Learning Action Models as Reactive Behaviors

Alan C. Schultz
John J. Grefenstette

Navy Center for Applied Research in Artificial Intelligence
Code 5514, Information Technology Division
Naval Research Laboratory, Washington, DC 20375-5000, U.S.A.
EMAIL: schultz@aic.nrl.navy.mil
(202) 767-2684

Abstract

Autonomous vehicles will require both projective planning and reactive components in order to perform robustly. Projective components are needed for long-term planning and replanning where explicit reasoning about future states is required. Reactive components allow the system to always have some action available in real-time, and themselves can exhibit robust behavior, but lack the ability to explicitly reason about future states over a long time period. This work addresses the problem of learning reactive components (normative action models) for autonomous vehicles from simulation models. Two main thrusts of our current work are described here. First, we wish to show that behaviors learned from simulation are useful in the actual physical system operating in the real world. Second, in order to scale the technique, we demonstrate how behaviors can be built up by first learning lower level behaviors, and then fixing these to use as base components of higher-level behaviors.

1. Introduction

Creating reactive behaviors is generally difficult, requiring the acquisition of knowledge from domain experts, a problem referred to as the knowledge acquisition bottleneck. SAMUEL is a system that learns reactive behaviors for autonomous agents (Grefenstette, Ramsey, and Schultz, 1990). SAMUEL learns these behaviors under simulation, automating the process of creating stimulus-response rules and therefore reducing the bottleneck. The learning algorithm was designed to learn useful behaviors from simulations of limited fidelity. The motivation for learning action models from simulation is pragmatic; learning from real systems can be expensive in terms of operating costs, time and in the possibility of loss of or damage to the vehicle.

One important assumption of this work is that, for many interesting tasks, behaviors learned under simulation can be successfully applied to real world environments. Previously, we demonstrated that behaviors learned in one simulation model would work in a different, target simulation model where known differences between the learning simulation and the target simulation were created. In this work, we show behaviors learned under simulation controlling an actual physical system.

Another important assumption is that these behaviors can be built up into more complex behaviors. While we have shown the ease of learning many behaviors for simulated autonomous agents (Grefenstette, Ramsey, and Schultz, 1990; Schultz, 1991; Schultz and Grefenstette, 1992), it is hard to learn all of the autonomous vehicle's behaviors at one time. What we propose is to incrementally build up complexity by learning more basic behavior models and then assuming and using these models as we learn higher level behaviors. This work focuses on *normative* action models, meaning that the system learns the most desirable actions associated with classes of situations. This can be contrasted with *descriptive* action models in which the system learns a description of state transitions associated with various actions.

The next section will describe the simulation and physical robots being used. Section 3 will describe the overall task that is being learned. Section 4 describes the current work in using simulation-learned behaviors on the actual robots. The work on building on existing behaviors in described in Section 5. Section 6 gives concluding remarks.

2. Description of Robots and Simulation

The two robots are each Nomadic Technologies, Inc.'s Nomad 200. The robot consists of a mobile base and a turret. The base is a three servo, three wheel synchronous

drive system with zero gyro-radius. That is, the three wheels rotate and translate together. One motor controls the translation of the three wheels, another motor controls the rotation of the wheels, and a third motor controls the rotation of the turret. The base contains two rings of tactile sensors. Each ring contains 10 touch sensors and the rings are rotationally offset from each other to allow better resolution. The turret contains three sensor systems: a sonar system, an infrared system, and a structured light range finder.

The sonar system is composed of 16 Polaroid sonars mounted in a ring around the turret. Each sonar has a width of 22.5 degrees and an effective range from 17 to 250 inches. The firing order and firing rate of the sonar cells is controllable. Also, individual sonar cells can be deactivated.

The active infrared system is composed of 16 active infrared sensors in a ring around the turret. Each is made up of two infrared emitters and one detector and has a 22.5 degree width. The system makes measurements before and during output from the emitters and compares the results to reduce the effect of ambient infrared radiation. Although the infrared sensors detect reflective intensity, an internal table attempts to correlate the intensity with distance, yielding an infrared ranging system. The system has an effective range of 1 to 24 inches.

The robot also has a structured light range finder (planar range finder). The system uses a laser diode as its light source and a CCD array camera for image generation and has an operating range of 18 to 120 inches. This sensor can produces an array of 60 range values.

Each robot uses a shared memory multiprocessor system. The master processor is a 50 Megahertz 60386 processor. The sensors are controlled by additional processors (Motorola MC68HC11F1 at 16 megahertz). In addition, the system uses a Motorola 68008/ASIC three axis motor controller. Three 12 volt batteries supply power to the robot, can which communicates back to a host computer using radio modems.

The robots are supplied with a robot control/communication server process, and associated programming libraries for creating user control programs. The robot can be controlled by using the supplied robot control routines in a user process that can run on a host computer. The user process then communicates with the robot server process which handles the low level communication with the robot. The user program can also be compiled to run directly on the robot. The robots also come with a medium fidelity simulation that can be used in place of the actual robot with no changes to the user's code.

In these experiments, the robot will be controlled from a host computer over the radio modems. Learning will be performed using the simulation, and results are tested on the actual robot.

3. Description of Task to be Learned

At the highest level, the robot must learn a tracking task. A target object wonders around the room. The robot must learn to keep that object within a given distance radius from itself. However, if the robot gets too close to the object, then the robot fails the task. The robot will use the planar light range finder to track the target object. The robot must learn to control the rotational rate of the turret to keep the target within its sensor range, and must determine the trajectory it wishes to follow for the next decision cycle. This trajectory is expressed as the x, y coordinate the robot wishes to go to during the next decision cycle.

In addition to learning to track an object, the environment contains obstacles that the robot must avoid as it performs the tracking task. This requires that the robot also learn a collision avoidance and local navigation behavior. If the bumpers detect a collision, the robot fails at the overall task. Here, the sonar and infrared sensors will be used to navigate and avoid obstacles. The robot must learn to control its translational and steering rates to avoid the obstacles and to advance towards the goal location.

4. Using Simulation-Learned Behaviors in Real World

Previously, we demonstrated that behaviors learned in simulation would work in a target environment, even with differences in noise between the learning environment and the target environment (Schultz, Ramsey and Grefenstette, 1990) In that study, we created two simulations that differed in only the aspects we were studying (noise and variability of initial conditions in the environment), and used them as the training and testing environments. However, many other differences tend to exist between simulation models and the systems they model, usually due to simplifying assumptions -- for example, discretization of the actual world in the model, differences in sensor behavior, temporal differences, and environmental differences.

This work extends our previous studies to actual physical devices, using laboratory robots to demonstrate the utility of learning reactive behaviors in a simulation model of limited fidelity. Our simulation model differs from the actual robots in several important aspects. In particular, communication and processing time lags are not well-modeled in the simulation, and certain types of noise in the

real world are not captured in the simulation model. Also, although the sensors in the simulation can model linear noise, they do not model the full non-linearities that exist on the real robot's sensors.

There is a design tradeoff and human effort tradeoff between creating a "perfect" simulation model, and the quality of the learned knowledge. More time can be spent designing a better simulation, but this in itself requires the codification of domain knowledge in the simulation. In essence you are building the action model into the simulation. The result is easier learning with simpler learning mechanisms. Conversely, you can put less effort into the simulation model, and design learning systems that can exploit this knowledge to create an internal action model more immune to differences between the learning environment and the real world.

This work will attempt to show that despite many differences between model and target, useful action models can be learned in the form of reactive rules for controlling the robot. The next section describes building up behaviors. The behaviors will be learned in simulation, but the performance of the systems will be measured on the physical robot.

5. Building on Existing Behaviors

Although SAMUEL has been used to learn behaviors for autonomous agents, as the overall complexity of the autonomous agent increases, it is clear that a complete behavioral model cannot be learned at one time. By carefully decomposing the task into component behaviors, separate behaviors can be learned and composed back into a complex system. Here, we start by learning basic needed behaviors using appropriate low-level sensors. These learned behaviors then become the action models for higher level behaviors. In general, the lower level behaviors use lower level sensors and control lower level effectors. Higher level behaviors can then make use of the lower level behaviors which abstract away the details of those lower level sensors and effectors. In this study we learn collision avoidance and navigation, and then tracking behaviors to solve the task.

Reactive rules for the behavior of collision avoidance and local navigation are first learned. The learned action model maps active infrared sensor data and sonar sensor data and internally integrated robot position and orientation data, plus a goal location for the robot into translation and turning rates for the robot. During this task the goal location, which is given to the robot as an x, y position, is fixed. The robot's starting location, as well as the number and location of obstacles is varied for each trial during learning.

After this behavior is learned, the tracking task is learned. Here, we learn reactive rules that map higher level sensors that indicate the location of the tracked object into a goal location for the tracking robot. The robot uses the earlier learned behavior for collision avoidance and local navigation to map the goal position of the robot into actual velocity mode commands (translational and rotational rates). For this task, a planar light range finder is used to detect the object being tracked.

At this level, the robot determines the location of the tracked object using the planar range finder, and then using this information, and its own integrated position and orientation data, must produce a goal trajectory for itself. This trajectory is expressed as a goal location for the current decision cycle. This goal location is used by the collision avoidance and navigation behavior as its goal location, as described above.

Note that these two behaviors, navigation/collision avoidance and tracking, operate with difference rates of decision cycles. The collision avoidance and navigation behavior makes decisions for the robots steering and translational rates approximately two to four times a second. The tracking behavior moves the goal location for the robot and the controls the rate of rotation of the turret approximately once every few seconds.

6. Conclusion

One assumption in this work is that it is possible to decompose a complex system into more basic behaviors and that these learned behaviors can then be recombined into a useful system. It is possible that the base behaviors are not optimal given the additional tasks that are to be performed. Future work will look at this question and on how to retrain the system with all behaviors in place.

Another interesting area that has been examined by many researchers is what to do when several behaviors try to control the same effectors. This work does not address that issue, although as we scale up, this will have to be examined.

References

Grefenstette, J. J., C. L. Ramsey, and A. C. Schultz (1990). "Learning sequential decision rules using simulation models and competition," Machine Learning, 5(4), (pp. 355-381).

Schultz, Alan C., Grefenstette, John, J. (1992). "Using a genetic algorithm to learn behaviors for autonomous

vehicles," Proceedings of the of the AIAA Guidance, Navigation and Control Conference, Hilton Head, SC, August 10-12, 1992.

Schultz, Alan C. (1991). "Using a genetic algorithm to learn strategies for collision avoidance and local navigation," Proceedings of the Seventh International Symposium on Unmanned Untethered Submersible Technology, September 23-25, 1991, Durham, N.H.

Schultz Alan C., C. L. Ramsey, and J. J. Grefenstette (1990). "Simulation-assisted learning by competition: Effects of noise differences between training model and target environment," Proceedings of the Seventh International Conference on Machine Learning, Austin, TX: Morgan Kaufmann (pp. 211-215).