Learning Robot Action Plans for Controlling Continuous, Percept-driven Behavior *

Michael Beetz[1] and Thorsten Belker[2]

[1] Computer Science Dept. IX, Munich University of Technology, Munich, Germany
[2] Dept. of Computer Science III, University of Bonn, Bonn, Germany

Abstract. Autonomous robots, such as robot office couriers, need control routines that support flexible task execution and effective action planning. This paper describes XFRMLEARN, a system that learns structured symbolic robot action plans for navigation tasks. Given a navigation task, XFRMLEARN learns to structure continuous navigation behavior and represents the learned structure as compact and transparent plans. The structured plans are obtained by starting with monolithical default plans that are optimized for average performance and adding subplans to improve the navigation performance for the given task. Compactness is achieved by incorporating only subplans that achieve significant performance gains. The resulting plans support action planning and opportunistic task execution. XFRMLEARN is implemented and extensively evaluated on an autonomous mobile robot.

1 Introduction

Robots operating in human working environments and solving dynamically changing sets of complex tasks are challenging testbeds for autonomous robot control. The dynamic nature of the environments and the nondeterministic effects of actions requires robots to exhibit concurrent, percept-driven behavior to reliably cope with unforeseen events. Most physical control processes are continuous in nature and difficult to represent in discrete symbolic structures.

In order to apply high-level plan-based control techniques, robots need adequate and realistic representations of their control processes that enable their planning routines to foresee problems and forestall them. In this paper we take the navigation behavior of autonomous mobile robots as our prototypical example.

Different approaches have been proposed to specify the navigation behavior of such robots. A number of researchers consider navigation as an instance of Markov decision problems (MDPs) [KCK96]. They model the navigation behavior as a finite state automaton in which navigation actions cause stochastic state transitions. The robot is rewarded for reaching its destination quickly and reliably. A solution for such problems is a *policy*, a mapping from discretized robot poses into fine-grained navigation actions.

MDPs form an attractive framework for navigation because they use a uniform mechanism for action selection and a parsimonious problem encoding. The navigation policies computed by MDPs aim at robustness and optimizing the average performance. One

* This research is partially funded by the Deutsche Forschungsgemeinschaft.

of the main problems in the application of MDP planning techniques is to keep the state space small enough so that the MDPs are still solvable.

Another approach is the specification of environment- and task-specific navigation plans, such as ***structured reactive navigation plans (*SRNPs*)*** [Bee99]. SRNPs specify a default navigation behavior and employ additional concurrent, percept-driven subplans that overwrite the default behavior while they are active. The default navigation behavior can be generated by an MDP navigation system. The (de-)activation conditions of the subplans structure the continuous navigation behavior in a task-specific way.

SRNPs are valuable resources for opportunistic task execution and effective action planning because they provide high-level controllers with subplans such as traverse a particular narrow passage or an open area. More specifically, SRNPs (1) can generate qualitative events from continuous behavior, such as entering a narrow passage; (2) support online adaptation of the navigation behavior (drive more carefully while traversing a particular narrow passage), and (3) allow for compact and realistic symbolic predictions of continuous, sensor-driven behavior. The specification of good task and environment-specific SRNPs, however, requires tailoring their structure and parameterizations to the specifics of the environment.

We propose to bridge the gap between both approaches by learning SRNPs, symbolic plans, from executing MDP navigation policies. Our thesis is that a robot can autonomously learn compact and well-structured SRNPs by using MDP navigation policies as default plans and repeatedly inserting subplans into the SRNPs that significantly improve the navigation performance. This idea works because the policies computed by the MDP path planner are already fairly general and optimized for average performance. If the behavior produced by the default plans were uniformly good, making navigation plans more sophisticated would be of no use. The rationale behind requiring subplans to achieve significant improvements is to keep the structure of the plan simple.

2 An Overview on XFRMLEARN

We have implemented XFRMLEARN a realization of this learning model and applied it to learning SRNPs for an autonomous mobile robot that is to perform office courier service. XFRMLEARN is embedded into a high-level robot control system called ***structured reactive controllers*** (SRCs) [Bee99]. SRCs are are controllers that can revise their intended course of action based on foresight and planning at execution time. SRCs employ and reason about plans that specify and synchronize ***concurrent percept-driven*** behavior. Concurrent plans are represented in a transparent and modular form so that automatic planning techniques can make inferences about them and revise them.

XFRMLEARN is applied to the RHINO navigation system [BCF+00], which has shown impressive results in several longterm experiments. Conceptually, this robot navigation system works as follows. A navigation problem is transformed into a Markov decision problem to be solved by a path planner using a value iteration algorithm. The solution is a policy that maps every possible location into the optimal heading to reach the target. This policy is then given to a reactive collision avoidance module that executes the policy taking the actual sensor readings into account [BCF+00].

The RHINO navigation system can be parameterized in different ways. The parameter PATH is a sequence of intermediate points which are to be visited in the specified order. COLLI-MODE determines how cautiously the robot should drive and how abruptly it is allowed to change direction. RHINO's navigation behavior can be improved because RHINO's path planner solves an idealized problem that does not take the desired velocity, the dynamics of the robot, the sensor crosstalk, and the expected clutteredness fully into account. The reactive collision avoidance component takes these aspects into account but makes only local decisions.

We propose an "analyze, revise, and test" cycle as a computational model for learning SRNPs. XFRMLEARN starts with a default plan that transforms a navigation problem into an MDP problem and passes the MDP problem to RHINO's navigation system. After RHINO's path planner has determined the navigation policy the navigation system activates the collision avoidance module for the execution of the resulting policy. XFRMLEARN records the resulting navigation behavior and looks for stretches of behavior that could be possibly improved. XFRMLEARN then tries to explain the improvable behavior stretches using causal knowledge and its knowledge about the environment. These explanations are then used to index promising plan revision methods that introduce and modify subplans. The revisions are then tested in a series of experiments to decide whether they are likely to improve the navigation behavior. Successful subplans are incorporated into the symbolic plan.

3 Structured Reactive Navigation Plans

Let us now take a more detailed look at the representation of SRNPs.

navigation plan *(desk-1,desk-2)*
  with subplans
    TRAVERSE-NARROW-PASSAGE(⟨*635,1274*⟩,⟨*635,1076*⟩)
      parameterizations *colli-mode* ← *slow*
      path constraints ⟨*635,1274*⟩,⟨*635,1076*⟩
      justification *narrow-passage-bug-3*
    TRAVERSE-NARROW-PASSAGE(...)
    TRAVERSE-FREE-SPACE(...)
  DEFAULT-GO-TO (*desk-2*)

The SRNP above contains three subplans: one for leaving the left office, one for entering the right one, and one for speeding up the traversal of the hallway. The subplan for leaving the left office is shown in more detail. The path constraints are added to the plan for causing the robot to traverse the narrow passage orthogonally with maximal clearance. The parameterizations of the navigation system specify that the robot is asked to drive slowly in the narrow passage and to only use laser sensors for obstacle avoidance to avoid the hallucination of obstacles due to sonar crosstalk.

SRNPs are called ***structured*** because the subplans explicitly represent task-relevant structure in continuous navigation behavior. They are called ***reactive*** because "perceived" qualitative events, such as entering or leaving a narrow passage, trigger the activation and termination of subplans.

4 XFRMLEARN in Detail

A key problem in learning structured navigation plans is to structure the navigation behavior well. Because the robot must start the subplans and synchronize them, it must be capable of "perceiving" the situations in which subplans are to be activated and deactivated. Besides being perceiveable, the situations should be relevant for adapting navigation behavior. Among others, we use the concept of ***narrowness*** for detecting the situations in which the navigation behavior is to be adapted. Based on the concept of narrowness the robot can differentiate situations such as free space, traversing narrow passages, entering narrow passages, and leaving narrow passages.

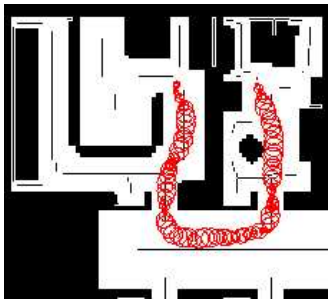

Fig. 1: Visualization of a behavior trace. The center of the circles denote the robot's position and the size of the circle the current speed of the robot.

Diagnosis of Conspicous Subtraces. Upon carrying out a navigation task, RHINO produces a behavior trace like the one shown in Figure 1. The robot's position is depicted by a circle where the size of the circle is proportional to the robot's translational speed. Behavior feature subtraces such as low and high translational speed, turning in place, and frequent stopping often hint at which behavior stretches can be improved. To infer how the improvements might be achieved, XFRMLEARN first tries to explain a behavior feature subtrace by finding environment feature subtraces, such as "traversing a narrow passage" or "passing an obstacle" that overlap with it. We use the predicate MAYCAUSE(F1,F2) to specify that the environment feature F1 may cause the behavior feature F2 like narrow passages causing low translational velocity. If there is sufficient overlap between a behavior feature subtrace *b* and an environment feature subtrace *e* then the behavior is considered to be a ***behavior flaw***. The diagnosis step is realized through a simple diagnostic rule that, depending on the instantiation of MAYCAUSE(?F1,?F2) can diagnose different kinds of flaws:

D-1 Low translational velocity is caused by the traversal of narrow passages.
D-2 Stopping is caused by the traversal of narrow passage.
D-3 Low translational velocity is caused by passing an obstacle too close.
D-4 Stopping caused by passing an obstacle too close.
D-5 High target velocity caused by traversing free space.

The "revise" step uses programming knowledge about how to revise navigation plans and how to parameterize the subsymbolic navigation modules that is encoded in the form of plan transformation rules. In their condition parts transformation rules check their applicability and the promise of success. These factors are used to estimate the expected utility of rule applications. XFRMLEARN selects the rule with a probability proportional to the expected utility and applies it to the plan.

For the purpose of this paper, XFRMLEARN provides the following revisions:

R-1 If the behavior flaw is attributed to the traversal of a narrow passage then insert a subplan to traverse the passage orthogonally and with maximal clearance.
R-2 Switch off the sonar sensors while traversing narrow passages if the robot repeatedly stops during the traversal.

R-3 Insert an additional path constraint to pass a closeby obstacle with more clearance.
R-4 Increase the target velocity for the traversal of free space where the measured velocity almost reaches the current target velocity.
R-5 Insert an additional path constraint to avoid abrupt changes in the robot's heading.

Because XFRMLEARN's transformation rules are heuristic, their applicability and the performance gain that can be expected from their application is environment and task-specific. Therefore XFRMLEARN learns the environment and task specific expected utility of rules based on experience.

The "test" step. Because plan transformation rules check their applicability and parameterization with respect to idealized models of the environment, the robot, the perceptual apparatus, and operation of the subsymbolic navigation system, XFRMLEARN cannot guarantee any improvements of the existing plan. Therefore, XFRMLEARN tests the resulting candidate plans against the original plan by repeatedly running the original and the revised plan and measuring the time performance in the local region that is affected by the plan transformation. The new candidate plan is accepted, if based on the experiments there is a 95% confidence that the new plan performs better than the original one.

5 Experimental Results

To empirically evaluate XFRMLEARN we have performed two long term experiments in which XFRMLEARN has improved the performance of the RHINO navigation system for given navigation tasks by up to 44 percent within 6 to 7 hours.

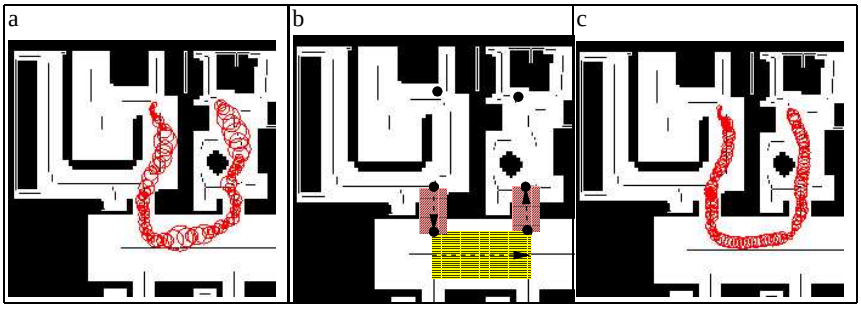

Fig. 2. Behavior trace of the default plan (a). Low T-Vel subtraces (b). Learned SRNP (c).

Figure 2(a) shows the navigation task (going from the desk in the left room to the one in the right office) and a typical behavior trace generated by the MDP navigation system. Figure 2(b) visualizes the plan that was learned by XFRMLEARN. It contains three subplans. One for traversing the left doorway, one for the right one, and one for the traversal of the hallway. The ones for traversing the doorways are TRAVERSENARROWPASSAGE subplans, which comprise path constraints (the black circles) as well as behavior adaptations (depicted by the region). The subplan is activated when the region is entered and deactivated when it is left. A typical behavior trace of the learned

SRNP is shown in Figure 2(c). We can see that the behavior is much more homogeneous and that the robot travels faster. This visual impression is confirmed by statistical tests. The t-test for the learned SRNP being at least 24 seconds (21%) faster returns a significance of 0.956. A bootstrap test returns the probability of 0.956 that the variance of the performance has been reduced.

In the second learning session the average time needed for performing a navigation task has been reduced by about 95.57 seconds (44%). The t-test for the revised plan being at least 39 seconds (18%) faster returns a significance of 0.952. A bootstrap test returns the probabilty of 0.857 that the variance of the performance has been reduced.

6 Conclusions

We have described XFRMLEARN, a system that learns SRNPs, symbolic behavior specifications that (a) improve the navigation behavior of an autonomous mobile robot generated by executing MDP navigation policies, (b) make the navigation behavior more predictable, and (c) are structured and transparent so that high-level controllers can exploit them for demanding applications such as office delivery.

XFRMLEARN is capable of learning compact and modular SRNPs that mirror the relevant temporal and spatial structures in the continuous navigation behavior because it starts with default plans that produce flexible behavior optimized for average performance, identifies subtasks, stretches of behavior that look as if they could be improved, and adds subtask specific subplans only if the subplans can improve the navigation behavior significantly.

The learning method builds a synthesis among various subfields of AI: computing optimal actions in stochastic domains, symbolic action planning, learning and skill acquisition, and the integration of symbolic and subsymbolic approaches to autonomous robot control. Our approach also takes a particular view on the integration of symbolic and subsymbolic control processes, in particular MDPs. In our view symbolic representations are resources that allow for more economical reasoning. The representational power of symbolic approaches can enable robot controllers to better deal with complex and changing environmants and achieve changing sets of interacting jobs. This is achieved by making more information explicit and representing behavior specifications symbolically, transparently, and modularly. In our approach, (PO)MDPs are viewed as a way to ground symbolic representations.

References

[BCF+00] W. Burgard, A.B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. Experiences with an interactive museum tour-guide robot. *ArtificialIntelligence*, 114(1-2), 2000.

[Bee99] M. Beetz. Structured reactive controllers — a computational model of everyday activity. In O. Etzioni, J. Müller, and J. Bradshaw, editors, *Proceedings of the Third International Conference on Autonomous Agents*, pages 228–235, 1999.

[KCK96] L. Kaelbling, A. Cassandra, and J. Kurien. Acting under uncertainty: Discrete bayesian models for mobile-robot navigation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1996.