum cleaning robot control and programming.

8.3. Future Work

To obtain a comprehensive multi-modal interactive robot programming system, several elements still need to be added in the future. Although the programs generated by the current system can be re-executed, they are limited to fixed task sequences. To expand the generality of the paradigm, we need to add the ability to re-configure the task parameters interactively and define non-sequential flow structures such as conditional branching and looping.

On-line learning of new gestures and speech vocabulary may help the users, since the gesture adaptation will not work for gestures that do not exist in the vocabulary. In the future, it is reasonable to rely on cross-modal gestures to implement on-line learning. There are already attempts to automatically discover new gestures (Wren et al. 2000); however, it is easier to rely on a redundant input mode to manage the learning process. For instance, speech could be used to signal the beginning and end of the learning process for gestures, and vice versa.

Another area where multi-modal input may be beneficial is a multi-robot control domain. The user may specify and control group of people by using two-handed gestures. It is also interesting to extend the framework from vacuum cleaning robot to manipulators. It would be necessary to upgrade the gesture vocabulary from those working in 2D to Manipulator programming domain.

How can user convey intentions in general? My hypothesis is that the user can convey true intention through close interaction through intuitive interface, and I hope that this would lead to the generalized skill acquisition and generalized intelligence in the future.

Appendix A.

User Study

I describe the procedure and results of the user study conducted to investigate the benefits of direct interactive robot control on the multi-modal robot programming environment compared to indirect robot control on a graphical user interface. The user is asked to perform vacuum cleaning tasks in the laboratory environment using the same vacuum cleaning robot and the performance was measured in time, user satisfaction, and the amount of trash collected. This study is added as an appendix since it is a test of the part of the system and it does not necessarily reflect the performance of an entire system, although it can be used to implicate the performance.

A.1. Introduction

Intuitive interface for mobile robot operation is a key requirement for novice friendly robot programming system. Up to this point, the interactive multi-modal robot programming system is implemented under the assumption that direct mobile robot control using hand gestures is more intuitive than indirect mobile robot control on a graphical user interface. In this chapter, I verify this assumption through number of controlled experiments using both interfaces.

Figure 27: Direct Control (pointing)

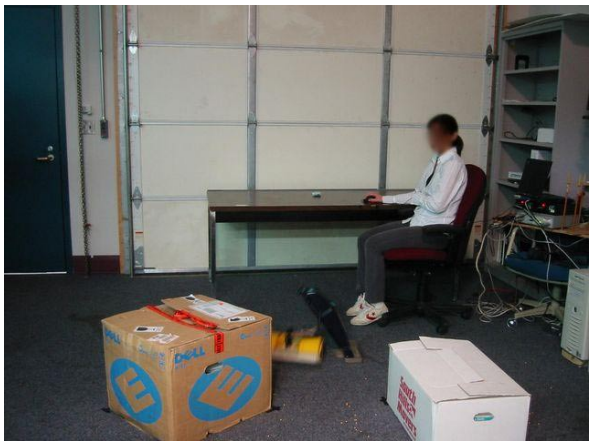


Figure 28: Indirect Control (mouse)

In this set of experiments, terms direct, and indirect robot operation are defined as the following. Direct mobile robot control refers to the method where the test subject faces the environment and the vacuum cleaning robot directly, and operates the robot in a global reference frame. The user is asked to use a pointing hand gesture to specify where the robot should go (Figure 27). Indirect mobile robot control refers to the method where the user faces the environment and the graphical user interface displaying current environment. The user is asked to use a mouse to specify robot destinations in the graphical user interface (Figure 28). Figure 29 summarizes the difference. The system provides the control vector for the robot to execute in both methods. However, the direct method provides an abstraction so that the user is able to specify the control vector directly to the robot,

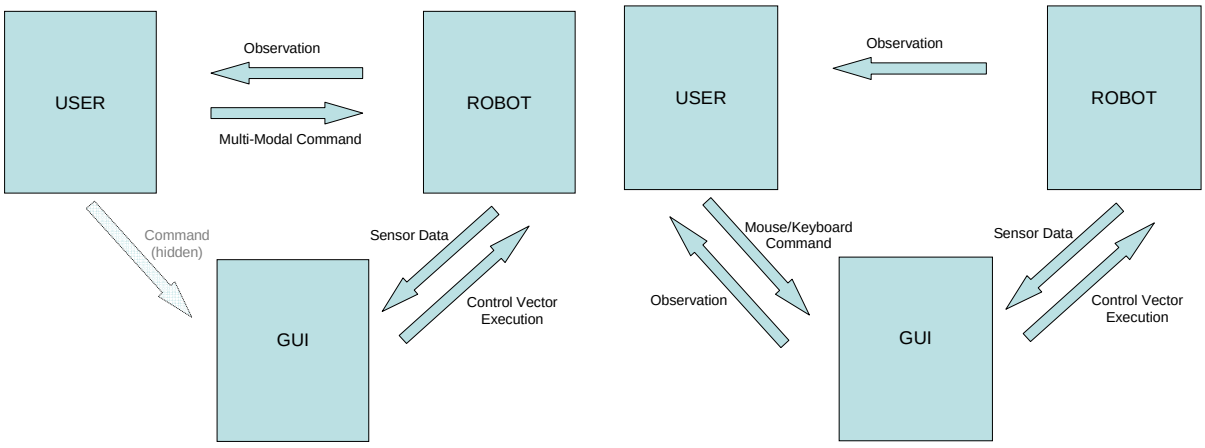


Figure 29: Direct Multi-Modal Interaction vs. Indirect GUI Based Interaction

whereas the user needs to specify the control vector through virtual environment displayed on the graphical user interface, which is not necessarily accurate for the other method. The user needs to verify the correspondence between the real and virtual environment at all time for the latter method.

The hypothesis I am verifying through this experiment is that the gesture based direct control method provides better performance and user satisfaction than the mouse based indirect control method on a graphical user interface. There is a prior user study on mobile robot teleoperation by Fong (Fong 2001), which provides a qualitative evaluation on how users teleoperate the mobile robot. For this project, two methods are tested and compared quantitatively to verify the preference of the user.

A.2. Study Objectives

The objective of the study is to compare gesture based direct robot control method against mouse based indirect robot control method in terms of travel distance, time and amount of plastic pellets collected using the vacuum cleaning mobile robot. The study is going to provide a qualitative basis of comparison between two methods. Since this particular study is a comparison of control methods, almost all vocabularies in the interactive multi-modal robot programming system described in Chapter 4 are not used except for the pointing gesture that drives the robot to the place being pointed at.

A.3. Study Environment

The environment provides an open space for a user to practice controlling a vacuum cleaning robot and an area with obstacles where some degree of agility is required to move a robot around without collision (Figure 30). The regions describe where the plastic pellets are placed for the user to vacuum.

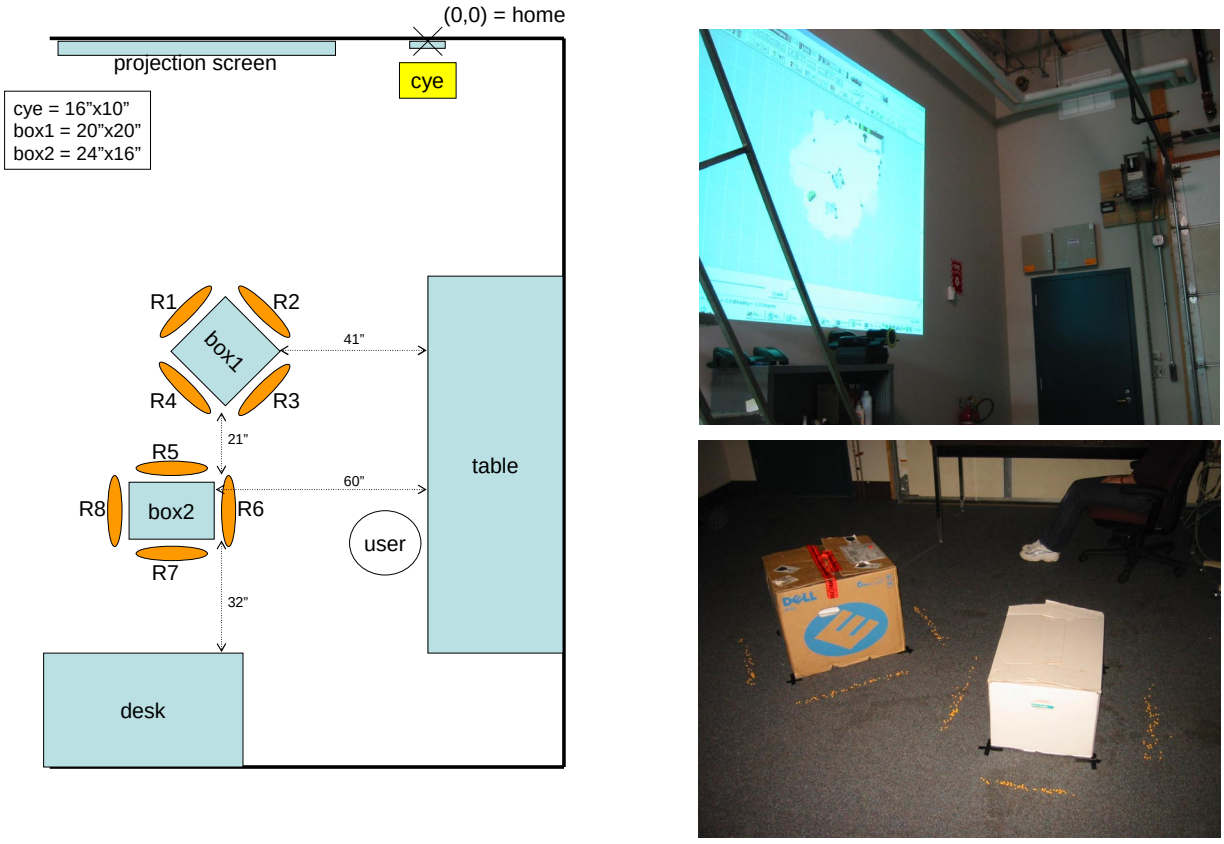


Figure 30: Test Environment

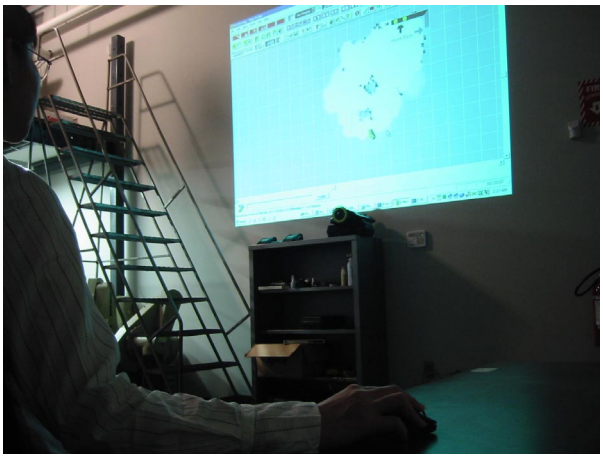


Figure 31: GUI Projected on the Wall

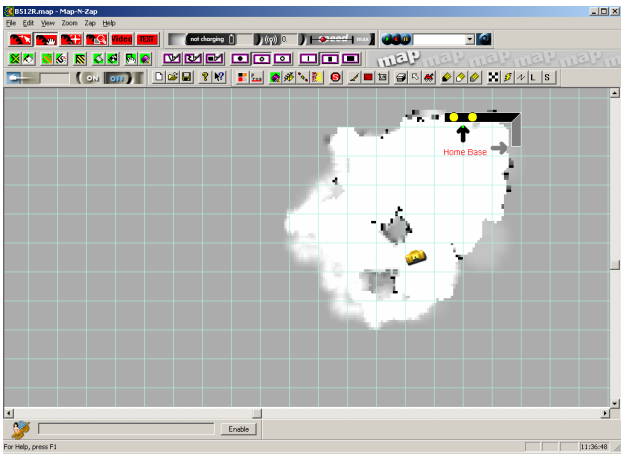


Figure 32: Map-N-Zap Screen Shot

The graphical user interface is projected on the opposite wall for the user to see how the system understands the environment (Figure 31). The probability grid map provided to the user through the projector (Figure 32) is not the perfect representation of the real environment. The pellets that the user must vacuum for the tasks are not displayed in the graphical user interface. The map was created by the robot colliding against obstacles while localizing itself using odometry, and it is unreasonable to assume that the map is perfect. Therefore, it is necessary for the user to check both the graphical user interface and the real environment while operating the robot using a mouse interface. The user can either operate the robot by clicking and dragging the robot using a mouse, or by dragging a robot using a pointing gesture pointing at a place in the real environment.

A.4. Procedure

The user first receives an introduction and the aim of the study. Then the user goes

Figure 33: Distributing Pellets around the Box

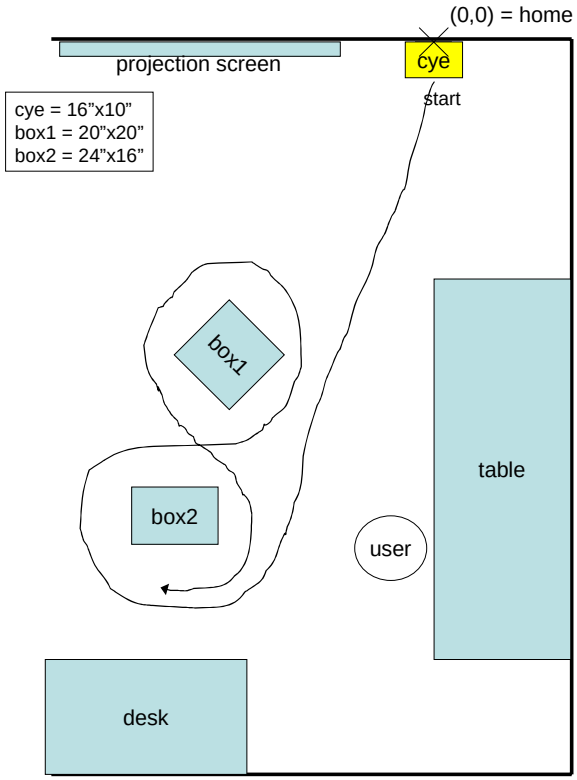


Figure 34: Eight-Curve

through a short gesture adaptation and hand calibration procedure for the gesture recognition module. The user is asked to lay the hand flat on the table to calibrate home position, and follow hand gestures after the study conductor to provide a training sequence.

The user receives an instruction on how to move the robot using a mouse interface on Map-N-Zap application. The user spends three minutes to practice operating the robot. Then the user spends another three minutes operating the robot using a pointing gesture for a practice. The user may move around during both operations. However, the mouse needs to be operated on the table, and the user may not walk with the robot while operating on pointing gestures.

After practicing the basic operation, the conductor distributes plastic pellets (BB bullets of a toy gun, 0.2g/bullet) on the floor. Seven grams of pellets are distributed 7-3/4 inch away from the edge, for each edge of the obstacles in the environment (Figure 33). There are total of 56 grams of pellets on the floor. Then the user is asked to perform the following tasks in given order, using both mouse and hand gesture interface:

(1) Collect as many pellets as possible in undefined path (5 min).
(2) Make as many eight-curves (Figure 34) as possible (3min)
(3) Collect as many pellets as possible in eight-curves (3min)

The task (1) measures the baseline performance of the user on both input modes. During the first task, the user is not experienced enough to collect all pellets, and it is unlikely that the pellets not collected in five minutes be collected in eight or ten minutes. Some edges are difficult for the user to see, and pellets in this region require much more expertise than what was provided prior to the experiment. The performance of the task (1) is measured in terms of total weight of collected pellets.

The task (2) measures smoothness in user interactivity where precision is not a decisive factor. The user needs to operate the robot to draw an eight-curve, and the curve can be wide. Precise control is only required when moving through the center of an eight-curve. If the interaction is smooth, the user is able to draw more curves. If the interaction is not smooth and takes the user more time to operate the robot from point to point, the user is expected to draw less number of eight-curves.

The task (3) measures interaction smoothness and precision. The user is encouraged to make eight-curves while collecting as many pellets as possible. Since pellets are placed close to the obstacle edges, the user needs to improve in precision from what was performed in the last task. The performance of the task (3) is measured in terms of total weight of collected pellets and number of eight-curves made during collection. The number also provides information on how well the user learned to user the interface by comparing the number against task (1).

A.5. Results

The study is conducted on three users with variety of PC gaming (mouse operation) and robotics experiences. The users A and C are novice in both PC gaming and robot teleoperation. Their limited experience in gesture based interaction and mouse usage

User A

	Task 1	Task 2	Task 3
Pointing Gesture	**44g**	**3+1/8 rounds**	**35g, 1+3/8 rounds**
Mouse	**46g**	**2 rounds**	**24g, 1+2/8 rounds**

User B

	Task 1	Task 2	Task 3
Pointing Gesture	**54g**	**3+5/8 rounds**	**51g, 1+3/8 rounds**
Mouse	**48g**	**3+4/8 rounds**	**45g, 1+4/8 rounds**

User C

	Task 1	Task 2	Task 3
Pointing Gesture	**43g**	**3+1/8 rounds**	**38g, 1+6/8 rounds**
Mouse	**20g**	**1+5/8 rounds**	**38g, 1+7/8 rounds**

Table 7: Test Performances

provide unbiased basis of comparison on how novice users react to the robot interaction study. User B is an experienced user in both gaming and robot operations.

Table 7 summarizes the performance of each task for each interface. In general, collection performances in the regions far from the user were the worst due to visibility and accumulated sensor errors.

Comparison between direct and indirect interaction for the task (1) shows that the performance between two interfaces are comparable, given initial practice and enough time to operate. User A tried to clear the region in order by repeatedly vacuuming on the same region then moving on to the next one. Region 1 and 2 were apparently hard for a pointing gesture interface because of positioning error caused by the accumulation of error in Polhemus position sensor reading and error in hand calibration. Error in pointing position is always present unless the position sensor and the hand sensors are calibrated perfectly. The same regions were hard for a mouse interface too because of inaccurate map combined with blocked view from the user.

The task (2) provides basis of comparison for the interfaces where the criteria is smoothness in interaction. The mouse based interaction through graphical user interface requires that the user constantly switch his/her view between the robot in real environment and graphical user interface, since the control input is provided by manipulating a mouse pointer in the graphical user interface. This “two-view” problem is inherent to the system with graphical user interface to a robot in real environment, unless the model can provide accurate information to the user. The gesture based direct interaction performed very well in this task, since pointing precision was not a deciding factor, while the user could face the robot to provide control input at all time. Although there are inherent problems in both interfaces, having to look at two environments (real environment and virtual environment in graphical user interface) at one time is a structural problem.

Comparison between the interfaces for the task (3) provides interesting result where smoothness and precision are required from the robot operation. It was apparently hard for the subject to provide an accurate positioning through mouse interface due to the “two-view” problem. To do so, the subject is required to click on the robot in the graphical user interface, followed by dragging the mouse to move the robot to a good position without looking at the graphical user interface. This turned out to be the most confusing aspect of

the mouse based interaction, and the subject preferred the gestural direct interaction for the fact that there is no need to look at two places at one time.

A.6. Summary

The user study was performed to investigate the benefits of direct interactive robot control on the multi-modal robot programming environment, compared to indirect robot control on a graphical user interface. Based on qualitative measurements of the performances of three tasks, “two-view” problem in the mouse based indirect interaction is evident for the user. Gesture based direct robot control allowed the user to keep focusing on the robot and the environment.

Appendix B.

User Study Support Document

This appendix[1] contains documents related to the user study performed in Appendix A.

B.1. User Study Support Document

The following is excerpted from CMU Protocol HS03-346, under the project title "Interactive Multi-Modal Robot Programming". All questions regarding the approval process and certification of this research protocol should be addressed to:

Institutional Review Board
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213-3890

[1] I followed the format of a similar appendix in (Fong 2001) which dealt with the user study on mobile robot teleoperation.

B.1.1. User Study Application Cover Page

CARNEGIE MELLON UNIVERSITY
HUMAN SUBJECTS IRB APPLICATION

DATE: **Nov 17, 2003** CMU Protocol No. _____ (for office use only)

New Request **X** Renewal _____

Principal Investigator(s): **Pradeep Khosla (supervisor) / Soshi Iba**

P.I. Title/Degree: **Dr. / Ph.D Student** Department **Robotics Institute**

Phone: **x8-4864** E-mail: **iba@cs.cmu.edu**

Project Dates: From **Nov 24, 2003** To **Feb 31, 2003**

Project Title: **Interactive Multi-Modal Robot Programming**

Name of Experimenter(s): **Soshi Iba**

Source of Funding (Sponsor): Internal: _____ External: ARO DAAD19-02-1-0389

Brief Description of Research: **Quantitative evaluation of an on-line mobile robot programming system using hand gestures and speech interface**

1. How many subjects will be used in this experiment? **8**

2. From what source do you plan to obtain subjects? **Volunteers (student & community)**

3. Is there any benefit gained by the subject for participating? **No monetary benefits**

4. Will the subjects include any of the following: **NO** **X** **YES** (please check below)

_____ Fetuses	_____ Mentally Retarded
_____ Hospitalized Patients	_____ Minors
_____ Institutionalized Patients	_____ Pregnant Women
_____ Mentally Disabled	_____ Prisoners

5. Degree of Physical Risk: _**X**__Negligible ____Mild ____Moderate ____High

6. Degree of Psychological Risk: _**X**__Negligible ____Mild ____Moderate ____High

7. Do you or any individual who is associated with/responsible for the design, the conduct, or the reporting of this research have an economic interest in or act as an officer or a director for any outside entity whose financial interests would reasonably appear to be affected by this research project?

_____**Yes*** ____**X**____**No**

(*if yes, please provide detailed information to permit the IRB to determine if such involvement should be disclose to potential research subjects.)

Please submit each of the following with this Clearance Request form:

1. A draft of the proposal or abstract
2. A clear definition of how the subjects will be utilized or how the experimental treatment will be administered
3. A copy of the "informed" consent form(s) that the subjects will be required to sign
4. An indication of how confidentiality/anonymity will be protected
5. The name(s) and address(es) of official(s) authorizing access to any subjects in cooperating institutions not under the direct control of Carnegie Mellon
6. Risk/Benefit analysis
7. A statement describing how participants will be recruited (include advertisement flyers/invitation letters/invitation emai
8. A copy of your on-line training certificate (http://cme.nci.nih.gov/)

B.1.2. User Study Proposal

Background

Novice friendly user interface is an important aspect to a successful robotic system, as personal robots such as vacuum cleaning robots are becoming increasingly popular (Musser 2003). However, currently available modes to control mobile robots still require expertise. For example, teleoperating a mobile robot using joystick, keyboard, or mouse through a graphical user interface (GUI) requires an user to make two types of transitions during the operation: from a real scene to the map-oriented world in a GUI, and from the user's perspective to the robot's perspective. Moreover, automating a vacuum cleaning task through robot programming can be a challenging problem for novices.

To make robot control and robot programming more accessible to all users, novices and experts alike, I have developed interactive multi-modal robot programming system based on an interactive robot programming paradigm (Iba, Paredis, and Khosla 2002). In this system, the user faces a mobile vacuum cleaning robot directly, and navigates the robot using speech and hand-gestures. Symbolic commands such as "Vacuum On", "Stop", or "Execute Program One" are conveyed through speech, whereas hand-gestures are used to complement deictic terms in the commands such as "Go *There*", "Move *This* Way", or "Vacuum *This* Area" to convey symbolic and parametric commands. Since gestures and speech are more direct mode of inputs than mouse and keyboard, they help the users from having to make connections between motions in the real world and motions displayed on the GUI. The user can create a sequential program by first setting the system into a programming mode, and then controlling the robot to register actions. In order to further assist the user, the system is able to suggest the robot program that the user may want to execute, based on its observation history (Iba, Paredis, and Khosla 2003). For example, if the user wants to vacuum the area in a certain path (around a coffee table, then in front of a TV, then around a dining table, etc.) the system may suggest a program does just that given observations of the user controlling the robot in a similar path for a while. Such suggestion will be made on a map of the GUI by displaying the program execution of a simulated robot. The user may accept or simply ignore suggestions. The user may interrupt the program and take over the control at any time to give user the sense of being in full control.

Objective

In this study, I intend to acquire quantitative measurement of four potential advantageous factors of the interactive multi-modal programming system. They are;

1) ability for the user to interrupt robot execution at any time
2) capability to interact directly with a mobile robot without GUI in the middle
3) capability of the system to suggest the most likely program to the user
4) capability to create robot programs on-line

In order to evaluate and compare these factors quantitatively, I am going to measure the performance of the robotic vacuum cleaning task with and without these factors in controlled experiments. The measurements used are;

a) percentage of plastic beads collected from the floor
b) time required to sweep the floor
c) distance traveled by the vacuum cleaning robot
d) number of purposeful gestures executed by the user

Above measurements will be used to support the claim that four factors in the interactive multi-modal programming system are advantageous against robot control and programming systems with conventional paradigms. Only volunteers will be used in this study, and they will be chosen from different robotics and PC experience levels.

References

Iba, S., Paredis, C. J. J., and Khosla, P. K., "Interactive Multi-Modal Robot Programming," International Conf. on Robotics and Automations, Washington, D.C., pp. 161-68, 2002.

Iba, S., Paredis, C. J. J., and Khosla, P. K., "Intention Aware Interactive Multi-Modal Robot Programming," International Conf. on Intelligent Robots and Systems, Las Vegas, NV., 2003.

Musser, G., "Robots That Suck," Scientific American, vol. 288, no. 2, pp. 84-6, 2003.

B.1.3. Consent Form

CARNEGIE MELLON UNIVERSITY

CONSENT FORM

Project Title: **Interactive Multi-Modal Robot Programming**
Conducted By: Soshi Iba

I agree to participate in the observational research conducted by Professor Khosla or by students or staff under the supervision of Professor Khosla. I understand that the proposed research has been reviewed by the University's Institutional Review Board and that to the best of their ability they have determined that the observations involve no invasion of my rights of privacy, nor do they incorporate any procedure or requirements which may be found morally or ethically objectionable. If, however, at any time I wish to terminate my participation in this study I have the right to do so without penalty.

If you have any questions about this study, you should feel free to ask them now or anytime throughout the study by contacting:

Professor Pradeep Khosla
Electrical and Computer Engineering Department
5000 Forbes Avenue, HBH1106
412-268-5090
pkk@ece.cmu.edu

You may report any objections to the study, either orally or in writing to:

Dr. Ann Baldwin Taylor, IRB Chair
at0j@andrew.cmu.edu
Carnegie Mellon University
(412) 268-4727

Purpose of the Study: I understand I will be learning about a system to control and program a mobile vacuum cleaning robot interactively. I know that the researchers are studying how well people can control a robot using multi-modal interface such as hand gestures and speech. I realize that in the experiment I will learn how to control and program a mobile vacuum cleaning robot using the system, and then explore controlled experimental tasks for about an hour in a lab environment.

Study Procedure: I understand that, in this study, I will first complete a short questionnaire. After a briefing of the study goals and system preparation, I will be asked to perform few tasks. Soshi Iba will observe what I am doing and record the process through video and PC while I am working.

I understand that the following procedure will be used to maintain my anonymity in analysis and publication/presentation of any results. Each participant will be assigned a number, names will not be recorded. The researchers will save the data and videotape files by participant number, not by name. Only members of the research group will view the tapes in detail. The videotapes and records will be stored in locked files by Soshi Iba until May 2004. No other researchers will have access to these files.

Optional Permission: I understand that the researchers may want to use a short portion of a videotape for illustrative reasons in presentations of this work. I give my permission to do so provided that my name and face will not appear.

_____ YES _____ NO (Please initial here _____)

I understand that in signing this consent form, I give Professor Khosla and his associates, permission to present this work in written and oral form, without further permission from me.

_____ _____
Name (please print) Signature

_____
Date

B.1.4. How Subjects Will Be Utilized

The study will be conducted in a lab (NSH-B512). I will administer all tests with the authorization of Dr. Pradeep Khosla. All subjects will be given a short, one-page questionnaire (on the next page) to establish background information. The questionnaire will be used solely to ascertain education and experience in mobile robot control and robot programming. No names or other biographical data will be collected. Approximately 60~70 minutes will be required per subject. No physical work will be required other than hand gesture execution (on data groves), speech, and computer mouse and keyboard operation. Psychological risk is expected to be negligible or zero. Data collection method will be primarily performed within the PC, and there will be a video recording of the experiment.

B.1.5. Confidentiality

To safeguard the anonymity and confidentiality of study subjects, each subject will be assigned a number. All collected data (written notes) will be referenced by this number. No names or other biographical information will be recorded.

B.1.6. Risk and Benefit Analysis

The level of physical and psychological risk is negligible. The subjects will perform no physical labor.

This study will provide valuable evaluation of a new paradigm, interactive multi-modal robot programming, to robot control and programming domain. As such, it will benefit researchers and engineers working to construct more flexible and effective robot control and programming system. Also, this study will provide evidence to help support my claim that features of this new paradigm is beneficial to both novice and experts in control and programming of mobile robot system.

In this study, the subjects will learn about robot control and programming. The subjects will have the opportunity to operate mobile vacuum cleaning robot and will gain first-hand experience with hand gesture recognition and robot programming.

B.1.7. Participant Recruitment

All participants of the study are volunteers and will be given no monetary benefits. Volunteers will participate in the study solely from the interest in the new paradigm to control and program mobile vacuum cleaning robot through speech and hand gestures. Recruitment will be based on e-mail and verbal invitations.

B.1.8. NIH Training Certificate

Completion Certificate

This is to certify that

Soshi Iba

has completed the **Human Participants Protection Education for Research Teams** online course, sponsored by the National Institutes of Health (NIH), on 09/25/2003.

This course included the following:

- key historical events and current issues that impact guidelines and legislation on human participant protection in research.
- ethical principles and guidelines that should assist in resolving the ethical issues inherent in the conduct of research with human participants.
- the use of key ethical principles and federal regulations to protect human participants at various stages in the research process.
- a description of guidelines for the protection of special populations in research.
- a definition of informed consent and components necessary for a valid consent.
- a description of the role of the IRB in the research process.
- the roles, responsibilities, and interactions of federal agencies, institutions, and researchers in conducting research with human participants.

National Institutes of Health
http://www.nih.gov

B.2. Quantitative User Study Procedure

Objectives

The focus of this study is to acquire quantitative measurement of four factors of the interactive multi-modal programming system to compare against iconic programming system. They are;

1) ability for the user to interrupt robot execution at any time
2) capability to interact directly with a mobile robot without GUI in the middle
3) capability of the system to suggest the most likely program to the user
4) capability to create robot programs on-line

Study Procedure

1. Introduction

- Presentation of study goals, and study procedure (duration of test, what will be recorded and how, etc.)
- Description of confidentiality procedures
- Subject will be asked to read and sign the informed consent form

2. Empirical Investigation

- Subject will practice GUI based robot control and program system (5 min)
- Subject will be asked to read few sentences to perform speech recognition system training (~1 minute)
- Subject will be asked to perform a sequence of hand gestures to train and adapt gesture recognition system (~5 minutes)
- Gesture and speech recognition system is tested, and retrained if necessary
- Presentation of interactive multi-modal robot programming system (3 min)
- Subject will practice multi-modal robot control and program a sample task (3 min)
- Subject will practice robot control and program a sample task on GUI based system (5 min)
- Subject will be asked to perform the following controlled experiments;
    - Navigate the vacuum cleaning robot using gestures and speech to collect groups of visible plastic pellets placed far from each other. Do this with and without interrupt capability. (max 3 min each)

- Navigate the vacuum cleaning robot using gestures and speech to collect groups of visible plastic pellets placed far from each other, in an environment with few obstacles. Perform the task with and without program suggestion. (max 3 min each)
- Navigate the vacuum cleaning robot to collect as many plastic pellets as possible in a given time period. Do the task first without GUI using gestures and speech, and then on GUI with mouse. (1 min each)
- Program the robot to follow an exact path on GUI using mouse, and then without GUI using gesture and speech. Execute both programs to compare the path. (max 3 min each)

As the subject works, the test administrator records the user action on video as well as on the PC. The test administrator will put a short break in between the preparation phase and the controlled experiments.

3. Wrap-up

- Short discussion regarding the experience. (will be recorded on video)
- Conclusion (ask if can contact if needed, thank for participating)

B.2.1. User Questionnaire

General:

Age: ___
Gender: _____

Education (highest degree obtained):

__ High School
__ Bachelor's
__ Master's
__ Ph.D.
Major: _____

Computer Experience:

How often do you use a computer (please check one)
__ daily __ weekly __ monthly __ rarely __ never
What type of task do you perform on a computer (check all that apply):
__ word processing __ gaming __ slide presentation __ web browsing __ email
__ drawing __ other (please describe:)
How do you rate yourself on computer expertise, on a scale of 1(novice) to 5(expert)
___

Video Game Experience:

How often do you play video games:

__ daily __ weekly __ monthly __ rarely __ never

What type of games do you play (check all that apply):

__ action __ card __ simulation __ adventure __ role-playing __ other

Robotics Experience:

Have you ever tele-operated a robot?:

__ yes __ no

Have you ever programmed a robot?:

__ yes __ no

If yes, please describe the robot and its task:

_____

Appendix C.

Merging Sampled Statistics without Prior Samples

The following is the derivation of the method used to merge *p*-dimensional sampled statistics of set *X* and set *Y* without using samples themselves (Koyama 2002). In other words, solve for $\underset{\sim}{\overline{Z}}$ and $S_Z{}^2$ from n_X, $\underset{\sim}{\overline{X}}$, $S_X{}^2$, n_Y, $\underset{\sim}{\overline{Y}}$, and $S_Y{}^2$ (without *X* and *Y*).

Sample Set	Size	Mean	Variance
$X = \left(\underset{\sim}{X}_1, \cdots, \underset{\sim}{X}_{n_X} \right)$	n_X	$\underset{\sim}{\overline{X}}$	$S_X{}^2$
$Y = \left(\underset{\sim}{Y}_1, \cdots, \underset{\sim}{Y}_{n_Y} \right)$	n_Y	$\underset{\sim}{\overline{Y}}$	$S_Y{}^2$
$Z = \left(\underset{\sim}{X}_1, \cdots, \underset{\sim}{X}_{n_X}, \underset{\sim}{Y}_1, \cdots, \underset{\sim}{Y}_{n_Y} \right)$	$n_Z = n_X + n_Y$	$\underset{\sim}{\overline{Z}}$	$S_Z{}^2$

Given samples:

$$\underset{(p \times n_X)}{X} = \left(\underset{\sim}{X}_1 \cdots \underset{\sim}{X}_{n_X} \right) = \begin{bmatrix} X_{11} & \cdots & X_{n_X 1} \\ \vdots & \ddots & \vdots \\ X_{1p} & \cdots & X_{n_X p} \end{bmatrix}$$

$$\underset{(p \times n_Y)}{Y} = \left(\underset{\sim}{Y}_1 \cdots \underset{\sim}{Y}_{n_Y} \right) = \begin{bmatrix} Y_{11} & \cdots & Y_{n_Y 1} \\ \vdots & \ddots & \vdots \\ Y_{1p} & \cdots & Y_{n_Y p} \end{bmatrix}$$

where $\underset{\sim}{X}_1 = \begin{bmatrix} X_{11} \\ \vdots \\ X_{1p} \end{bmatrix} \cdots \underset{\sim}{X}_{n_X} = \begin{bmatrix} X_{n_X 1} \\ \vdots \\ X_{n_X p} \end{bmatrix}$ and $\underset{\sim}{Y}_1 = \begin{bmatrix} Y_{11} \\ \vdots \\ Y_{1p} \end{bmatrix} \cdots \underset{\sim}{Y}_{n_Y} = \begin{bmatrix} Y_{n_Y 1} \\ \vdots \\ Y_{n_Y p} \end{bmatrix}$

Merged sample mean:

$$\underset{(p \times 1)}{\underset{\sim}{\overline{Z}}} = \frac{n_X \underset{\sim}{\overline{X}} + n_Y \underset{\sim}{\overline{Y}}}{n_Z}$$

where $\underset{(p\times 1)}{\underset{\sim}{\bar{X}}} = \begin{bmatrix} \bar{X}_1 \\ \vdots \\ \bar{X}_p \end{bmatrix}$ and $\underset{(p\times 1)}{\underset{\sim}{\bar{Y}}} = \begin{bmatrix} \bar{Y}_1 \\ \vdots \\ \bar{Y}_p \end{bmatrix}$

Merged sample variance:

$$\underset{(p\times p)}{S_Z^2} = \frac{1}{n_Z - 1}\Big\{(n_X - 1)S_X^2 + n_X(\underset{\sim}{\bar{X}} - \underset{\sim}{\bar{Z}})(\underset{\sim}{\bar{X}} - \underset{\sim}{\bar{Z}})^T + (n_Y - 1)S_Y^2 + n_Y(\underset{\sim}{\bar{Y}} - \underset{\sim}{\bar{Z}})(\underset{\sim}{\bar{Y}} - \underset{\sim}{\bar{Z}})^T\Big\}$$

given

$$\underset{(p\times p)}{S_X^2} = \frac{1}{n_X - 1} W_X W_X^T$$

where $\underset{(p\times n_X)}{W_X} = X - \underset{\sim}{\bar{X}} \cdot \underset{\sim}{1}^T = \begin{bmatrix} X_{11} & \cdots & X_{n_X 1} \\ \vdots & \ddots & \vdots \\ X_{1p} & \cdots & X_{n_X p} \end{bmatrix} - \begin{bmatrix} \bar{X}_1 & \cdots & \bar{X}_1 \\ \vdots & \ddots & \vdots \\ \bar{X}_p & \cdots & \bar{X}_p \end{bmatrix}$

and

$$\underset{(p\times p)}{S_Y^2} = \frac{1}{n_Y - 1} W_Y W_Y^T$$

proof

$$\underset{(p\times p)}{S_Z^2} = \frac{1}{n_X + n_Y - 1} W_Z W_Z^T$$

where $\underset{(p\times n_Z)}{W_Z} = \begin{bmatrix} X_{11} & \cdots & X_{n_X 1} & Y_{11} & \cdots & Y_{n_Y 1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ X_{1p} & \cdots & X_{n_X p} & Y_{1p} & \cdots & Y_{n_Y p} \end{bmatrix} - \underbrace{\begin{bmatrix} \bar{Z}_1 & \cdots & \bar{Z}_1 \\ \vdots & \cdots & \vdots \\ \bar{Z}_p & \cdots & \bar{Z}_p \end{bmatrix}}_{n_Z \text{ columns}}$

$$= \frac{1}{n_Z - 1}\left\{\sum_{i=1}^{n_X}\{(\underset{\sim}{X}_i - \underset{\sim}{\bar{Z}})(\underset{\sim}{X}_i - \underset{\sim}{\bar{Z}})^T\} + \sum_{j=1}^{n_Y}\{(\underset{\sim}{Y}_j - \underset{\sim}{\bar{Z}})(\underset{\sim}{Y}_j - \underset{\sim}{\bar{Z}})^T\}\right\}$$

whose $\sum_{i=1}^{n_X}\{(\underset{\sim}{X}_i - \underset{\sim}{\bar{Z}})(\underset{\sim}{X}_i - \underset{\sim}{\bar{Z}})^T\}$

$$= \sum_{i=1}^{n_X}\{(\underset{\sim}{X}_i - \underset{\sim}{\bar{X}} + \underset{\sim}{\bar{X}} - \underset{\sim}{\bar{Z}})(\underset{\sim}{X}_i - \underset{\sim}{\bar{X}} + \underset{\sim}{\bar{X}} - \underset{\sim}{\bar{Z}})^T\}$$

$$= \sum_{i=1}^{n_X} \left\{ (\underset{\sim}{X}_i - \overline{\underset{\sim}{X}})(\underset{\sim}{X}_i - \overline{\underset{\sim}{X}})^T \right\} + \sum_{i=1}^{n_X} \left\{ (\overline{\underset{\sim}{X}} - \overline{\underset{\sim}{Z}})(\overline{\underset{\sim}{X}} - \overline{\underset{\sim}{Z}})^T \right\}$$

$$+ \sum_{i=1}^{n_X} \left\{ (\underset{\sim}{X}_i - \overline{\underset{\sim}{X}})(\overline{\underset{\sim}{X}} - \overline{\underset{\sim}{Z}})^T \right\} + \sum_{i=1}^{n_X} \left\{ (\overline{\underset{\sim}{X}} - \overline{\underset{\sim}{Z}})(\underset{\sim}{X}_i - \overline{\underset{\sim}{X}})^T \right\}$$

$$= \sum_{i=1}^{n_X} \left\{ (\underset{\sim}{X}_i - \overline{\underset{\sim}{X}})(\underset{\sim}{X}_i - \overline{\underset{\sim}{X}})^T \right\} + n_X (\overline{\underset{\sim}{X}} - \overline{\underset{\sim}{Z}})(\overline{\underset{\sim}{X}} - \overline{\underset{\sim}{Z}})^T$$

$$+ \underbrace{\sum_{i=1}^{n_X} \left\{ (\underset{\sim}{X}_i - \overline{\underset{\sim}{X}}) \right\}}_{=0} (\overline{\underset{\sim}{X}} - \overline{\underset{\sim}{Z}})^T + (\overline{\underset{\sim}{X}} - \overline{\underset{\sim}{Z}}) \underbrace{\sum_{i=1}^{n_X} (\underset{\sim}{X}_i - \overline{\underset{\sim}{X}})^T}_{=0}$$

$$= \sum_{i=1}^{n_X} \left\{ (\underset{\sim}{X}_i - \overline{\underset{\sim}{X}})(\underset{\sim}{X}_i - \overline{\underset{\sim}{X}})^T \right\} + n_X (\overline{\underset{\sim}{X}} - \overline{\underset{\sim}{Z}})(\overline{\underset{\sim}{X}} - \overline{\underset{\sim}{Z}})^T$$

$$= W_X {W_X}^T + n_X (\overline{\underset{\sim}{X}} - \overline{\underset{\sim}{Z}})(\overline{\underset{\sim}{X}} - \overline{\underset{\sim}{Z}})^T$$

$$= (n_X - 1){S_X}^2 + n_X (\overline{\underset{\sim}{X}} - \overline{\underset{\sim}{Z}})(\overline{\underset{\sim}{X}} - \overline{\underset{\sim}{Z}})^T$$

similarly,

$$\sum_{j=1}^{n_Y} \left\{ (\underset{\sim}{Y}_j - \overline{\underset{\sim}{Z}})(\underset{\sim}{Y}_j - \overline{\underset{\sim}{Z}})^T \right\} = (n_Y - 1){S_Y}^2 + n_Y (\overline{\underset{\sim}{Y}} - \overline{\underset{\sim}{Z}})(\overline{\underset{\sim}{Y}} - \overline{\underset{\sim}{Z}})^T$$

$$\therefore {S_Z}^2 = \frac{1}{n_Z - 1} \Big\{ (n_X - 1){S_X}^2 + n_X (\overline{\underset{\sim}{X}} - \overline{\underset{\sim}{Z}})(\overline{\underset{\sim}{X}} - \overline{\underset{\sim}{Z}})^T$$

$$+ (n_Y - 1){S_Y}^2 + n_Y (\overline{\underset{\sim}{Y}} - \overline{\underset{\sim}{Z}})(\overline{\underset{\sim}{Y}} - \overline{\underset{\sim}{Z}})^T \Big\}$$

References

Agah, A., and Tanie, K. (1996). "Human-Machine Interaction Through an Intelligent User Interface Based on Contention Architecture." *IEEE International Workshop on Robot and Human Communication RO-MAN'96*, Tsukuba, Japan, 537-42.

Bahlmann, C., and Burkhardt, H. (2001). "Measuring HMM similarity with the Bayes probability of error and its application to online handwriting recognition." *Sixth International Conference on Document Analysis and Recognition*, Seattle, WA, USA, 406-411.

Batavia, P. H., and Nourbakhsh, I. (2000). "Path planning for the Cye personal robot." *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 15-20.

Boehme, H. J., Brakensiek, A., Braumann, U. D., Krabbes, M., and Gross, H. M. (1997). "Neural architecture for gesture-based human-machine- interaction." *Gesture and Sign Language in Human-Computer Interaction*, Bielefeld, Germany, 219-32.

Bremmer, J. N., and Roodenburg, H. (1992). *A Cultural history of gesture*, Cornell University Press, Ithaca, N.Y.

Caminero, J., De La Torre, D., Villarrubia, L., Martin, C., and Hernandez, L. (1996). "On-line garbage modeling with discriminant analysis for utterance verification." *Proceedings ICSLP 96. Fourth International Conference on Spoken Language Processing*, New York, NY, 2111-2114.

Conway, L., and Cohen, C. J. (1998). "Video mirroring and iconic gestures: enhancing basic videophones to provide visual coaching and visual control." *IEEE Transactions on Consumer Electronics*, 44(2), 388-97.

Costanzo, C., Iannizzotto, G., and La Rosa, F. (2003). "Virtualboard: real-time visual gesture recognition for natural human-computer interaction." *IEEE International Parallel and Distributed Processing Symposium*, Nice, France, 112-119.

CyberGlove Reference Manual (1998). Virtual Technologies Inc., Palo Alto, CA.

Dillmann, R., Rogalla, O., Ehrenmann, M., Zollner, R., and Bordegon, M. (1999). "Learning robot behaviour and skills based on human demonstration and advice: the machine learning paradigm." *9th International Symposium of Robotics Research*, Snowbird, Utah, 229-238.

Dixon, K. R. (2004). "Inferring User Intent for Learning by Observation," PhD thesis, Department of Electrical & Computer Engineering, Carnegie Mellon University, Pittsburgh.

Ehrenmann, M., Zollner, R., Rogalla, O., and Dillmann, R. (2002). "Programming service tasks in household environments by human demonstration." *IIEEE International Workshop on Robot and Human Interactive Communication*, Piscataway, NJ, USA, 460-467.

Fong, T. (2001). "Collaborative Control: A Robot-Centric Model for Vehicle Teleoperation," PhD thesis, The Robotics Institute, Carnegie Mellon University, Pittsburgh.

Forsyth, D., and Ponce, J. (2003). *Computer Vision : A Modern Approach*, Prentice Hall, Upper Saddle River, N.J.

Freeman, W. T., Anderson, D., Beardsley, P., Dodge, C., Kage, H., Kyuma, K., Miyake, Y., Roth, M., Tanaka, K., Weissman, C., and Yerazunis, W. (1998). "Computer vision for interactive computer graphics." *IEEE Computer Graphics and Applications*, 18(3), 42-53.

Friedrich, H., Dillmann, R., and Rogalla, O. (1999). "Interactive robot programming based on human demonstration and advice." Sensor Based Intelligent Robots. International Workshop. Selected Papers (Lecture Notes in Artificial Intelligence Vol.1724), H. I. B. H. N. H. Christensen, ed., Springer-Verlag, Berlin, Germany, 96-119.

Fujita, M. (2001). "AIBO: toward the era of digital creatures." *International Journal of Robotics Research*, 20(10), 781-794.

Gertz, M., Stewart, D., and Khosla, P. K. (1993). "A software architecture-based human-machine interface for reconfigurable sensor-based control systems." *IEEE International Symposium on Intelligent Control*, Chicago, IL, USA, 75-80.

Gertz, M. W., and Khosla, P. K. (1994). "Iconic language for reconfigurable sensor-based control systems." *Annual Meeting of American Nuclear Society*, New Orleans, LA, USA, 420-421.

Ghidary, S. S., Nakata, Y., Saito, H., Takamori, T., and Hattori, M. (2001). "Multi-Modal Human Robot Interaction for Map Generation." *International Conference on Intelligent Robot and Systems*, Maui, Hawaii, USA, 2246-2251.

Gruver, W. A., Soroka, B. I., Craig, J. J., and Turner, T. L. (1984). "Industrial robot programming languages: a comparative evaluation." *IEEE Transactions on Systems, Man and Cybernetics*, 14(4), 565-70.

Han, K., and Veloso, M. (1999). "Automated robot behavior recognition applied to robotic soccer." *In Proceedings of the 9th International Symposium of Robotics Research (ISSR'99)*, 199-204.

Hexmoor, H., and Yang, J. (2000). "Pointing: A Component of a Multimodal Robotic Interface." *Workshop in Interactive and Entertainment Robots (WIRE-2000)*, CMU, Pittsburgh, PA, 103-107.

Huang, X., Alleva, F., Hon, H.-W., Hwang, M.-Y., and Rosenfeld, R. (1993). "The SPHINX-II speech recognition system: an overview." *Computer Speech and Language*, 7(2), 137-48.

Iba, S., Paredis, C. J. J., and Khosla, P. K. (2002). "Interactive Multi-Modal Robot Programming." *International Conf. on Robotics and Automations*, Washington, D.C., 161-168.

Iba, S., Paredis, C. J. J., and Khosla, P. K. (2003). "Intention Aware Interactive Multi-Modal Robot Programming." *International Conf. on Intelligent Robots and Systems (IROS) 2003*, Las Vegas, NV.

Iba, S., Vande Weghe, J. M., Paredis, C. J. J., and Khosla, P. K. (1999). "An Architecture for Gesture-Based Control of Mobile Robots." *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Kyongju, Korea, 851-857.

Ikeuchi, K., and Suehiro, T. (1994). "Toward an Assembly Plan from Observation, Part I: Task Recognition with Polyhedral Objects." *IEEE Transactions Robotics and Automation*, 10(3), 368-385.

Ishida, T. (2003). "A small biped entertainment robot SDR-4X II." *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, Kobe, Japan, 1046-1051.

Jayaraman, R., and Deisenroth, M. P. (1987). "An interactive programming system for the IBM 7545 robot." *Computers & Industrial Engineering*, 12(4), 275-82.

Kamakura, N. (1989). *Te no katachi, te no ugoki*, Ishiyaku Shuppan Kabushiki Kaisha, Tokyo.

Kang, S. B., and Ikeuchi, K. (1997). "Toward automatic robot instruction from perception-mapping human grasps to manipulator grasps." *IEEE Transactions on Robotics and Automation*, 13(1), 81-95.

Kawamura, K., Alford, A., Hambuchen, K., and Wilkes, M. (2000). "Towards a Unified Framework for Human-Humanoid Interaction." *First IEEE-RAS International Conference on Humanoid Robots*, Boston, MA.

Kimura, H., Horiuchi, T., and Ikeuchi, K. (1999). "Task-Model Based Human Robot Cooperation Using Vision." *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Kyongju, Korea, 701-706.

Kortenkamp, D., Bonasso, R. P., and Subramanian, D. (2001). "Distributed, Autonomous Control of Space Habitats." *IEEE Aerospace Conference*, Piscataway, NJ, USA, 2751-62.

Kortenkamp, D., Huber, E., and Bonasso, R. P. (1996). "Recognizing and interpreting gestures on a mobile robot." *National Conference on Artificial Intelligence*, Portland, OR, USA, 915-21.

Koyama, T. (2002). "On Combining Sampled Statistics without Prior Samples," Personal Communication to Author, Pittsburgh.

Kuno, Y., Murashima, T., Shimada, N., and Shirai, Y. (2000). "Interactive gesture interface for intelligent wheelchairs." *International Conference on Multimedia and Expo*, New York, NY, USA, 789-92.

Lee, C., and Xu., Y. (1996). "Online, Interactive Learning of Gestures for Human/Robot Interfaces." *IEEE International Conference on Robotics and Automation*, Minneapolis, MN, 2982-2987.

Mardia, K. V. (1972). *Statistics of directional data*, Academic Press, London and New York.

Mascaro, S., and Asada, H. H. (1998). "Hand-in-glove human-machine interface and interactive control: task process modeling using dual Petri nets." *IEEE International Conference on Robotics and Automation*, Leuven, Belgium, 1289-95.

Matsumoto, Y., and Zelinsky, A. (2000). "An algorithm for real-time stereo vision implementation of head pose and gaze direction measurement." *Fourth International Conference on Automatic Face and Gesture Recognition*, Grenoble, France, 499-504.

Morrow, J. D., and Khosla, P. K. (1997). "Manipulation task primitives for composing robot skills." *IEEE International Conference on Robotics and Automation*, Albuquerque, NM, USA, 3354-9.

Musser, G. (2003). "Robots That Suck." *Scientific American*, 288(2), 84-6.

Nagchaudhuri, A., Singh, G., Kaur, M., and George, S. (2002). "LEGO robotics products boost student creativity in precollege programs at UMES." *32nd Annual Frontiers in Education*, Piscataway, NJ, S4D-1-6.

Nakatani, M., Suzuki, K., and Hashimoto, S. (2003). "Subjective-Evaluation Oriented Teaching Scheme for a Biped Humanoid Robot." *IEEE-RAS International Conference on Humanoid Robots (Humanoids2003)*, Karlsruhe and Munich, Germany.

Nishimura, T., Mukai, T., Nozaki, S., and Oka, R. (1998). "Adaptation to gesture performers by an on-line teaching system for spotting recognition of gestures from a time- varying image." *Transactions of the Institute of Electronics, Information and Communication Engineers D-II*, J81D-II(8), 1822-30.

Ogawara, K., Takamatsu, J., Iba, S., Tanuki, T., Kimura, H., and Ikeuchi, K. (2000). "Acquiring hand-action models in task and behavior levels by a learning robot through observing human demonstrations." *IEEE-RAS International Conference on Humanoid Robots*, Boston.

Onda, H., Ogasawara, T., Hirukawa, H., Kitagaki, K., Nakamura, A., and Tsukune, H. (2000). "A Telerobotics System using Planning Functions Based on Manipulation Skills and Teaching-by-Demonstration Technique in VR." *Journal of the Robotics Society of Japan*, 18(7), 979-994.

Oviatt, S. (2000). "Taming recognition errors with a multimodal interface." *Communications of the ACM*, 43(9), 45-51.

Pavlovic, V. I., Sharma, R., and Huang, T. S. (1997). "Visual interpretation of hand gestures for human-computer interaction: a review." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7), 677-95.

Perzanowski, D., Schultz, A. C., Adams, W., Bugajska, M., Marsh, E., Trafton, J. G., Skubic, M., and Abramson, M. (2002). "Communicating with Teams of Cooperative Robots." Multi-Robot Systems: From Swarms to Intelligent Automata, A. C. Schultz and L. E. Parker, eds., Kluwer, The Netherlands, 185-193.

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

Quek, F. (1994). "Toward a Vision-Based Hand Gesture Interface." *Virtual Reality System Technology Conference*, Singapore, 17-29.

Rabiner, L. R. (1989). "A tutorial on hidden Markov models and selected applications in speech recognition." *Proceedings of the IEEE*, 77(2), 257-86.

Rao, R. S., Conn, K., Jung, S. H., Katupitiya, J., Kientz, T., Kumar, V., Ostrowski, J., Patel, S., and Taylor, C. J. (2002). "Human Robot Interaction: Application to Smart Wheelchair." *IEEE International Conference on Robotics and Automation*, Washington, DC.

Rapid Reference Manual 3.0 (1994). ABB Flexible Automation AB.

Rogalla, O., Ehrenmann, M., Zollner, R., Becher, R., and Dillmann, R. (2002). "Using gesture and speech control for commanding a robot assistant." *IEEE International Workshop on Robot and Human Interactive Communication*, Piscataway, NJ, USA, 454-459.

Rybski, P. E., and Voyles, R. M. (1999). "Interactive task training of a mobile robot through human gesture recognition." *IEEE International Conference on Robotics and Automation*, Detroit, MI, USA, 664-9.

Schofield, M. (1999). ""Neither master nor slave...". A practical case study in the development and employment of cleaning robots." *IEEE International Conference on Emerging Technologies and Factory Automation*, Barcelona, Spain, 1427-1434.

Shastri, S. V., and Iberall, T. (1990). *Dextrous robot hands*, Springer-Verlag, New York.

Skubic, M., Perzanowski, D., Schultz, A., and Adams, W. (2002). "Using Spatial Language in a Human-Robot Dialog." *International Conference on Robotics and Automation*, Washington, DC, 4143-4148.

Stallman, R. M. (1984). "Emacs: The extensible, customizable, selfdocumenting display editor." Interactive programming environments, D. R. Barstow, H. E. Shrobe, and E. Sandewall, eds., McGraw-Hill, New York, 300-325.

Starner, T., and Pentland, A. (1995). "Real-time American Sign Language recognition from video." *IEEE International Symposium on Computer Vision*, 265-270.

Thrun, S., Bennewitz, M., Burgard, W., Cremers, A. B., Dellaert, F., Fox, D., Hahnel, D., Rosenberg, C., Roy, N., Schulte, J., and Schulz, D. (1999). "MINERVA: a second-generation museum tour-guide robot." *IEEE International Conference on Robotics and Automation*, Piscataway, NJ, USA, 1999-2005.

Todd, D. J. (1986). *Fundamentals of robot technology: an introduction to industrial robots, teleoperators, and robot vehicles*, Wiley, New York.

Voyles, R. M., Agah, A., Khosla, P. K., and Bekey, G. A. (1997). "Tropism-Based Cognition for the Interpretation of Context-Dependent Gestures." *IEEE International Conference on Robotics and Automation*, Albuquerque, NM, USA, 3481-6.

Voyles, R. M., Morrow, J. D., and Khosla, P. K. (1999). "Gesture-based programming for robotics: human-augmented software adaptation." *IEEE Intelligent Systems*, 14(6), 22-31.

Waldherr, S., Romero, R., and Thrun, S. (2000). "A gesture based interface for human-robot interaction." *Autonomous Robots*, 9(2), 151-73.

Wren, C. R., Clarkson, B. P., and Pentland, A. P. (2000). "Understanding purposeful human motion." *Fourth IEEE International Conference on Automatic Face and Gesture Recognition*, 378 -383.

Yamada, Y., Morizono, T., Umetani, Y., and Yamamoto, T. (2002). "Human error recovery for a human/robot parts conveyance system." *International Conf. on Robotics and Automation*, Washington, DC, USA, 2004-9.

Yamada, Y., Umetani, Y., Daitoh, H., and Sakai, T. (1999). "Construction of a human/robot coexistence system based on a model of human will-intention and desire." *IEEE International Conference on Robotics and Automation*, Detroit, MI, USA, 2861-2867.

Yang, M.-H., and Ahuja, N. (2001). *Face detection and gesture recognition for human-computer interaction*, Kluwer Academic, Boston.

Yang, M.-H., Ahuja, N., and Tabb, M. (2002). "Extraction of 2D motion trajectories and its application to hand gesture recognition." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8), 1061-74.

Yared, W. I., and Sheridan, T. B. (1991). "Plan recognition and generalization in command languages with application to telerobotics." *IEEE Transactions on Systems, Man and Cybernetics*, 21(2), 327-38.

Young, S. J., Kershaw, D., Odell, J., Ollason, D., Valtchev, V., and Woodland, P. (2000). *HTK: Hidden Markov Model Toolkit V3.0*, Microsoft Corporation, Redmond, Washington, USA.

Young, S. J., Russell, N. H., and Thornton, J. H. S. (1989). "Token Passing: A Simple Conceptual Model for Connected Speech Recognition Systems." Cambridge University Engineering Dept.