Some Notes on the Integration of Planning and Reactivity in Autonomous Mobile Robots

Alessandro Saffiotti[*]
Artificial Intelligence Center, SRI International
333 Ravenswood Ave, Menlo Park, CA 94025
saffiotti@ai.sri.com

I glance at the "planning vs. reactivity" debate from the viewpoint of building autonomous mobile robots, and sketch an integration proposal.

Why autonomous navigation is difficult

In its essence, the problem of planning and controlling the motion of a mobile robot may appear pretty innocuous: you are given an environment, a robot with certain motor capabilities, a starting location, and a goal location, and you must find a way to have the robot move to the goal location. Fine: you compute a sensible path, grab the robot's controls, and guide it through this path to its goal. *Voilà*. But consider what happens when the robot itself , expected to behave autonomously, must solve this problem. According to the above scenario, it should use its knowledge of the environment, and of the location of the target and of itself in this environment, to plan a path to the goal. It should also use its knowledge of its own motion capabilities — the actions it can perform, their preconditions, and their expected effects. Finally, it should know what its goals are. (In this case, simply going to that single location).

We can make a number of assumptions on the accuracy of each of these types of knowledge. Depending on the assumptions made, the problem's statement and the problem's challenges vary enormously. Unfortunately, in any non-idealized case involving a real robot in a real-world environment *none* of the above types of knowledge is accurate. More to the point, most of the inaccuracy is inherent to the problem, and cannot be overcome by building a more sophisticated machine. The accuracy of the agent's knowledge about its environment is bounded by the noise and limited range of its sensors, and by the precision and granularity of the prior information. This is true, in particular, of agent's knowledge about its own location, and about the location of the target position. The complexity of the environment and the presence of other agents may cause unpredictable changes, and knowledge of the current state cannot be safely projected, whatever its quality. The effect of each action is muddled by errors in the effectors, as well as by the weakness of the knowledge about the situation where it is performed. Finally, the agent's knowledge about its very goals may be inaccurate: e.g., goals may become clearly defined only during execution — I find out where exectly Versailles is only after my arrival in Paris; or new goals may be discovered that were unknown (and unknowable) at planning time — the telephone rings while I am carrying a fruitcake to the kitchen.

This inherent weakness of knowledge hinders any naive solution of the problem of autonomous mobile robot motion planning. The granularity of any planning activity is limited by the granularity of the knowledge available at planning time. But a coarse plan involves complex atomic steps, and the uncertainty about the success (or even feasibility) of each step increases with its complexity. To make things worse, all agent's decisions and actions must take place at the time-scale of its environment: new goals, or unexpected world states, may need immediate attention. To be sure, the consequences of uncertainty and the delay of decision making could both be mitigated by limiting the range of the planning activity — ideally, by "planning" one step at a time. But this would be done at the cost of running into a number of well-known limited horizon problems. The field's literature often mentions the obituaries of two agents that failed in the delicate balance of careful forward-thinking and quick reactive response: Ponder, who was computing the provably best way to get out of the rails as the 10:54 express train was approaching; and Dynamo, who went downstairs to look for a gas leak carrying his reliable oil lantern.

Current solutions

Not surprisingly, AI researchers have faced this seemingly overwhelming web of problems by trying to isolate some of them as the *focus* of exploration, while treating all the others as *noise* to (momentarily) abstract from. The bet was that ideal solutions developed for the zero-noise case would extend to the full case by just adding some specific mechanisms to them. The first problem to be isolated was the automatic generation of abstract plans — plans were generated, possibly proved correct,

[*]On leave from Iridia, Université Libre de Bruxelles, Brussels, Belgium.

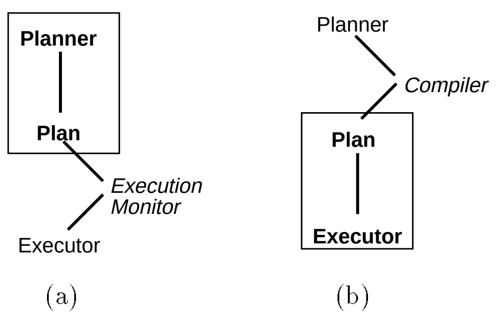

Figure 1: (a) "Planning-first." (b) "Reactivity-first."

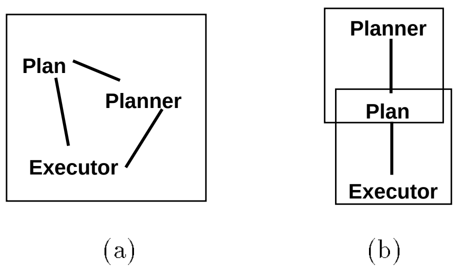

Figure 2: (a) "Integrated." (b) "Two-levels."

and executed in simulated worlds where all the necessary information was available, complete and correct (e.g., [Fikes and Nilsson, 1971]). When the necessity arose of having a robot to physically execute those plans in a real (though simplified) environment, the planning mechanisms would be enriched with devices that try to cope with some of the messy aspects — see Figure 1(a). Typical devices include a sophisticated execution module to take care of the low-level execution details; an execution monitor to check the actual state of the world during execution and re-direct branches of conditional plans accordingly; and re-planning capabilities in case the monitor detects an excessive deviation from the assumptions made at planning time. All of these techniques were already incorporated, at the beginning of the seventies, in Shakey the robot [Nilsson, 1984], and approaches based on extending a somewhat classic planning framework to cope with some of the challenges of real-world autonomous operations are still proposed in the literature. For example, [Simmons, 1990] incorporates some reactivity *within* a planning framework by introducing a number of real-time monitoring and deliberation devices; and [Lazanas and Latombe, 1992] show how to use path-planning techniques despite some (bounded) uncertainty in the sensors and effectors.

By the second half of the eighties mobile robots were starting to inhabit many AI laboratories, and the finger was now on how they could effectively interact with their environment. It was felt that the situation where "things do not go as expected" was much more the norm than the exception. As a consequence, many researchers rejected the idea of temporal projection as based on too unrealistic assumptions, instead trying to base agent's behavior on local, instantaneous perceptual data. Many proposal of reactive systems for controlling mobile robots flourished almost simultaneously. Most of these systems can be seen as specialized robot programming languages, normally based on short stimulus–response loops incorporating little or not state. The program itself was meant to be written by a human programmer for a specific task: the need for an abstract automatic planning activity was now relegated in the background as noise. Some researchers focussed on how complex activities can result from simple reactive mechanisms (e.g., [Agre and Chapman, 1987; Firby, 1987]); others went as far as denying the necessity of any symbolic process *tout court* [Brooks, 1987]. A third group acknowledged the need for symbolic planning in autonomous agents, and tried to extend reactive systems with some hooks to higher-level deliberation process — either in the concepts or in the architecture. One solution was to provide ways to bias a general reactive system toward the achievement of a goal by imposing constraints [Drummond, 1989], or by exploiting geometric reasoning [Payton *et al.*, 1990]. A perhaps more widespread approach sees planning as the goal-directed compilation of one instance of some type of reactive machine where the goal, and the corresponding plan, have been hard-wired (e.g., [Kaelbling, 1988]; [Nilsson, 1992]) — see Figure 1(b). This view of planning is reminiscent of "universal plans" [Schoppers, 1987], where a plan is seen as a pre-compiled specification of reactions to each possible situation. However, the languages of reactive systems are often much more complex than the action notations manipulated by existing planners, and the feasibility (and cost!) of automatically generating programs for these systems remains largely unexplored.

Whether planning-oriented or reactivity-oriented, most researchers working on situated activity seem to agree that autonomous agents need both types of capabilities. Some work has been done that takes this need as primitive, and adopt the integration itself as the main focus. The idea here is that action is determined by the dynamic interaction between an intended plan and the environmental contingencies, and the urging question is how to model this interaction. A possible approach consists in embedding both the deliberation and the reactive control mechanisms into one homogeneous framework — see Figure 2(a). An interesting example of this approach is provided by the PRS system [Georgeff and Lansky, 1987]. PRS is meant to be a computational embodiment of a formal theory of rational agency based on the interaction between belief, desires and intentions. Although PRS does not enforce any particular interaction schema in its architecture,[1]

[1]To this respect, PRS is not a planner nor a reactive controller — it just provides mechanisms to write them.

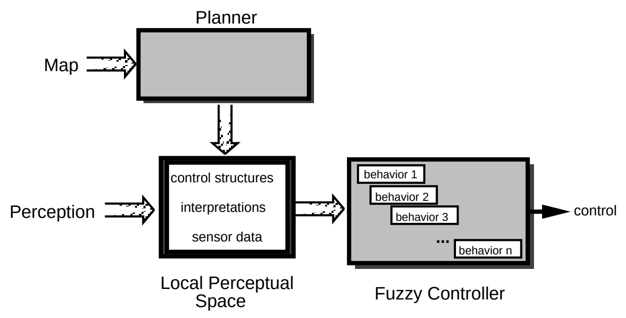

Figure 3: Architecture for the proposed integration.

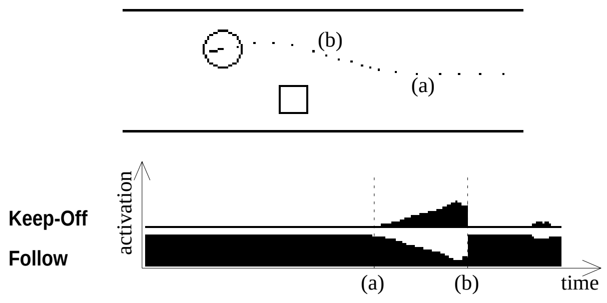

Figure 4: Fuzzy blending of behaviors.

its mechanisms suggest that low-level control and high-level deliberation activities be tightly intermingled. In fact, deliberative and reactive activities are carried on in PRS by essentially the same algorithms using the same representations inside one single execution loop. An alternative approach consists in keeping strategic planning and reactive control in two separate modules, on the ground that these two essentially different problems call for essentially different representation and reasoning mechanisms (e.g., [Arkin, 1990; Gat, 1991]). However, in order to ensure integration, some authors have insisted on the importance of a shared plan representation [Hanks and Firby, 1990]; i.e., a representation that is both adequate for specifying complex reactive controls, and for being reasoned about by higher level deliberative processes. The proposal I sketch below is meant to illustrate this point.

Sketch of an integration proposal

I outline here a proposal for a two-level integrated approach that takes the idea of shared plan representation seriously. This proposal has been implemented and tested on the SRI mobile robot Flakey, and is fully described elsewhere [Saffiotti *et al.*, 1993].

Figure 3 shows the architecture used. All the elementary *behaviors* that the agent can exhibit are implemented by a fuzzy reactive controller: these include "innate" behaviors, like avoiding obstacles; "purposeful" behaviors, like reaching a given position; and "perceptual" behaviors, like trying to detect a door. Behaviors take their input from a common storage structure, named *local perceptual space* (LPS), where perceptual data, interpretations built from these data, and *control structures* (CS's) are maintained. CS's are the basic ingredient of the shared plan representation: they are representation of subgoals created by the planner and exploited by the controller to orient agent's activity. In order to understand how this functions, I need to say how fuzzy behaviors are defined and combined.

Fuzzy behaviors are peculiar in two respects. Firstly, they do not directly generate effector commands; instead, each behavior generates a desirability measure over the space of possible effector commands from its particular viewpoint [Ruspini, 1991]. Secondly, many behaviors can be active at the same time. However, each behavior is associated with a *context* of applicability: the activation level of each behavior depends on how much the present situation matches that behavior's context. The fuzzy controller blends all the active behaviors, by generating effector commands that best satisfy all of them, modulo their respective activation levels. Figure 4 illustrates this blending: KEEP-OFF and FOLLOW are behaviors for obstacle avoidance and corridor following, respectively. In (a), an obstacle has been detected, and the preferences of KEEP-OFF are gradually gaining importance: Flakey slows and turns away from the obstacle. In (b), the path is clear, and the goal-oriented preferences expressed by FOLLOW regain dominance. (See [Ruspini, 1990; Saffiotti and Ruspini, 1993; Congdon *et al.*, 1993] for more on Flakey's fuzzy controller.)

Behaviors are activated by the presence in the LPS of a control structure. In a nutshell, a CS is a triple $S = \langle A, B, C \rangle$, where A is the representation of some object in the agent's workspace; B is a behavior involving that object; and C is a fuzzy predicate expressing the context of applicability. A typical CS can be

$$\langle \text{Corr1}, \textsc{Follow}, \text{near(Corr1)} \rangle$$

A CS can be thought of as an active object: as the agent approaches Corr1, the truth value of near(Corr1) increases; corresponding, the preferences generated by FOLLOW gain importance, and the agent is lead to follow the corridor. More complex behaviors can be produced by sets of CS's. For example, suppose that the following CS's are in LPS

$$S1 = \langle \text{Corr1}, \textsc{Follow}, \text{near(Corr1)} \land \neg\text{near(Door1)} \rangle$$
$$S2 = \langle \text{Door1}, \textsc{Cross}, \text{near(Door1)} \rangle$$

and that the agent is near(Corr1). Then, it will be driven by $S1$ to follow the corridor until it eventually approaches Door1 (supposed to be part of Corr1), where $S2$ will become dominant, leading the agent through the door. Notice that sequencing of $S1$ and $S2$

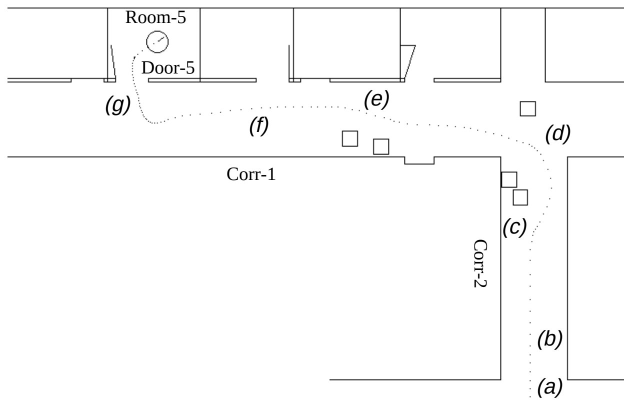

Figure 5: A sample run.

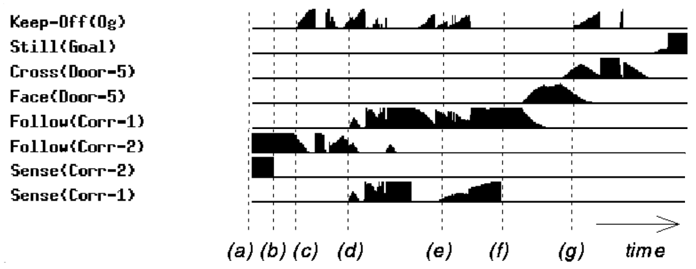

Figure 6: Temporal evolution of CS activations.

is not pre-imposed, but emerges as a result of behaviors' successfully producing their effects (in this case, near(Door1)). The actual activation order of the CS's depends on the contingencies encountered.

The above suggests that sets of CS's (CS-sets) are an adequate mechanism for writing complex real-time controls. In order to fit them in place in Figure 2(b) above, I will now show that they are also an adequate representation to be reasoned about by a high level deliberation process. To see this, notice that each fuzzy behavior can be abstractly described by a template that states its expected effect, when executed under certain conditions. For instance, the following templates describe the FOLLOW and CROSS behaviors.

```
Preconditions:  near(?c)
      Protect:  perceived(?c)
       Effect:  near(?p)
     Behavior:  Follow
     Artifact:  ?c
 With-binding:  corridor(?c), part-of(?c, ?p)

Preconditions:  near(?d)
       Effect:  at(?p)
     Behavior:  Cross
     Artifact:  ?d
 With-binding:  door(?d), leads-to(?d, ?p)
```

The "With-Binding" field constrains variable instantiation; the "Protect" field expresses a condition required during all the activation of the behavior (see below). Given a goal G, the task of the planner is to generate a CS-set from the templates such that: 1) G is entailed by the expected effect of some of the CS's in the plan; and 2) the preconditions of each CS are either already true in the initial state, or are expected to be brought about by some other CS. Each CS's context is built by ANDing that CS's preconditions and the negation of its effect (see $S1$ and $S2$ above).

Figure 5 shows an actual run on Flakey. Flakey was given the goal "at(Room-5)", and a sparse map of its environment. Based on these, the planner has generated the set of CS's listed on the left of Figure 6.[2] Beside each CS is a plot of its activation level over time during the execution of the plan. Notice the emerging sequencing of behaviors at (d), and around (g), and the interaction between purposeful behaviors and reactive obstacle avoidance — after (c), and around (d), (e) and (g). The interaction right after (g) is particularly interesting. CROSS(DOOR-5) relies on prior information about the door position; as this turns out to be fairly off, KEEP-OFF(OG) raises to avoid colliding with the edge of the wall. The combined effect is that Flakey engages the opening that is "more or less at the estimated position." The ability to integrate prior knowledge, extracted from the map used at planning time, and perception is one potential advantage of having a shared plan representation.

The two SENSE CS's provide a more sophisticated example of integration between perception and action. The "Protect" condition in the corridor-follow template expresses the need for real-time registration of the corridor. To ensure this, the planner has included the SENSE CS's, based on the following template:[3]

```
Preconditions:  near(?c)
       Effect:  perceived(?c)
     Behavior:  Sense
     Artifact:  ?c
 With-binding:  corridor(?c)
```

When entering a new corridor, Flakey needs to register it — (a) and (d); after traveling a few feet, the walls are perceived, and Flakey can move more freely (b). However, if Flakey loses track of the walls for a while (in (e) a wall is occluded by boxes) the SENSE behavior regains importance and slows Flakey down until re-registration occurs (f). Notice that no reasonable *a priori* order could have been established between the executions of corridor-following and corridor-sensing. Also, notice that explicit planning for perceptual actions could hardly be accounted for by plain path-planning.

[2]CS's are named after their behaviors and artifacts. OG is an "Occupancy-Grid", used for obstacle avoidance.

[3]The Sense behavior prefers slow and straight motion, as this is known to help sonar-based perception of walls; to perceive a corridor, Flakey needs to perceive its walls.

Conclusion

A solution to the problem of intelligent situated agency should include capabilities for both planned and reactive behavior. Many researchers have tried to extend solutions developed for the planning or the reactivity side alone to cover the other half of the problem. However, it is suggested that an integrated solution should attack both sides from the beginning, i.e., from the very definition of what a plan is. To illustrate this point, I have reported some recent work on an integrated architecture [Saffiotti *et al.*, 1993]. This architecture does not redefine in any substantial way the task of the planner, or the one of the controller: it redefines the notion of a plan. Instead of being a (partially) ordered set of operators, the plans I have discussed are sets of situation→behavior rules; and instead of specifying a sequence of discrete steps to execute, these plans induce a bias on a continuous reactive execution. Overall, the example I have discussed can be seen as a computational embodiment of the "plans as resources for actions" suggestion of Lucy Suchman [Suchman, 1987].

Acknowledgments

The integrated architecture I have discussed has been developed in cooperation with Kurt Konolige and Enrique Ruspini. David Israel and Daniela Musto also provided enlightening discussions. This work has been supported by a grant from the National Council of Research of Italy.

References

[Agre and Chapman, 1987] Agre, P. E. and Chapman, D. 1987. Pengi: an implementation of a theory of activity. In *Procs. of the AAAI Conf.* 268–272.

[Arkin, 1990] Arkin, Ronald C. 1990. The impact of cybernetics on the design of a mobile robot system: a case study. *IEEE Trans. on Systems, Man, and Cybernetics* 20(6):1245–1257.

[Brooks, 1987] Brooks, R. 1987. Intelligence without representation. In *Procs. of the Workshop on the Foudations of AI*, MIT, Cambridge, MA.

[Congdon *et al.*, 1993] Congdon, C.; Huber, M.; Kortenkamp, D.; Konolige, K.; Myers, K.; and Saffiotti, A. 1993. CARMEL vs. Flakey: A comparison of two winners. *AI Magazine.* To appear.

[Drummond, 1989] Drummond, M. 1989. Situated control rules. In *Procs. of the 1st Int. Conf. on Principles of Knowledge Representation and Reasoning.* 103–113.

[Fikes and Nilsson, 1971] Fikes, R. E. and Nilsson, N. J. 1971. STRIPS: a new approach to the application of theorem proving to problem solving. *Artificial Intelligence* 2:189–208.

[Firby, 1987] Firby, J. R. 1987. An investigation into reactive planning in complex domains. In *Procs. of the AAAI Conf.*

[Gat, 1991] Gat, E. 1991. *Reliable Goal-Directed Reactive Control for Real-World Autonomous Mobile Robots.* Ph.D. Dissertation, Virginia Polytechnic Institute and State University.

[Georgeff and Lansky, 1987] Georgeff, M.P. and Lansky, A.L. 1987. Reactive reasoning and planning. In *Procs. of the AAAI Conf.*

[Hanks and Firby, 1990] Hanks, S. and Firby, R. J. 1990. Issues and architectures for planning and execution. In *Workshop on Innovative Approaches to Planning, Sheduling and Control*, San Diego, CA.

[Kaelbling, 1988] Kaelbling, L. P. 1988. Goals as parallel program specifications. In *Procs. of the AAAI Conf.*, Minneapolis–St. Paul, MN.

[Lazanas and Latombe, 1992] Lazanas, A. and Latombe, J.C. 1992. Landmark-based robot motion planning. In *Procs. of the AAAI Fall Symposium*, Stanford, CA.

[Nilsson, 1984] Nilsson, Nils J. 1984. SHAKEY the robot. Technical Note 323, SRI Artificial Intelligence Center, Menlo Park, California.

[Nilsson, 1992] Nilsson, N. J. 1992. Toward agent program with circuit semantics. Technical Report STAN–CS–92–1412, Stanford University, Computer Science Dept., Stanford, CA.

[Payton *et al.*, 1990] Payton, D. W.; Rosenblatt, J. K.; and Keirsey, D. M. 1990. Plan guided reaction. *IEEE Trans. on Systems, Man, and Cybernetics* 20(6).

[Ruspini, 1990] Ruspini, E. H. 1990. Fuzzy logic in the Flakey robot. In *Procs. of the Int. Conf. on Fuzzy Logic and Neural Networks (IIZUKA)*, Japan.

[Ruspini, 1991] Ruspini, E. H. 1991. On the semantics of fuzzy logic. *Int. J. of Approximate Reasoning* 5.

[Saffiotti and Ruspini, 1993] Saffiotti, A. and Ruspini, E. H. 1993. Integrating reactivity and goal-directedness in a fuzzy controller. In *Procs. of the 2nd Fuzzy-IEEE Conference*, San Francisco, CA.

[Saffiotti *et al.*, 1993] Saffiotti, A.; Konolige, K.; and Ruspini, E. H. 1993. Now, do it! Technical report, SRI Artificial Intelligence Center, Menlo Park, California. Forthcoming.

[Schoppers, 1987] Schoppers, M. J. 1987. Universal plans for reactive robots in unpredictable environments. In *Procs. of the Int. Joint Conf. on Artificial Intelligence.* 1039–1046.

[Simmons, 1990] Simmons, R. 1990. An architecture for coordinating planning, sensing and action. In *Workshop on Innovative Approaches to Planning, Scheduling and Control*, San Diego, CA. 292–297.

[Suchman, 1987] Suchman, Lucy 1987. *Plans and situated actions: the problem of human machine communication.* Cambridge University Press, Cambridge, MA.