16.412 Pset #1
Shuonan Dong
2/15/05

Part A: Topics of Fascination

1. Probabilistic reasoning

When an autonomous agent is interacting with the real world, it is often the case that the environment is non-deterministic. This could occur if the agent is dealing with a dynamic environment, or if the agent needs to interact with human beings. In such situations, the agent may need probabilistic reasoning techniques to model the environment and make decisions. If the agent can make observations over time, then techniques such as hidden Markov models, Kalman filters, and dynamic Bayesian networks could be useful to make future predictions. Probabilistic reasoning techniques could be useful in applications such as speech recognition.

2. Machine learning with incomplete data

One possible attribute of an intelligent system is the capability to learn to make correct decisions based on available and possibly incomplete training data. It may require learning probability models such as Bayesian networks and making maximum probable hypotheses. Algorithms such as Markov chain Monte Carlo and expectation-maximization, and models such as Kernel models and neural networks may be useful.

3. Communicating, perceiving, and acting

A robot that needs to interact with the environment must have a variety of methods of sensing and communication. Language recognition, tactile sensing, and visual image processing become critical for an interactive robot. In visual perception, object recognition, edge detection, contour mapping, and motion detection would be useful to study.

Part B: Favored Cognitive Robot

Current spacesuit development has been focused largely on the mechanical design, and little work has been done with respect to information interfacing with the astronaut. As a far-stretched idea, it may be useful for an astronaut to have immediate computer access within the spacesuit, where the information is projected onto the suit helmet. The controls to such a device could be voice-activated, or controlled by motion in the rest of the spacesuit. Imagine if an astronaut needed to get to a target ground feature that is not in plain view. S/he could pull up an archived aerial map of the area on the display, select an area on the display by simply moving his/her arm and finger in a circular motion, and the computer would be able to acknowledge the selected feature and have the map generate an optimal safest path to the feature. The astronaut can then follow the path in real time toward the destination.

In this complex system, I would like to focus on the motion control interface. If one has seen the Hollywood film Minority Report, or played video games with similar motion controls, then one has a good idea of how this interface is desired to work. Essentially the goal is that an astronaut can give commands by making some kind of gesture, since other forms of inputs (such as a keyboard) can be cumbersome or difficult to implement.

The spacesuit would need to sense the motion of the user, filter the important input signals, recognize the command, and respond accordingly. To implement these functions, the system needs a motion sensing system to gather the input data, probabilistic reasoning techniques to filter the data, and pattern recognition capabilities to actually use the data. Figure 1 shows a flow diagram of how a user command is detected.

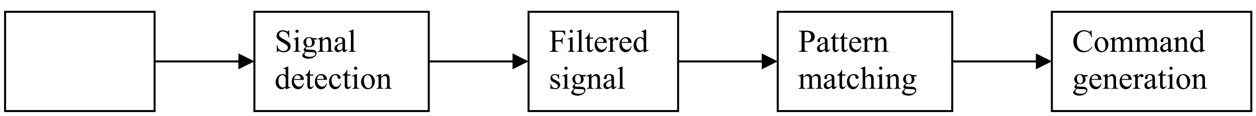

Figure 1. Motion controlled interface breakdown

Part C: Important Form of Reasoning

It seems that a difficult aspect of pattern recognition in the case mentioned in Part B is the uncertainty in the pattern of motion that the agent must sort through. Once the agent decides on a set of possible patterns that a person is trying to convey, then the actual pattern matching step only deals with deterministic patterns and should be much easier. Therefore the machine's ability for probabilistic reasoning should be the first issue to be addressed.

Part D: Researching a Critical Reasoning Method

On the topic of probabilistic reasoning, I have found a myriad of papers about Bayesian networks and various aspects of Markov Decision Processes. Having little initial knowledge on the relevance of these papers on my project goal, the following three papers were somewhat chosen at random from the Uncertainty in Artificial Intelligence Conference Proceedings.

1. Myers, J. W., Laskey, K. B., and Levitt, T. S., *Learning Bayesian Networks from Incomplete Data with Stochastic Search Algorithms*

Current research in Bayesian networks generally assume complete data so that the Bayesian network problem can be reduced to learning the graph structure and parameters. However, problems arise when there is incomplete data because there are no closed form expressions for the scoring metric used to evaluate network structures. Some researchers have taken parametric approaches such as the expectation-maximizatino (EM) algorithm, but this method is prone to "getting stuck" on local optima.

Myers et al. propose a stochastic search method to avoid the local optima problem. They use a combination of evolutionary algorithms (EA) and Markov chain Monte Carlo (MCMC) algorithms to create a new algorithm called the Evolutionary Markov Chain Monte Carlo (EMCMC) in order to learn Bayesian networks from incomplete data.

The evolutionary algorithm is similar to the evolution processes that occur in nature. Evolutionary algorithms consist of "a population of individual solutions that are selected and modified in order to discover overall better solutions in the search space."

Initially, a population of solutions are selected, from which individuals are chosen based on how good a solution it provides. These individuals are genetically modified using the common genetic operators crossover and mutation. Thus evolutionary algorithms deal with incomplete data by "turning the incomplete data problem into a complete data problem by evolving the missing data and imputing these values into the data."

The Markov chain Monte Carlo algorithm is often used to model systems that seek a minimal free energy state. The microstate of a system, or the state of a system's atomic structure, has an associated energy, E(s). For a given microstate of a system, the equilibrium probability that minimizes free energy is given by the Boltzman distribution. Markov chains are useful because they can be designed to converge to a stationary distribution (such as Boltzman distribution), and from that point the convergent distribution is the only one necessary to predict occurrences in the future.

Evolutionary algorithms "sample from modes of the distribution of interest." "Markov chain Monte Carlo algorithms evolve to sample from the target stationary distribution." Since EA improve fitness through exchanging information, "the same approach with MCMC will speed convergence by finding better fit solutions faster." Empirical results show that the MCMC with adaptive mutation has a much faster rate of convergence than regular MCMC algorithm.

2. Flores, M. J., Gámez, J.A, and Olesen, K G., *Incremental Compilation of Bayesian Networks*

Bayesian networks are usually compiled into a junction tree or a join tree (JT) before probability propagation is performed. The compilation process is usually performed as one unit, where the entire join tree is created from the Bayesian network. If the network gets modified, the whole tree needs to be recompiled. The authors of this paper have devised a method for incrementally compiling a Bayesian network so that it increases the efficiency and stability of the join tree.

This paper was chosen because it may be useful to have the capability to deal with a more dynamic Bayesian network in an efficient way. Since the states of the environment that I will be dealing with in my project goal may not be completely determinable at the beginning, the generated Bayesian network may need to be modified through time, and thus it may be necessary to incrementally compile the network to a join tree.

The paper provides algorithms for dealing with all the structural changes in a Bayesian network, including modification of the probability distribution for a variable, modification of the states of a variable, removing an arc, adding an arc, removing a node, and adding a node. It starts off by decomposing the Bayesian network into its maximum prime subgraphs (MPS), which contain minimal triangulation. Then these MPSs can be triangulated independently.

The paper provides a clear set of algorithms for dealing with the steps for incremental compilation. Modifying potentials and the states of a variable generally leave the structure of the network the same, and thus can be simply updated in the join tree. When removing or adding an arc to the network, large changes could be made due to cycles in the network. Thus "it is beneficial to be able to identify the minimal part of the join tree that could be affected by the change and concentrate on a retriangulation of only that part. The removal or addition of a node can be performed by first removing or adding all the necessary arcs around the node, and then simply removing or adding the node itself.

3. Dearden, R., Friedman, N., and Andre, D., *Model Based Bayesian Exploration*

Reinforcement learning deals with the learning process of an agent in a dynamic environment, where the agent has the choice of exploring an untested set of actions or choosing a previously used set of actions that are known to be good (i.e. exploitation). Markov Decision Processes (MDPs) are useful frameworks to support reinforcement learning by enabling us to determine the actions that would maximize the expected future reward.

It may be useful to create a model of the dynamics of the environment so that the expected values of actions in the environment can be easily determined. Modeling the environment can aid the efficiency of computation because it reduces repetition of approximating value functions that measure the desirability of each environment state.

The paper shows “how to maintain Bayesian belief states about MDPs” and “how to use the Bayesian belief state to choose actions in a way that balances exploration and exploitation. The paper was chosen because it provides a possible useful approach to reinforcement learning systems, which may be handy for my project.

Part E: A Simple Project

The complex system described in Part B can be toned down to a two-dimensional pattern matching problem. A user can make a motion using a mouse on a computer, and the goal of the agent is to learn this motion and correctly identify it as a certain command. The agent would need to filter out random motions of the mouse input to capture the important information. It then needs to build a library of the meanings of these motions, and after a period of training, be able to correctly identify the command.

This project is feasible and challenging enough that there is a relatively high probability that I will pursue this project during this course.