Integrating Deliberation in an Intelligent Agent Architecture

R. Peter Bonasso

Mobile Robotics Laboratory
ER4, NASA-JSC
Houston , TX 77058
(bonasso@aio.jsc.nasa.gov)

Abstract

We have been pursuing the integration of a state-based planner as the deliberative top layer of a three-tiered robot control architecture. The middle layer is implemented in the RAP system, while the bottom layer consists of a suite of reactive skills which can be configured by the RAPs system into Brooksian machines. Our target environment has been the ground control center of a space station command center. Our current system has been shown to provide a higher level of human supervision that preserves safety while allowing for task level direction, reaction to out-of-norm parameters, and human intervention at all levels of control. For this workshop, we discuss the issues we have been addressing with regard to where the boundary lies between local reactive control and long-range deliberation. Such issues include whether the seat of control should be at the RAPs system or at the planner, interruption of the RAPs system by the planner due to priority changes, how much of the planner's instruction can be taken as guidance versus direction, and levels of human interaction between the planner and the RAPs system.

Background and Motivation

Since the late eighties we have investigated ways to combine deliberation and reactivity in robot control architectures [Sanborn et al 1989, Bonasso 91, & Bonasso et al 92], in order to program robots to carry out tasks robustly in field environments. The robot control software architecture we are using is an outgrowth of several lines of situated reasoning research in robot intelligence [Firby 89, Gat 91, Connell 91, Slack 92, Yu et al 94, Elsaesser & Slack 94], and has proven useful for enabling mobile robots to accomplish tasks in field environments. This architecture, which is similar to ATLANTIS [Gat 91] and Cypress [Wilkins et al 94] in its concept if not in its implementation, separates the general robot intelligence problem into three interacting pieces (see Figure 1):

o A set of robotic specific reactive skills. For example, grasping, object tracking, and local navigation. These are tightly bound to the specific hardware of the robot and must interact with the world in real-time.

o A sequencing capability which can differentially activate the reactive skills in order to direct changes in the state of the world and accomplish specific tasks. For example, exiting a room might be orchestrated through the use of reactive skills for door tracking, local navigation, grasping, and pulling. In each of these phases of operation, the skills of the lower-level are connected to function as a what might be called a "Brooksian" robot -- a collection of networked state machines. We are using the Reactive Action Packages (RAPs) system [Firby 89] for this portion of the architecture.

o A deliberative planning capability which reasons in depth about goals, resources and timing constraints. We are using a state-based non-linear hierarchical planner known as AP [Elsaesser & MacMillan 91], the adversarial planner, since it grew out of research in counter planning. AP is a multi-agent planner which can reason about metric time for scheduling, monitor the execution of its plans, and replan accordingly.

These capabilities allow a robot, for example, to accept guidance from a human supervisor, plan a series of activities at various locations, move among the locations carrying out the activities, and simultaneously avoid danger and maintain nominal resource levels.

We have been successful in applying the reactive portion of this architecture to mobile land [e.g., Bonasso et al 92] and undersea robots [Bonasso & Barrett 93]. Recently, we have integrated the planner to control a free-flying, two-armed manipulator system in support of the maintenance of NASA's planned space station. This paper discusses issues associated with integrating planning with reactive control. Such issues include whether the seat of control should be at the RAPs system or at the planner, interruption of the RAPs system by the planner due to priority changes, how much of the planner's instruction can be taken as guidance versus direction, and levels of human interaction between the planner and the RAPs system.

Automated Robotic Maintenance of Space Station (ARMSS)

We have recently been using the architecture as a framework for controlling from the ground a two-armed manipulator robot maintaining a space station on orbit. The idea is that an intelligent ground control station can enable a ground crew to supervise the routine maintenance activities of the robot, and thus allow the on-orbit personnel

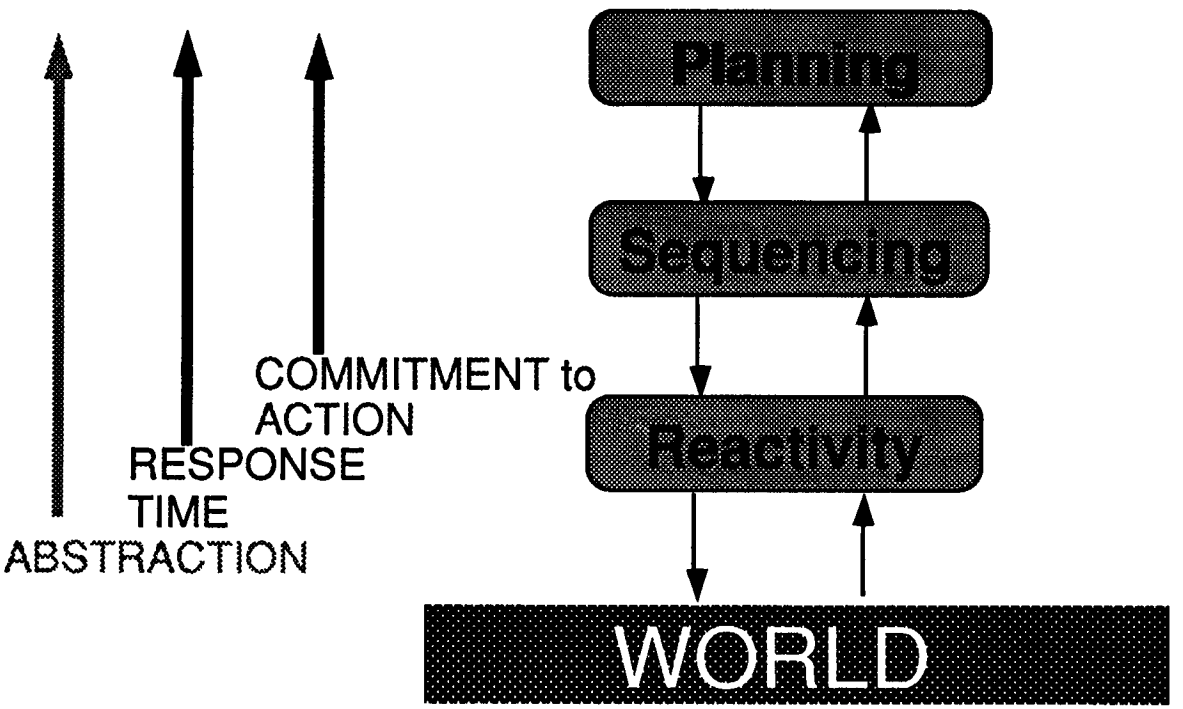

Figure 1: The Subject Intelligent Robot Control Architecture.

Commitment to action: At the skill level, skills are active only for the phase of the activity for which they are relevant. The sequencer has a longer commitment to the sequence of phases, and the planner has an even longer commitment to the overall plan.

Response Time: The skills operate normally at from 10 to 30 hz. The sequencer switches methods on the order of fractions of seconds to minutes, while the planner operates in the minutes to hours time frame.

Abstraction: The skill level is grounded in the sensor signals and motor commands of the physical robot. The sequencer abstracts these primitives to routine procedures, and the planner abstracts the procedures to the level of the overall goal.

to concentrate on scientific missions. We are using a 3-D kinematic simulation of such a robot, the EVA Helper/Retriever (EVAHR), to investigate our planning issues.

A subset of these issues can subsequently be examined in a testbed at NASA-JSC known as the ARMSS facility. ARMSS consists primarily of two 7 DOF Robotic Research manipulators, each mounted and moveable on vertical towers which in turn can move horizontally via a floor gantry. These arms can reach a portion of a space station truss on which are mounted a variety of components that can be inspected or repaired. Camera views are available from each end effector and from the towers and the floor. Existing ARMSS "skills" include inverse kinematic trajectory generation, velocity and position sensing and control, force/torque sensing and control, and ratchet and gripper control.

The Maintenance Scenario

The generic scenario for EVAHR (approximated physically in the ARMSS facility) is that each day it must carry out a set of routine maintenance tasks dictated by the status of components on the outside of the station. A repair plan is generated by the planner (at ground control). Such a plan might consist of inspections of items at various sites, retrieval of broken components or positioning of previously repaired items, or the simple changeout of malfunctioning items. The plan is executed by the RAPs system (on-orbit) and monitored by the planner. At any point, the plan may need to be altered by a major unplanned event such as a reboost of the space station to achieve a new orbit or the temporary loss of station power, or by minor interruptions, such as an astronaut needing help on an EVA mission.

Moreover, the planner must adjust the current plan by taking into account what is transpiring as the plan is executed. The RAPs system allows the robot to carry out most plan steps even in the face of minor problems, but those problems must be noted by the planner which may need to modify future plan steps (see Dealing with Planner Instructions below).

Implementation

Integrating the AP system with the RAPs system, while not trivial, has been relatively straightforward with regard to knowledge representation and timing. AP runs asynchronously from the rest of the architecture. Having generated the plan, the planner, in its execution monitoring mode, invokes a RAP for executing the first plan step. A typical AP primitive might be:

```
(Operator load-at-site
      :purpose (item-loaded ?arm ?site-item)
      :arguments ( (?site
         (get-site-from-memory ?site-item))) )
      :preconditions ( (attached-to ?planner ?site) )
      :effects (   (item-loaded ?arm ?site-item)
                   (palette-mount-for ?site-item ?mount)
                   (attached-to ?site-item ?mount)
                   (arms-status ?planner unfolded) )
      :task-time duration-of-load-at-dock )
```

There is one or more reactive action packages (RAPs) of methods to carry out this primitive in the RAP library. These are indexed via the purpose, preconditions, and possibly the effects clauses of the AP primitive. A typical RAP might be

```
(define-rap (load ?arm ?item)
 (succeed   (and    (palette-mount-for ?item ?mount)
                    (attached-to ?item ?mount)))
 (method normal-method
      (context (and (arm-place ?arm ?anywhere)
               (not (= ?arm dock-arm))
               (not (arm-holding ?arm ?any))
               (not    (or
                        (and (last-result ?Arm ?result)
                             (=?result MALFUNCTION))
                        (and (last-result gripper ?arm ?res)
                             (= res
                          GRIP_MALFUNCTION))))))
      (task-net
       (t1 (evahr-position-body ?item) (for t2))
       (t2 (arm-extract ?arm ?item) (for t3))
       (t3 (arm-install ?arm ?item))))
(method broken-arm-method
      (context (and    (or  (and (last-result ?Arm ?result)
                                 (= ?result
                                   MALFUNCTION))
                            (and (last-result gripper ?arm
                                   ?res)
                                 (= ?res
                              GRIP_MALFUNCTION)))
                            (arm-place ?other-arm
                              ?anywhere)
                            (not(or    (= ?other-arm ?arm)
                                       (= ?other-Arm
                                          dock-arm)))
                            (not (arm-holding ?other-arm
                                   ?any))))
      (task-net
       (t0 (disable-arm-move)  (for t1))
       (t1 (disable-linear-arm-move) (for t2))
       (t2 (evahr-position-body ?item) (for t3))
       (t3 (arm-extract ?other-arm ?item) (for t4))
       (t4 (arm-install ?other-arm ?item)))))
```

AP has a memory of the class structure of the world and some global items, but the majority of the posting and reporting takes place in the RAP memory. As this example should show, since both AP and RAPs use a propositional form of knowledge representation, an interlingua such as ACTs in Cypress is unecessary. So implementing the integration from this standpoint was more of an engineering exercise. However, dealing with other issues is not so clear.

Issues Involved In Integrating Deliberation

The following is a discussion of some of the issues we are addressing in our research beyond making two systems communicate with each other.

Who's In Control?

Gat, in ATLANTIS, put forth the notion that in reactive agent architectures, the local reactive intelligence (the sequencer) should treat the planner as an expert of sorts, invoking it when some deliberation is necessary or when it is at a loss for something to do. This stand may have been an outgrowth of the use of path planners wherein there is little if any interaction between subtasks. But for achieving conjunctive goals which have interacting subgoals, we believe that the sequencer doesn't have enough information to know when to request some deliberation. Nor can it infer when it should give up on a local task because of global constraints.

A representative example concerns time management. If the sequencer is making progress on a task it will continue working the task even though it is taking several standard deviations more time than is normal. The sequencer could be given a time limit of course, but whether that time limit should be a hard one depends on what else is going on globally. The planner can monitor the time on the task and choose to a) allow the task to continue because the task was started ahead of schedule due to unexpected success on other tasks, b) abandon future tasks of lower priority in favor of the current task, or c) dictate the abandonment of the current task in favor of higher priority future tasks.

Local Task Interruptions

Because safety and robustness are hallmarks of our architecture, we are especially concerned with the need to interrupt local task execution without leaving the robot in an awkward state. Since the RAPs system has knowledge of how to halt a task and gracefully safe the robot, one way to interrupt a task is for the planner to post an alert tag to the RAP memory. Eventually, the RAP interpreter will note the flag and stop the task while safing the robot. But in some instances -- e.g., an emergency boost of station to a new orbit -- such "cleanup" operations as folding the arms need to be omitted in order to get the robot to a secure dock site. The robot is an expendable in the light of certain crew emergencies, and the planner must recognize this and be able to command the equivalent of "power down and detach" (the equivalent of being cast adrift) in these cases.

Dealing with Planner Instructions

In our scenarios, the planner considers each arm and camera as a system resource, and thus selects which arm and/or camera to use for each maintenance task based on wear and tear criteria. During a repair task, the selected arm can fail in a number of ways (our simulation allows the interactive introduction of anomalies during execution). If the primary mode of the selected arm fails, RAPs will simply switch to the redundant mode and complete the current task. But the planner then needs to insure that future plan steps do not use that arm as the primary resource. If the selected arm fails both modes, RAPs will switch arms, but the planner will need to make major adjustments to the plan, i.e., no two-armed tasks can be undertaken.

But there is a more subtle problem. The point of the RAPs is to keep a certain amount of detail hidden from the planner so as to minimize planner computations. The planner essentially directs that an item be loaded and that the right arm ought to be used. For instance, what if the local situation is such that the selected arm is not feasible to use? This can occur when a specific item is reachable for example by one arm and not the other due to the positions of items in its delivery palette. So the question becomes in this case whether to include configuration space computations in the planner's repetoire. We prefer not to, but the result can be an infeasible suggestion of resources from one layer to the other.

Where to Interact With the Robot?

The architecture was designed to move the human from teleoperation to task supervision as one moves from skills to sequencing to planning. So it would seem that the point of interaction of the human in the three-tiered system is the planner. Yet, to understand where the robot is in its progress, the human may wish to know something about the state of execution below that of the planner; for example, which step in the current RAP is being executed. A pass-through could be effected for the human-robot-interaction at the planner level to query the RAP system for this information.

Often however, a RAP will fail to achieve the planner's purpose because of a mechanical situation which the human could remedy but not at the task level. In these situations, the planner usually has no recourse but to abandon the task. The human however, given an interface, can execute certain low-level RAPs to help the robot recover. An example is when communications between the RAPs and the skills breaks down momentarily. With an interface at the RAPs level, the human can intervene and restore the commmunications and then restart the current RAP. We have implemented a simple human-robot interface at both the planning and RAPs level to evaluate their use.

Summary

Integrating planning with a reactive architecture can be greatly facilitated by similar knowledge-representations. We have discussed an agent architecture instance where this is true. However, deeper issues are involved, particularly with environments that are dynamic not only locally, but globally as well. We have identified several of these issues and we are learning more about our architecture through implementation on more robots and

different scenarios. We believe our application environment is an excellent forcing function for investigating these issues.

References

R. Peter Bonasso. Integrating Reaction Plans and Layered Competences Through Synchronous Control. In Proceedings of the 12th International Joint Conference on Artificial Intelligence. Sydney, Australia. Morgan Kaufman.

R. Peter Bonasso, H.J. Antonisse, M.G. Slack. . A Reactive Robot System for Find and Fetch Tasks In An Outdoor Environment. In Proceedings of the Tenth National Conference on Artificial Intelligence. San Jose, CA. July 1992.

R. Peter Bonasso, R.P. & Barrett, J.A. A Reactive Robot System for Find and Visit Tasks in a Dynamic Ocean Environment. In Proceedings of the 8th International Symposium on Unmanned Untethered Submersible Technology, IEEE Oceanographic Society, Sep 1993.

J. H. Connell. A hybrid architecture applied to robot navigation. In Proceedings of the IEEE International Conference on Robotics and Automation, April 1992.

C. Elsaesser . Representation and Algorithms for Multiagent Adversarial Planning, MTR-91W000207, The MITRE Corporation, Dec 1991.

C. Elsaesser & M.G. Slack. Deliberative Planning in a Robot Architecture. In Proceedings of the AAIA/NASA Conference on Intelligent Robots in Field, Factory, Service and Space, March 1994.

James R. Firby. Adaptive Execution in Complex Dynamic Worlds. PHD Diss. YALEU/CSD/RR #672, Yale University. 1989.

Erann Gat. Reliable Goal-Directed Reactive Control of Autonomous Mobile Robots. PhD thesis, Virginia Polytechnic Institute Department of Computer Science, April 1991.

Jim Sanborn, B. Bloom, and D. McGrath. A Situated Reasoning Architecture for Space-based Repair and Replace Tasks, In 1989 Goddard Conference on Space Applications of Artificial Intelligence, Greenbelt, MD. NASA Pub # 3033.

M. G. Slack. Sequencing Formally Defined Reactions for Robotic Activity: Integrating RAPS and GAPPS. In Proceedings of the SPIE Conference on Sensor Fusion, November 1992.

Wilkins, David E., Myers, Karen L., Lowrance, John D., and Wesley, Leonard P. Planning and Reacting in Uncertain and Dynamic Environments. Journal of Experimental and Theoretical AI, Vol 7. 1994.

S. Yu, M. G. Slack, and D. P. Miller. A streamlined software environment for situated skills. In Proceedings of the AAIA/NASA Conference on Intelligent Robots in Field, Factory, Service and Space, March 1994.