Proceedings of the 2003 IEEE/RSJ
Intl. Conference on Intelligent Robots and Systems
Las Vegas, Nevada · October 2003

Intention Aware Interactive Multi-Modal Robot Programming

Soshi Iba[1], Christiaan J.J. Paredis[3], and Pradeep K. Khosla[1, 2]
[1)] *The Robotics Institute, Carnegie Mellon University*
[2)] *Electrical and Computer Engineering, Carnegie Mellon University*
Pittsburgh, Pennsylvania 15213-3890
[3)] *G. W. Woodruff School of Mechanical Engineering, Georgia Institute of Technology*
Atlanta, Georgia 30332-0405
iba@ri.cmu.edu, chris.paredis@me.gatech.edu, pkk@ece.cmu.edu

Abstract

As robots enter the human environment, there are increasing needs for novice users to be able to program robots with ease. A successful robot programming system should be intuitive, interactive, and intention aware. Intuitiveness refers to the use of intuitive user interfaces such as speech and hand gestures. Interactivity refers to the system's ability to let the user interact preemptively with the robot to take its control at any given time. Intention awareness refers to the system's ability to recognize and adapt to user intent.

This paper focuses on the intention awareness problem for interactive multi-modal robot programming system. In our framework, user intent takes on the form of a robot program, which in our context is a sequential set of commands with parameters. To solve the intention recognition and adaptation problem, the system converts robot programs into a set of Markov chains. The system can then deduce the most likely program the user intends to execute based on a given observation sequence. It then adapts this program based on additional interaction.

The system is implemented on a mobile vacuum cleaning robot with a user who is wearing sensor gloves, inductive position sensors, and a microphone.

1. Introduction and Related Work

Human-Robot interaction is an important aspect of a successful robotic system. As robots enter the human environment and come in contact with inexperienced users, they need to be able to interact with these users in an intuitive and interactive fashion. An interactive multi-modal robot programming system by Iba et al [1] demonstrated interactive sequential robot programming using voice commands and hand gestures on a mobile vacuum cleaning robot (Figure 1). The key elements behind this novice-friendly system are intuitive interfaces based on speech and hand gesture recognition, and interaction capabilities that allow the user to take over the control of the robot at any given time. Such interaction capabilities give a sense of assurance to the user and help him in dealing with loosely calibrated position sensors by including a human in the control loop using a combination of voice and hand-gesture commands. The user is 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 letting the user take control at any given time. The multi-modal human-robot interaction described above can be thought of as a parallel to WYSIWYG (*what you see is what you get*) interface introduced in the human-computer interaction domain. Instead of off-line robot programming, this method lets the user see what to expect from the program execution.

Intention awareness is an additional step towards a novice friendly robot programming system. Intent is the purpose or goal the user has in mind. An intention aware system can be used to reduce unnecessary and often redundant instructions by being aware of what the user really wants. The term intent is often loosely defined since it is heavily 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 be able to tell if such robot program exists in the system's database from inputs given by the user.

An intention aware system should be able to map user input to an intended robot action, if one exists, and adapt such a mappings based on additional input from interactions. Accordingly, intention awareness for interactive multi-modal robot programming can be divided into two sub-problems: intention recognition and intention adaptation. We approach the intention recognition problem by representing robot programs as a set of Markov Decision Processes (MDP). We build Hidden Markov Models (HMM) based on the MDPs using observations collected during the programming phase. A probabilistic representation of a robot program can be used to better represent the realistic model of a robot program. Such representations allow us to search

0-7803-7860-1/03/$17.00 © 2003 IEEE

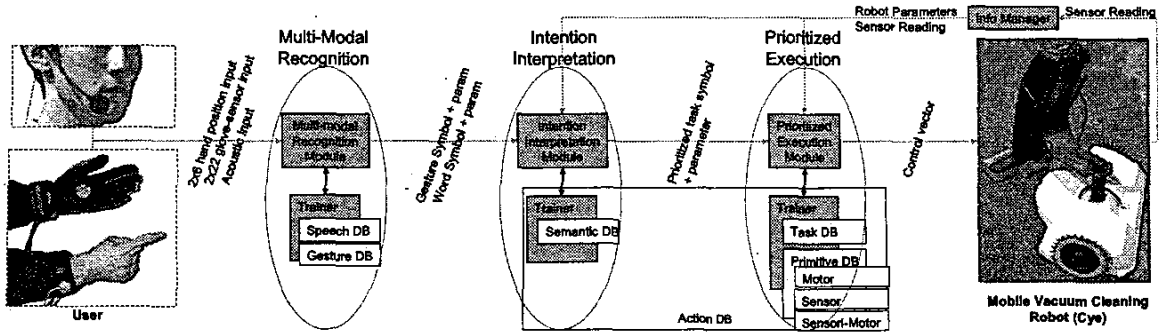

Figure 1: Framework for Interactive Multi-Modal Robot Programming

efficiently for the most likely intended action based on a sequence of user inputs, as well as recognizing the possibility that the user intends to perform no action. Probabilistic frameworks have been applied in the past to model intentions, for example, in the form of HMMs [2] or stochastic Petri-nets [3]. This research successfully demonstrated intention recognition capabilities. However, if one claims to be truly intention aware, it is essential to have a mechanism to adapt and modify intention models according to new situations. Without such a mechanism, the system can only work in the way its designer intended. We approach the intention adaptation problem by giving the user a choice to simply add more robot programs, or to adapt existing program to the current situation by merging additional observations into the probabilistic representation. To reflect the most current model of intended actions, all observations of the program execution and all interactive corrections made at the time of execution are incorporated into the program representation.

We describe the overall system and implementation in section 2, then describe the intention recognition and adaptation scheme in section 3, followed by the experimental result in section 4 and the conclusion in section 5.

2. System Description

The interactive robot programming system introduced by Iba et al. [1] offers the user, through an intuitive interface, the ability to provide interactive feedback to the robot to coach it throughout the programming and execution phase. Intention awareness is introduced in this paper to make the system even more novice-friendly.

The framework is composed of three functional modules as illustrated in Figure 1. 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 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, or composition of robot programs.

The first component, multi-modal recognition (the first block in Figure 1), initially interprets raw data into a stream of labeled data with parameters. In our system, it translates 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. In our implementation, spontaneous speech is translated into words using *SPHINX-II*, an off-the-shelf speech recognition package [4]. For hand gestures, we implemented a word spotting technique using the Hidden Markov Toolkit (*HTK*) [5]. [1] lists some of the initial candidate gestures and words that such a basic vocabulary could include.

The second component, intention interpretation module (the second block in Figure 1), generates a stream of prioritized action commands, given a sequence of labeled recognition results by the first component. The problem of intention interpretation can be considered as a mapping problem from the stream of user input, 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 output is a prioritized task symbol representing a configuration of robot primitives, where a primitive is an encapsulation of a low level robot behavior. The primitives used for the current implementation are described in [1]. The task is a robot program composed of

one or more primitives. The current implementation only supports sequential execution of primitives. Conditional statements will be integrated in future work. The intention interpretation module is implemented as a look-up table that connects recognition results to action commands.

The third component, prioritized task execution module (the third block in Figure 1) 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. For the arbitration, the task (a set of primitives) with the highest priority level is executed first. It can be interrupted by a task with equal or higher priority, if requested by the user. If possible, the system goes back to the interrupted task after the high priority task finishes. Such an interrupt scheme allows the user to stay in the control loop, and enables novice users to get acquainted with the system without feeling overwhelmed by the large number of unknowns in the system.

Generation of a robot program (task) is performed interactively. The basic approach is to take a coaching strategy using a redundant input mode. The user sets the module to a learning mode (by giving voice command "program one", for example) and executes primitives sequentially through hand gestures and voice commands; the system remembers the sequence as a task.

The next section goes over the implementation of intention awareness, which is closely associated with the intention interpretation module.

3. Intention Awareness

The system's intention awareness is composed of two capabilities: intention recognition and adaptation. Instead of merely mapping the sequence of multi-modal recognition results to the set of actions using the semantics database, the intention aware system should suggest which task (set of primitives) the user may want to execute based on an incomplete sequence of primitives executed by the user. This recognition ability can be thought of as similar to the auto-completion capability in a text editing program. It is especially helpful when there are a large number of programs, and picking any particular program may be difficult. The ability to perform online modification is also 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 registering the interrupts as additional primitives in the task.

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 P_i's describe robot positions in terms of (x, y). A Hidden Markov Model (HMM) [6] provides a way to model the task in a probabilistic framework, where both state transitions and observations can be expressed stochastically. 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 as HMMs are organized and compared to the current observation sequence to detect which task, if any, the user may want to execute.

Other systems such as the human intention recognition by Yamada *et al.* [2] and the online point-based hand-writing recognition by Bahlmann [7] employ similar strategies. Our method for representing and recognizing tasks from an observation sequence using HMMs is similar to their work. However, our method has the advantage that it is capable of disregarding non-task sequences through dynamic garbage collection, without the prior training of a garbage model.

Adaptation of models is necessary to account for changes in the robot's environment, or modifications to the 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 tasks are represented as HMMs, how they are constructed, and how they can be modified in real-time during execution.

3.1. Construction

When a robot is programmed interactively, the system collects an observation sequence $O_n = \{o_{0n} o_{1n} \ldots o_{tn}\}$ for program action n, where $o_t = \{x_t, y_t\}$ correspond to the robot position at time t. The sequence O is the collection of observations O_n resulting from program actions 1 to N (0 is the start). The robot program is then converted into an HMM through the process described in Figure 2. The program (top of Figure 2) is first converted into a Markov chain description whose states correspond to each programmed action. The number of states in the chain is the number of actions in the program plus two (start and the end). The collected set of observations is used to calculate necessary statistics to describe observations coming out of the arcs. The observation sequence collected during the action associated with the state is used to construct the observation density function $b_{ii}(o)$

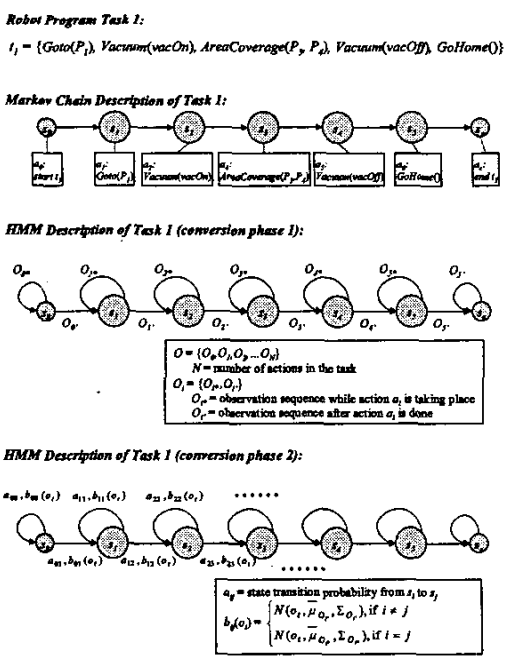

Figure 2: Conversion of a program to HMM

on the recurring arc, whereas observation sequence collected after the action but before the beginning of the next action is used to construct $b_{ij}(o)$ for the transition arc. The state transition probability a_{ij} is determined by the ratio of number of observations used at recurring and transition arcs.

3.2. Recognition

During recognition, the current observation sequence is evaluated and compared to all robot program HMMs stored in the database. It is necessary to be able to detect in real-time which program the user may be interested in, and to be able to reject observations that are not part of any existing program. Our goal here is to find the most likely state q_t at the current time t, given observations up to time t, and HMMs λ_1, λ_2 ... λ_U, constructed from U robot programs organized into λ_{net} as described in Figure 3. It should be noted that the transition probabilities a_{*0} on the arcs from the initial shared state s_{00} are fixed to $1/(1+U)$, regardless of the observation. This initial shared state collects observations that do not match any program.

In order to find the single best state q_t out of all states in the shared HMM network, we consider the variable $\delta_t(u,i)$ for program u, state i, for the Viterbi Algorithm extended from the definition in Rabiner's tutorial [6]:

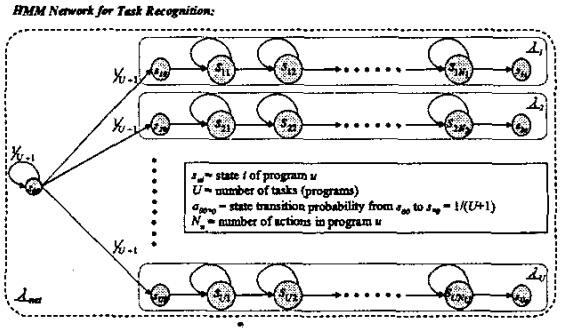

Figure 3: HMM Network with Shared Initial State

$$\delta_t(u,i) = \max_{q_1 \ldots q_{t-1}} P(q_1, \ldots q_t = s_{u,i}, o_1, \ldots o_t \mid \lambda_{net}) \qquad (1)$$

Since we are only interested in the most likely state, we only need to keep the HMM trellis of δ_t scores to apply the algorithm when the new observation shows up. The score (probability) in each trellis entry describes 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 δ_t, the entire trellis is normalized to 1.0 for every observation. The initial shared state becomes the most likely state if the observation sequence can not be explained by other models $\lambda_1..\lambda_U$. After finding the current HMM node, the system can determine the action that should be taken according to the most probable robot program.

3.3. Adaptation

Online seamless adjustment of the statistics that describe the robot program is 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, and the program description needs to be adjusted accordingly. The parameter adaptation can be used to improve HMM parameters over multiple executions of the same task. This can be done by first partitioning the observation sequence and merging statistics derived from additional samples with the old ones. Merging n_{add} additional samples with mean vector, μ_{add}, and covariance matrix, Σ_{add}, with n_{old} old samples with μ_{old}, Σ_{old} to derive new combined statistics, n_{new}, μ_{new}, Σ_{new} can be done as follows:

$$n_{new} = n_{add} + n_{old} \qquad (2)$$

$$\mu_{new} = (n_{add}\mu_{add} + n_{old}\mu_{old}) / n_{new} \qquad (3)$$

$$\Sigma_{new} = (A + B + C + D) / (n_{new} - 1) \qquad (4)$$

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

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

4. Experiment

The system's intention awareness was tested on the Cye personal robot [8] with vacuum cleaner against three robot programs in the database. Because observation sequences generated from an interactive user control are not reliable as a basis of comparison against HMM representation of a robot program, we used artificially generated observation sequences instead of direct user interaction to address issues such as intention recognition, and model adaptation. Three test programs used are:

t_1 = {Goto(P_1), Vacuum(vacOn), Goto(P_2), AreaCoverage(P_3, P_4), Vacuum(vacOff), GoHome()}

t_2 = {Vacuum(vacOn), Goto(S_1), Goto(S_2), AreaCoverage(S_3, S_4), GoHome()}

t_3 = {Goto(T_1), Goto(T_2), Goto(T_3)}

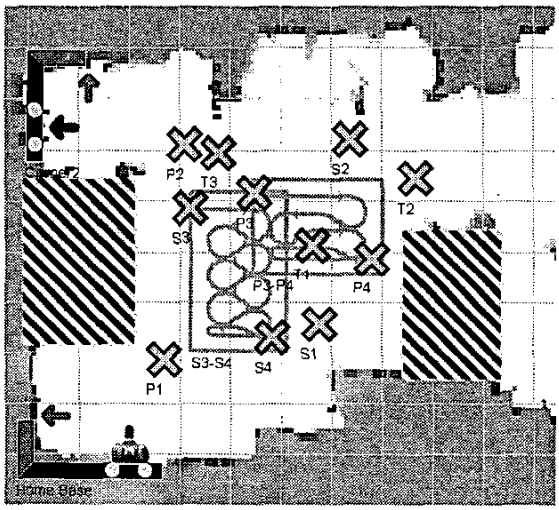

Figure 4: Positions used for the test programs t_1, t_2, and t_3,

where P_i, S_i, T_i all represent positions on the map in (x, y), as described on Figure 4.

HMM representations, λ_1, λ_2, and λ_3, for the test programs, t_1, t_2, and t_3 are created through the method described in section 3.1, after single execution of the program to collect observation sequence. Observations were collected in ~5Hz. The resulting HMM network representation, λ_{net}, is similar to that of Figure 3, but with three programs with different number of states. Their observation probability distributions are described on Figure 5.

Recognition was performed on the constructed HMM network using an artificial traversal data that travels in the order of {*Home*, P_2, P_1, T_1, T_2, *Home*}. Figure 6 describes the probability response of the states that correspond to s_{00}, the initial shared state, s_{31} and s_{32} that correspond to 1st and 2nd states in the third program, t_3. The system recognized that the most likely state while traversing

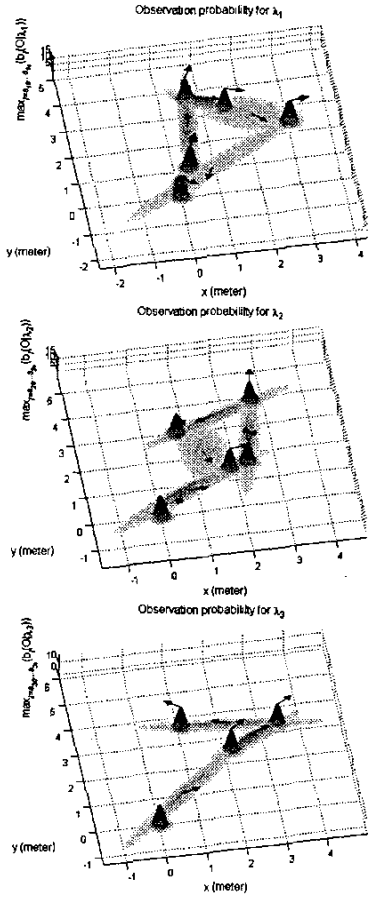

Figure 5: Observation probability distributions for the HMMs of the test programs λ_1, λ_2, λ_3

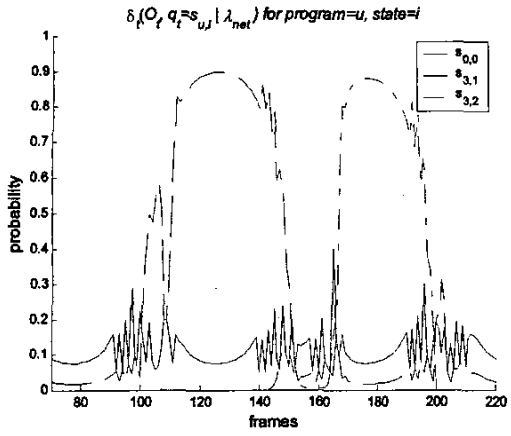

Figure 6: Probability Response from three states: s_{00} (initial shared state), s_{31}, s_{32}

through P_1, T_1 and T_2 were s_{31} and s_{32}. The observations used to model the probability on s_{31}'s recurring arc came from the path in between *Home* and T_1 that P_1 happens to be close to. Also, it should be noted that in between those regions with definite decisions, there exist a reading where the system seem bit confused. Those regions are where s_{00}, the initial shared state with solid line on the graph can tell you that there are no conclusive winner and thus the system should give no decision.

Adaptation of HMM statistics were performed for additional reliability using traversal data that matched closely to the ones in HMM network. Since the traversal data was artificial, the statistics didn't really change, but this feature should be a useful tool when it comes to the real world, where the path between two points may not be straight at all.

5. Conclusion and Future Work

Human-Robot interaction needs to be intuitive, interactive, and intention aware. This paper presented the novel approach to make the interactive multi-modal robot-programming framework also intention aware. The keys to managing human-robot interactions are the ability to recognize in real-time the program the user intends to execute, and the ability to adapt interactively to changes. In this paper, 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.

At this point, robot programs are sequential and non-branching, which is a serious shortcoming if the user intends to use a robot with sensor-based autonomy. Interactive programming of conditional statements and their conversion to models that are easily recognizable and adaptable needs to be worked out. An interactive robot-programming system that makes useful suggestions would further improve the usability for novice robot users.

6. Acknowledgements

This research was funded in part by DARPA under contract DAAD19-02-1-0389 and ABB under contract 1010068. Additional support was provided by the Robotics Institute at Carnegie Mellon University.

7. References

[1] 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.

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

[3] Manabe, Y., Hattori, M., Tadokoro, S., and Taiamori, T., "Generation of home robots movement based on prediction of human actions (A model of human actions by a Petri net and prediction of human acts)," *Symposium on Robotics and Cybernetics. CESA '96 IMACS Multiconference*, pp. 210-15, 1996.

[4] Huang, X., Alleva, F., Hon, H.-W., Hwang, M.-Y., and Rosenfeld, R., "The SPHINX-II speech recognition system: an overview," *Computer Speech and Language*, vol. 7, no. 2, pp. 137-48, 1993.

[5] Young, S., Kershaw, D., Odell, J., Ollason, D., Valtchev, V., and Woodland, P., *The HTK Book (Version 3.0)*. Redmond, Washington, USA: Microsoft Corporation, 2000.

[6] Rabiner, L. R., "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257-86, 1989.

[7] Bahlmann, C. and Burkhardt, H., "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, pp. 406-11, 2001.

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