Mobile Robot Planning with Incomplete Information in Dynamic Environments

Brian Yamauchi

Institute for the Study of Learning and Expertise
2164 Staunton Court
Palo Alto, CA 94306

Center for the Study of Language and Information
Stanford University
Stanford, CA 94305

Email: yamauchi@robotics.stanford.edu
WWW: http://robotics.stanford.edu/people/yamauchi/bio.html

Background

If mobile robots are to be useful in real-world environments containing human beings, these robots will need to be able to deal with dynamic changes. This paper describes the problems of incomplete information raised by dynamic environments and outlines questions intended to open a dialogue between planning researchers studying the role of incomplete information and roboticists building mobile robot systems for dynamic, real-world environments. In addition, this paper provides an overview of one approach to this problem which makes use of representations and algorithms designed specifically for robot exploration and navigation in dynamic environments.

Dynamic environments introduce the problem of incomplete information in a number of different ways. First, the robot may have incomplete information about the future state of the environment, since it may be impossible to predict exactly what changes will occur and when they will occur. Second, the robot may have incomplete information about the present state of the environment, since changes may be have occurred in a different part of the environment since the time of the robot's last visit. As a result, any attempt to complete the robot's information about the world can only be successful for a limited period of time. In large environments, the robot's world model may always be incomplete, regardless of attempts to gain new information.

For example: consider a mobile robot navigating through an office environment. Open doors allow travel and closed doors obstruct travel, so a complete model would require the knowledge of whether each door is open or closed. However, in a real office building, people will constantly be opening and closing doors as they move from one location to another, so the robot will never be able to know the complete state of the world.

In the case of doors, one could conceivably use the robot's current knowledge to determine a consistent set of possible world states, but now consider the problem of the people themselves. The robot may be able to navigate around a single person standing at one side of a hallway, but the robot may be blocked by a small group of people standing in the hall, or a single person standing in a doorway. In addition, the behavior of those people is likely to be unpredictable. Some may step aside to let the robot pass, but others may not. Unlike the states of doors, the positions and behaviors of people are not easily discretizable.

Most mobile robot systems have only been tested in static environments (Payton 1991; Kuipers & Byun 1993; Nourbakhsh, Powers, & Birchfield 1993; Kortenkamp & Weymouth 1994), while those developed for dynamic environments have tended to emphasize reactivity over planning (Brooks 1986; Connell 1987; Arkin 1989). Mataric has developed a mobile robot system that combines the ability to plan paths toward a goal with the ability to react to transient changes (Mataric 1992). I have developed a system that combines navigation planning with the ability to react to transient changes and the ability to adapt the spatial representation to reflect lasting topological changes encountered in the world (Yamauchi & Beer 1996). In both Mataric's system and mine, the planning consists of finding paths within a map encoding topological and geometric spatial information, rather than the traditional forms of symbolic reasoning performed by logic-based planners.

A central issue that I feel this symposium should address is to what extent the theoretical issues related to planning with incomplete information are relevant to the actual development of mobile robots that can operate in dynamic, real-world environments. With this in mind, I propose the following questions for discussion.

Questions

A) Planning/Executing/Reacting with Incomplete Information

1. In dynamic environments, is it sensible to plan all of the robots actions, or do some actions need to be "hardwired" reactions based on immediate sensing, to compensate for the robot's incomplete model of the world?
2. In the latter case, how should control be arbitrated between the planning system and the reactive system?
3. What is the relationship between planning discrete actions and sequencing behaviors?
4. Is "executing a plan" the best way to think about using current information to shape the dynamics of future actions?
5. What is the relationship between interleaving planning and execution and continuously using planning to modulate reactive behavior?

B) Logical vs. Spatial Representations

1. Is there a role for abstract logical representations of objects and relationships between objects? How would such representations be used in a real-world environment?
2. What are the advantages and disadvantages of using representations designed to capture specific spatial information about the world (e.g. topological maps, occupancy grids)?
3. Is there any advantage to combining logical representations and specific spatial representations in hybrid systems?

C) Uncertainty

1. When faced with incomplete information, what are the tradeoffs between general solutions to the planning problem and specific strategies for dealing with the particular types of information that are likely to be missing?
2. How should uncertainty be represented within the robot's model of the world?
3. How can the robot detect when its model of the world is incorrect?
4. How frequently should the robot check the accuracy of its model? Should these checks be planned or automatic?

D) Dynamic Environments

1. In dynamic environments, what are the tradeoffs between general solutions to the planning problem and specific strategies aimed at dealing with the particular types of change the robot is likely to encounter?
2. What types of change is a mobile robot likely to encounter?
3. How does each type of change affect the information that is likely to be incomplete or outdated?
4. How can the dynamic nature of the environment be represented in the robot's model of the world?
5. How will the dynamic nature of the environment affect the ability of the robot to acquire new information about the world?
6. Can using behaviors instead of atomic actions make plan execution more robust to transient changes?
7. Should the robot have an explicit model of time and how the world is likely to change over time?

A Task-Oriented Approach: ELDEN

System Architecture

One approach to navigation planning in dynamic environments is demonstrated by ELDEN (Exploration and Learning in Dynamic ENvironments). ELDEN utilizes representations and algorithms that I have developed specifically for the problems of exploration and navigation in dynamic environments..

ELDEN has three major components: a low-level control system composed of reactive behaviors, an adaptive place network that learns the topology and geometry of the environment, and a relocalization subsystem that uses evidence grids to recalibrate the robot's dead reckoning.

Each of these subsystems is designed to handle a different type of environmental change. The reactive controller deals with transient changes, such as people walking in front of the robot, and guarantees that the robot will be able to operate even when its spatial representation does not accurately reflect the current state of the environment. The adaptive place network deals with topological changes and constantly adapts to reflect any new places in the environment that are encountered. The relocalization subsystem deals with one specific form of perceptual change, the slippage that accumulates in the wheel encoders and leads to errors in the robot's dead reckoning estimate of its position and orientation.

ELDEN operates in two different modes: exploration mode and navigation mode. During exploration, ELDEN's goal is to learn the spatial structure of the environment. During navigation, ELDEN's primary goal is to move to a particular location. However, ELDEN continues to learn during navigation, so that if the world has changed, ELDEN will be able to detect those changes and adapt its spatial knowledge to match the new world.

Reactive Controller

ELDEN's behaviors are functions that map sensory inputs to output functions over the space of possible actions. Each output function is a sum of scaled Gaussians with variable mean, width, and height. So, instead of simply specifying a single desired action, each behavior can specify a number of desirable actions along with the priority of that

action and the tolerance for deviations from those desired actions. Behavior arbitration consists of summing all the output functions and selecting the action corresponding to the maximum in the quantized output space.

Adaptive Place Network

The adaptive place network (APN) provides a spatial representation and learning system for ELDEN that contains both metric and topological information about the structure of the environment. The APN is composed of place units, each of which corresponds to a region of Cartesian space, and place links, which represent the relationship between adjacent place units. The APN is always learning, during both exploration and navigation.

Initially, the APN is empty. At each timestep, the distance between the robot's current location and the nearest place unit is computed. If this distance is below a threshold, this unit becomes the winning unit, corresponding to the robot's current place. Otherwise, a new unit is created, with the Cartesian coordinates of the robot's current location, and this becomes the winning unit.

A place link is created the first time that ELDEN moves from one place to another. Each link connects two places and stores the direction that the robot must travel to move from one place to the other, along with a confidence value indicating the robot's certainty that this link can be traversed.

During navigation, the location associated with a particular place unit is specified as the destination. A cost is associated with each link equal to the reciprocal of the link confidence value. Then, Dijkstra's algorithm is used to find the shortest path from each place to the destination, and an orientation is associated with each link indicating the direction of travel towards the destination. After applying this algorithm, a single outgoing link is on the shortest path to the destination from each place unit. Behaviors within the reactive controller then orient the robot toward the direction of this link and move the robot forward (while also avoiding any nearby obstacles).

A link from place A to place B is considered successfully traversed, during either exploration or navigation, if at time t, the APN indicates that the robot is at place A, and at time $t+1$, the APN indicates that the robot is at place B. Whenever a link is traversed successfully, the link confidence is increased.

A link from place A to place B is considered unsuccessfully traversed during navigation, if at time t, the APN indicates that the robot is at place A, and at time $t+1$, the APN indicates that the robot is at place other then A or B, or if the place network indicates that the robot is at an unknown location, or if the robot attempts to move from A to B for a certain period of time, and yet remains at A. These correspond to three different situations that ELDEN may encounter. Instead of navigating to the correct location, sensor noise or motor error may cause it to end up at a different location. Or, steering around an unexpected obstacle, ELDEN may find itself in a completely new location. Or, the pathway may be blocked, and ELDEN may be unable to make progress. In any of these cases, the link confidence is decreased.

Whenever the network changes, either through a change in link confidence levels or the addition of a new link or unit, the shortest paths toward the destination are redetermined. This allows ELDEN to continually adapt to new information that is acquired as it navigates. In particular, it allows ELDEN to deal with topological changes in the environment that may require substantially different paths toward the destination.

Results

Experiments in a real-world office environment have demonstrated that ELDEN can explore and navigate robustly in the presence of both transient and lasting changes. These experiments are described in detail in (Yamauchi & Beer 1996).

References

Arkin, R. 1989. "Motor schema-based mobile robot navigation," *International Journal of Robotics Research,* 8:92-112.

Brooks, R. 1986. "A robust layered control system for a mobile robot," *IEEE Journal of Robotics and Automation,* 2:14-23.

Connell, J. 1987. "Creature building using the subsumption architecture," *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-87),* 1124-1126.

Kortenkamp, D., and Weymouth, T. 1994. "Topological mapping for mobile robots using a combination of sonar and vision sensing," *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94).*

Kuipers, B., and Byun, Y. 1993. "A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations," *Journal of Robotics and Autonomous Systems,* 8:47-63.

Mataric, M. 1992. "Integration of representation into goal-driven behavior-based robots," *IEEE Transactions on Robotics and Automation,* 8:304-312.

Nourbakhsh, I.; Powers, R.; and Birchfield, S. 1995. "DERVISH: An office-navigating robot," *AI Magazine,* 16:53-60.

Payton, D. 1991. "Internalized plans: A representation for action resources," in *Designing Autonomous Agents,* Maes, P., ed., Cambridge, MA: MIT Press, 89-103.

Yamauchi, B., and Beer, R. 1996. "Spatial learning for navigation in dynamic environments," to appear in *IEEE Transactions on Systems, Man, and Cybernetics-Part B,* Special Issue on Learning Autonomous Robots, Vol. 26, No. 3.