AI and Robotics Labs at the Undergraduate Robotics Laboratory, St. Bonaventure University

Robert M. Harlan

Computer Science Department
St. Bonaventure University
rharlan@cs.sbu.edu
http://web.sbu.edu/cs/harlan/index.html

Abstract

We discuss six labs that present two contrasting approaches to AI robotics. The first four labs present the traditional, representation-based approach to designing behavior control algorithms for autonomous robots. Robots use an internal representation of the world plus a planning algorithm to guide behavior. The second two labs present a behavior-based or reactive approach to managing robot behavior. Behavior is tied directly to sensor readings, eliminating the need for an internal representation of the world or a centralized planner.

Introduction

We have developed a set of laboratory assignments to support an undergraduate robotics course, Robotics and Computer Vision that can, in whole or in part, be adopted for courses in robotics or artificial intelligence.

Influenced by Robin Murphy's *Introduction to AI Robotics* and Jean-Claude Latombe's *Robot Motion Planning*, the set is designed to present two of the three distinct approaches to designing behavior control algorithms for autonomous robots. Labs 1 – 4 introduce the traditional, representation-based approach. Following Latombe's statement of the basic problem of autonomous robot navigation, the robot is given a representation if its environment and uses the representation to guide its navigation.

Labs 6 and 7 focus on the reactive or behavior-based approach. Lab 6 introduces the approach by setting up the robot so that its behavior is shaped by its sensing of the environment rather than any representation of it. Lab 7 introduces a more interesting example of the reactive approach based upon a case study by Murphy[1].

There is at present no lab that introduces the third or hybrid approach. The approach is covered in lecture by readings from Murphy and others and by examining Scout, a topological map-following program developed at St. Bonaventure.

The course has two fifty-minute lectures and two two-hour labs per week. With the exception of the first lab, each lab is allocated two or three weeks.

The labs were developed using Khepera robots, a small, portable, self-contained robot from Switzerland that is ideal for computer science programs in liberal arts colleges. While the platform does not qualify as inexpensive (about $1,500 per copy), it is worth the investment in at least one Khepera robot. The robot comes "ready to use" out of the box and requires no soldering or any other skills alien to all but engineering students[2]. The platform has proven to be very stable, having survived two course iterations with no losses at all.

We have developed an object-oriented interface, the kRobot class, for the Khepera that we distribute free of charge. Students write their first behavior control algorithm in the first lab using the interface and implement pseudo-code designs using it from that point onwards.

Our purchase of the Khepera hardware and the development of the kRobot class were funded by National Science Foundation CCLI – AI grant 9980999. The interface, the labs, and reports on autonomous robot navigation systems developed at the Undergraduate Robotics Laboratory, are available at http://web.sbu.edu/cs/roboticsLab/index.html

The kRobot Class: An Interface for the Khepera Miniature Robot

The kRobot class was designed to provide an intuitive, object-oriented interface for developing and implementing behavior control algorithms for the Khepera robot.

The Khepera robot is a miniature mobile robot with similar capabilities to larger sized robots used in research. Its small size (55 mm diameter, 30 mm height), light weight (approx. 70 grams), and compact shape permit micro-world

[1] Lab 5 is a computer vision lab that really does not fit Murphy's taxonomy of approaches. It also requires a special vision turret for the Khepera. For these reasons it is not included in the discussion.

[2] We did reengineer the communication and power cable that permits experimentation in the tethered mode.

experimentation with control algorithms. It has sufficient sensors and actuators to support a wide variety of tasks. Its eight infrared sensors can sense both ambient light levels and proximity to nearby objects. It also has two DC motors that are capable of independent variable-speed motion, allowing the robot to move forwards, backwards, and to complete a variety of turns at different speeds (Figure 1).

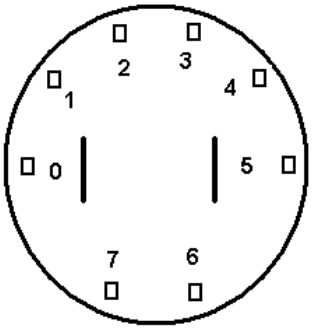

Figure 1: Schematic showing the Khepera's eight IR sensors and two effectors.

The most common mode for working with a Khepera is the tethered mode. A program to control the robot is developed and run on a host computer. Communication between the program and robot is done using a serial communications protocol called SerCom. Commands and requests for information are issued to the robot as strings; responses are sent back as strings.

The kRobot class hides this low-level communication and provides an intuitive interface for the Khepera robot that enables students to focus on developing behavior control algorithms. The class encapsulates state information about the robot with methods that enable programmers to access state information and to issue commands to the robot.

The interface has been used for several projects at the Bonaventure Robotics Lab and for two iterations of the robotics course. A simulator for the Khepera using the kRobot interface and a port to Java are planned in the near future.

The Traditional or Representation-Based Approached to Robotics

The first four lab assignments introduce the design and implementation of robot behavior control algorithms that rest on the robot's representation of its environment. The environment is represented by a two-dimensional grid where each position has a unique X, Y coordinate.

The environment was motivated by Latombe's formulation of the basis problem of robot motion planning: given a robot whose position and orientation is known, a set of obstacles the position of which are known, and a goal location, a robot must construct a path that takes it safely from its initial location to the goal location.

The grid we use is determined by the size of the Khepera robot and its wheels: the robot is 50mm in diameter, and each wheel rotation is 0.08mm. We selected a grid of 8 cm cells (1000 wheel rotations). Students construct the grid on paper, which permits easy evaluation of the robot's behavior in light of its internal representation of it position and its actual position. (Figure 2)

Lab 1 provides instructions for drawing the environment, connecting a Khepera robot to a host computer, and making and running a control program. The control program moves the robot north, south, east or west one or more cells at a time and demonstrates how the sensors can be monitored. The first part of the lab has students modify the program to generate the behavior of their choice (e.g., have the robot traverse a figure-8 pattern). The second part has them use the sensors to enable the robot to recognize and avoid obstacles.

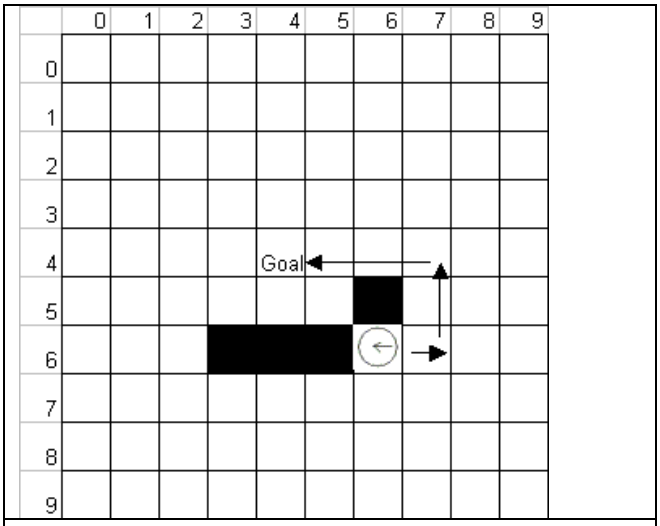

Figure 2: Sample grid used for navigation experimentation.

Lab 2 introduces motion planning. The robot is given its current position and orientation and a goal location in a world without obstacles. The robot must move from the current state to the goal state in a series of north/south/east/west motions. The lab provides a pseudo-code solution for generating the motion.

Lab 3 introduces obstacles that can block the motion of the robot. Obstacles each occupy one cell and may occupy adjacent cells. Students combine obstacle avoidance from the first lab with motion-planning from the second to build a behavior control algorithm that enables the robot to find a goal state given the presence of obstacles the location of which are unknown. The resulting algorithm implements an uninformed search for the goal that may or may not succeed.

Lab 4 guides students in the implementation of the A* algorithm for planning an optimum path from the current state to the goal state in a world where the location of obstacles is known. Students are given a pseudo-code version of the A* algorithm.

The Behavior-Based Approach to Robotics

The behavior-based approach to designing robot control algorithms addressed weaknesses in the traditional, representation-based approach. The traditional approach fails when the control program designer fails to anticipate situations in which the robot might find itself. The behavior-based approach holds that behavior should be motivated (stimulated) by the perceived environment without the intermediary of either a world model, through which the perceived environment is interpreted, or memory, which enables previous internal states to influence the current internal state of the robot. Perception is ``direct,'' and percepts of specific types give rise to corresponding behaviors without any intermediation.

Figure 3: Students experiment with corridor-following behavior.

Not all behavior is determined directly by percepts. Percept-behavior pairs can interact, resulting in what Rodney Brooks, a leading proponent of this approach, called emergent behavior, viz., behavior not programmed by the roboticist.

Lab 6 has students implement an example of subsumption discussed by Robin Murphy, which is based upon Brooks' work. In this example a higher-level behavior, wander, which periodically establishes a new direction for the robot to move, can subsume a lower-level behavior that enables the robot to avoid obstacles, resulting in novel responses to obstacles the robot encounters.

Students are directed to write an evaluateSensors() function that returns a Percept, an abstraction motivated by Murphy's discussion. In this case, the Percept indicates the position of an obstacle relative to the robot. Percepts, in turn trigger the correct avoidance behavior on part of the robot.

The lab provides pseudo-code for the lowest level avoidance behavior and provides guidance for adding the subsuming wander behavior.

Lab 7 guides students through a reactive-based case study discussed at length by Murphy (Chapter 5). Students are given pseudo-code guidance for implementing corridor-following behavior. The limitations of the IR distance sensors on the Khepera caused us to modify the case study to follow a wall on the right. However, the rapid speed at which the robot follows the corridor while avoiding contact contrasts very sharply with the relatively slow navigation resulting from the representation-based approaches used in the first four labs.

Conclusion

The six labs developed for the Khepera miniature robot provide an introduction to two distinct approaches to the design of behavior control algorithms for autonomous robots. The labs provide students with hands-on experimentation to each of the approaches, which sets the stage for a discussion of approaches that combine the reactive and traditional approaches.

The labs have been used in two offerings of our Robotics and Computer Vision course and will be used again in the spring of 2004.

References

Harlan, R., Levine, D. and McClarigan, S. 2001. The Khepera Robot and the kRobot Class: A Platform for Introducing Robotics in the Undergraduate Curriculum. 32nd SIGCSE Technical Symposium on Computer Science Education, pp. 105 – 109.

Harlan, R., Levine, D., Goodberry, J., Neel, M. and Zimmel, B. 2002. A Robust Mapper of Unknown Environments. http://web.sbu.edu/cs/roboticsLab.

Harlan, R., Cowles, M., and Casilio, J. 2003. Scout: Using Environmental Cues to Enhance Topological Map Following. http://web.sbu.edu/cs/roboticsLab

Latombe, J. 1991. Robot Motion Planning. Dordrecht, The Netherlands: Kluwer.

Mondada, F., Franzi, E., and Ienne, P. 1994. Mobile robot miniaturization: A tool for Investigation in Control Algorithms. Experimental Robotics III, Proceedings of the Third International Symposium on Experimental Robotics. Kyoto, Japan. Springer Verlag, pp. 501 -- 513.

Murphy, Robin. 2000. Introduction to AI Robotics. Boston: MIT Press.