Robot Planning

Drew McDermott
Yale University Computer Science Department
P.O. Box 2158 Yale Station
New Haven, Connecticut 06520

Robots and Planning

We used to know what planning was. It was the automatic generation of an action sequence to bring about a desired state of affairs. Given a goal like "All the green blocks must be off the table," a classical planner was supposed to generate a plan like, "Put block 33 on block 12; put block 16 on block 12; put block 45 on block 12," where blocks 16, 33, and 45 were the only green blocks on the table.

Nowadays nobody works on this any more. The problem as stated turned out to be too hard and too easy. It was too hard because it was intractable. It was too easy because action sequences are not adequate as a representation of a real robot's program. As often happens in AI, we are now trying to redefine the planning problem, or do away with it.

All robots have programs, in the sense that their behavior is under the control of a formally representable object (even though in some cases the object is compiled down to hardware). Once we focus on the robot's program, we realize that the robot's intentions and capabilities *are* its program. To change how the robot reacts to a situation, you change its formal specifications.

Now that we've explained robot programming, where does planning fit in? When we started, a plan was a sequence of actions, such as "First put block 33 on block 12; then put block 45 on block 12." If we ask how this fits into the world of programmed robots, the answer is clear: this is just a simple form of program. In particular, the robot's plan is that part of its program whose future execution it reasons about explicitly. The definition of robot planning is a natural corollary: Robot planning *is the automatic generation, debugging, or optimization of robot plans.* This list of three operations is merely a gloss on the phrase "explicit reasoning." There may be other operations that should be included.

A robot that learns by debugging parts of its program is an interesting special case. Here the system reasons about how to improve the program it just executed; but the reasoning is about how to generalize the current lesson to future similar circumstances. This counts as reasoning about the future execution of the program, so the part of the program that is improved by learning is the robot's plan.

A Survey of Approaches

There are several subareas of robot planning to be surveyed (plus a nonexhaustive sampling of names associated with each area):

1. *Minimalism:* Avoid planning as much as possible. Focus instead on finding ingenious ways to map current sensory inputs to immediate motor outputs. To the extent possible, eschew storing information inside the robot, instead attempting to resample it from the world on each sensory cycle. Encourage thinking about compiling behavior controllers down to "hardware," or at least software simulacra of logic circuits and finite-state machines. (Agre & Chapman 1987, Brooks 1991, Connell 1990, Kaelbling & Rosenschein 1990)
2. *Plan interpreters:* Devise new notations for plans capable of controlling a realistic robot. Include conditionals, loops, and bound variables. Make use of sensory information wherever possible, but keep plans available at run time as explicit objects to manipulate. (Firby 1987, Ingrand & Georgeff 1990, Simmons 1990)
3. *Time-constrained planning:* Plan as much as possible in the time available. Develop planning algorithms that return better answers if given more time to run ("anytime algorithms"). (Boddy & Dean 1989)
4. *Motion planning:* Plan the motion of a robot through space, given a description of the obstacles in its path. Take the resulting data structure — either a single trajectory, or a specification of good travel directions from every point in a region — and use it to guide the robot's actual motion. (Latombe 1990, Miller & Slack 1991)
5. *Transformational Planning:* Improve a plan by foreseeing problems and patching them away. Think of planning as an operation on already executable

plans instead of as an operation that derives plans from goal specifications. (Hammond 1990, Simmons 1988)

6. *Plan learning:* Improve a plan by experiencing problems and patching them away. When a plan is just a collection of behaviors, focus on adjusting the mapping of sensory data to each element of the collection. In more elaborate cases, generate explanations of plan failures to use in repairing plans. (DeJong 1990, Maes & Brooks 1990)

How Much Planning do Robots Need?

So far, no agreed-upon theory or even foundation has emerged in this area. The minimalists have had a positive influence on the field by urging researchers to stay close to what real robots can do. But the need for planning will not go away. A robot that does no planning might be able to live happily in its own ecosystem, but will not be able to connect its way of being to ours.

Fortunately, even though planning is in general intractable, a little bit of it pays off. For example, if a robot has a map of the world, and if its goals relate to locations in that map, then a little bit of route planning is cheap and useful. From the viewpoint of the classical framework, the amount of plan generation is quite small. The hard part when dealing with real robots is to superimpose the constraints from the travel plan on the robot behavior controllers. It is this intermediate zone that is the current focus of some interesting research.

Suppose we tell a robot to go to a particular place and get something. We can't even get started unless we and the machine share a common name for the place. Once the robot has been told its destination, it must figure out how to map sensory data to motion along each leg of its journey. It then activates controllers to make each mapping effective at the appropriate stage.

However, if the robot is to optimize its overall travel plans, we must make sure that the low-level sensory→behavior controllers are also described as high-level geometric objects. If those controllers operate purely in terms of low-level sensory expectations ("move aimlessly until the ambient light is bright"), then the travel planner will have no way of realizing that two of its planned trips actually come quite close to one another, and could be merged. In other words, it must be able to extract a space-time scheduling problem from its plan, solve the problem, and install a new plan that reflects the solution. The plan must retain all the important accomplishments of the original plan, but achieve greater efficiency. At the present stage of the field, we lack tools for stating such requirements, let alone verifying that they are satisfied.

One thing the field of robot planning currently needs is a theoretical framework that will allow us to talk about operations on plans. Such a framework would unify the theory of concurrent programs in Computer Science with the theory of feedback processes in Control Theory. This conclusion will be distasteful to those who have found refuge from theory in robot hacking, but they may be cheered by the knowledge that a lot more robot hacking needs to be done.

Acknowledgments

Some of the work reported was supported by DARPA contract number DAAA15-87-K-0001, administered by the Ballistic Research Laboratory.

Bibliography

1 Philip E. Agre and David Chapman 1987 Pengi: an implementation of a theory of activity. *Proc. AAAI* **6**, pp. 268–272

2 Mark Boddy and Thomas Dean 1989 Solving time-dependent planning problems. *Proc. Ijcai* **11**

3 Rodney A. Brooks 1991 Intelligence without representation. *Artificial Intelligence* **47**, special issue on foundations of AI, edited by David Kirsh, pp. 139–159

4 Jonathan H. Connell 1990 *Minimalist mobile robots.* Boston: Academic Press, Inc.

5 Gerald F. DeJong 1990 Explanation-based control: An approach to reactive planning in continuous domains. In Sycara 1990, pp. 325–336

6 R.J. Firby 1987 An investigation into reactive planning in complex domains. *Proc. AAAI* **6**, pp. 202–206

7 Kris Hammond 1990 Explaining and repairing plans that fail. *Artificial Intelligence* **45**, no. 1–2, pp. 173–228

8 Francois Felix Ingrand and Michael P. Georgeff 1990 Managing deliberation and reasoning in real-time AI systems. In Sycara 1990, pp. 284–291

9 Leslie Pack Kaelbling and Stanley J. Rosenschein 1990 Action and planning in embedded agents. In Patti Maes (ed.) *New architectures for autonomous agents: task-level decomposition and emergent functionality*, Cambridge: MIT Press

10 Jean-Claude Latombe 1990 *Robot motion planning.* Boston: Kluwer Academic Publishers

11 Pattie Maes and Rodney A. Brooks 1990 Learning to coordinate behaviors. *Proc. AAAI* **8**, pp. 796–802

12 David Miller and Marc G. Slack 1991 Global symbolic maps from local navigation. *Proc. AAAI* **9**

13 Reid Gordon Simmons 1988 A theory of debugging plans and interpretations. *Proc. AAAI* **7**, pp. 94–99.

14 Reid Gordon Simmons 1990 An architecture for coordinating planning, sensing, and action. In Sycara 1990, pp. 292–297

15 Katia Sycara (ed.) 1990 *Proceedings of the Workshop on Innovative Approaches to Planning, Scheduling, and Control.* San Mateo: Morgan Kaufmann Publishers, Inc.