Proceedings of the 1993 Winter Simulation Conference
G.W. Evans, M. Mollaghasemi, E.C. Russell, W.E. Biles (eds.)

SEMI-AUTOMATED FORCES FOR CORPS BATTLE SIMULATION

Erann Gat, Joe Fearey and Joe Provenzano
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, California 91109

ABSTRACT

This paper describes ongoing development of semi-automated forces (SAFORs) for the Corps Battle Simulation (CBS). Our approach adapts a control architecture which was originally developed to control autonomous mobile robots. The architecture extends a control methodology called behavior control with conditional sequencing and strategic planning capabilities. An initial implementation is scheduled to be completed in the third quarter of FY93.

1. INTRODUCTION

JPL is currently developing the capability of semi-automated play of simulated forces in the Corps Battle Simulation (CBS). CBS is the Corps/Division training simulation in the Army's Family of Simulations (FAMSIM). It is a command- and staff-oriented, unit-centered, medium-resolution combat simulation. It is in use as a local training tool in all the corps of the Army, and it is also the training simulation used by the Battle Command Training Program (BCTP), the Army's Corps/Division Combat Training Center (CTC).

The term semi-automated forces, or SAFORs, refers to forces in a combat simulation whose military activities are controlled by a computer program under guidance from a human controller, rather than directly by a human controller. For example, a SAFOR battalion might be given an order to secure a certain position. This order might specify a number of constraints such as schedules, available artillery resources, battalion boundaries, etc. The SAFOR would automatically fill in the details and issue appropriate game orders to its constituent units in order to achieve the objective within the given constraints.

The flexibility to mix units controlled semi-automatically with human-controlled units offers significant advantages both to trainers and to the training audience. Automating units on the opposition forces (OPFOR) would allow CBS to be played with a greatly reduced exercise-control staff. Depending on the pattern in which simulated units are designated for semi-autonomous control, the required manpower could be reduced up to 80% (approximately 300 personnel for a corps-level exercise). In an era where corps-level exercises are common, and far larger exercises are scheduled (e.g. Louisiana Maneuvers exercises with several thousand units in play), such manpower savings would be substantial.

Semi-automated play capability would also allow exercises to be conducted even when the personnel normally required to control allied units may not be available. For example, it is not always possible to coordinate U.S. exercise dates with the availability of Allied personnel to participate in critical roles. Semi-automated play would allow Allied formations to be realistically represented. It would also make possible specialized training for small training audiences without requiring coordination with large numbers of support personnel.

The rest of this paper consists of four sections. In the next section we present a brief summary of the technical background behind our approach. Section 3 describes the approach itself. Section 4 describes the proposed implementation. Section 5 summarizes.

2. BACKGROUND

OVERVIEW

Battle management is made difficult by three fundamental problems:

- Scarce and uncertain information about the environment
- Deadlines and other temporal constraints
- Unpredictability of the environment

The third issue is made particularly difficult in battle management by the adversarial nature of the task.

The control of tasks in the face of these three difficulties has been studied extensively in the context of controlling autonomous mobile robots [e.g. Gat90, Gat92a,b,c, Wilcox92, Miller92]. The fact that the underlying problems in the two domains are fundamentally the same has led us to investigate the

possibility of adapting these robot-control technologies to the battle management domain.

Our approach circumvents many of the technical obstacles which have been historically identified with the automation of battle management. (See [Bonasso88] for an overview.) Most of these historical difficulties arise from the widely held view that battle management is fundamentally a planning problem, i.e. that the fundamental goal of a battle management system is to produce a plan, or to assist in the production of a plan. Under this assumption, battle management inherits all of the myriad technical difficulties associated with planning [Hendler90].

We take the more recently introduced view that battle management can be seen as an example of situated activity, and is therefore fundamentally a process of continuous decision-making and improvisation [Agre90, Suchman87]. This view, embodied in a control paradigm called *behavior control*, has revolutionized the field of mobile robots over the past five years [Brooks91]. JPL has refined and extended this new approach (e.g. [Miller92]). In particular, JPL has developed a methodology for integrating classical AI planning techniques with behavior control, an extension which offers significant advantages in the battle management domain. Thus, we view planning as a component of the overall control problem, and not as the central problem.

In the following subsections we review the relevant technical aspects of behavior control, as well as the extensions which we have developed.

BEHAVIOR CONTROL

Behavior control is a control paradigm based on the concept of decomposing control problems by *task* rather than by *function*. The field of Artificial Intelligence (AI) has traditionally addressed the problem of autonomous control by decomposing a system into general-purpose functional modules, e.g. sensing, world-modelling, planning, plan execution, etc. Behavior Control advocates the decomposition of control systems into self-contained domain-specific task-achieving modules called behaviors. Typical behaviors in the mobile-robot domain include avoiding obstacles, navigating to some destination, searching for a target, etc. Simple behaviors are combined to produce more complex behaviors.

Behavior control has had dramatic success in solving many classically difficult problems in mobile robot control, most notably the problem of collision-free navigation. Behavior-based robots are typically much more reliable than classical robots, while requiring only a small fraction of the computational power. The underlying reason for this success is that it turns out to be very difficult to build general-purpose functional modules. In particular, general-purpose sensing and general-purpose planning turn out to be extremely difficult problems. Under the behavior-control paradigm the individual building blocks are task-specific. Information about the task can therefore be used to constrain and simplify the design of behaviors.

REACTIVE CONTROL AND THE ROLE OF STORED INTERNAL STATE

One of the results of work in Behavior Control is that useful behaviors very often can be made *reactive*. A reactive behavior is one which does not store any information in memory, but rather continuously reacts to its current situation. Reactive control has two significant advantages. First, because the computations required to implement reactive control are typically very simple, reactive systems usually have very fast response time and require only minimal computational resources. Second, basing the actions of a situated agent (i.e. an agent interacting with an environment) on information stored in memory can be dangerous because unexpected changes in the environment can invalidate the stored information and lead to erroneous actions. Reactive systems do not suffer from this problem because they do not store any information in memory.

Unfortunately, the scope of tasks which can be achieved using purely reactive systems is limited. Most realistic tasks require a certain amount of stored information. A semi-automated CBS unit, for example, would need to know (among other things) about its mission, current capabilities, surrounding terrain and the location of nearby enemy units in order to maneuver intelligently. In order to avoid the problems associated with stored information, the information must be managed carefully.

We use three techniques for dealing with this problem: abstraction, planning to advise, and conditional sequencing. These are described in the following subsections.

ABSTRACTION

The underlying problem with stored information is that it contains implicit predictions about the future state of the environment [Gat93]. Because of the unpredictable nature of realistic environments these predictions cannot be completely accurate. One way to increase the accuracy of stored information is

to reduce its precision. By using abstractions (representations with reduced precision) the accuracy of stored information can be increased. For example, stored information about the precise location of an enemy unit is not likely to remain accurate for very long. However, stored information about the general region in which an enemy unit is located is much more likely to be correct for a longer period of time. By abstracting away unpredictable aspects of the environment what remains is information pertaining to predictable phenomena which can be reliably stored for future use.

PLANNING TO ADVISE

Abstracted information is reliable, but likely to be incomplete. The decision-making apparatus which needs to use this information must therefore be able to handle imprecise, incomplete information. It is therefore impossible to treat plans in the classical way as analogous to a computer program, that is, as a step-by-step recipe to be executed more or less blindly.

Instead, we adopt Agre and Chapman's *Plans-as-Advice* approach [Agre90]. Under the plans-as-advice approach a plan is viewed not as a program which dictates future actions, but as a resource to be used to help make situated decisions. Plans-as-advice constrain actions, they do not dictate them.

Under this view it is quite straightforward to deal with incomplete and imprecise information. Because plans are no longer the prime movers in the system it is not necessary to have a plan at all in order to act. Because plans are viewed as information rather than commands, incomplete and imprecise plans can be used just as easily as complete and precise ones. Of course, a complete and precise plan usually leads to more efficient action, but it is not necessary (and usually not possible) to obtain one.

COGNIZANT FAILURE AND CONDITIONAL SEQUENCING

In unpredictable environments even imprecise and incomplete plans will occasionally fail. To ensure reliable operation, there must be a mechanism to detect and recover from failures. Detecting failures requires that a plan be annotated with a set of expectations which can be monitored. For example, a plan for a road march might have the expectation that no enemy units would be encountered. A failure that is detected is referred to as a cognizant failure.

If an expectation is violated, some alternate course of action needs to be taken. To continue the example, if the enemy is unexpectedly encountered on a road march, then the road march probably ought to be abandoned, at least temporarily. The alternate course of action may be built into the original plan (e.g. communicate the situation to the unit's superior unit and wait for new orders), or it may require the generation of a new plan on the basis of the newly acquired information (e.g. find a new route for the road march that avoids the enemy area).

For alternate courses of action built into a plan it is necessary to provide a fairly complex infrastructure for executing those plans. This is because in order to decide what alternate course of action to take, it is often necessary to remember information about what alternatives have been tried in the past. This is referred to as conditional sequencing, where the sequence of actions which actually occurs is conditional on what happens at each step. This is described in greater detail in the next section.

3. APPROACH

Our approach is based on Firby's Reactive Action Package (RAP) system [Firby89]. RAPs form a framework for doing conditional sequencing. We have extended the original RAP system to deal with simultaneous overlapping actions, and to perform on-line strategic planning.

The RAP system consists of three major components: the RAP memory, the RAP interpreter, and the RAP library. The RAP memory is the mechanism by which the system keeps track of what it knows about the state of the environment. The RAP interpreter is the execution mechanism which does conditional sequencing. Both of these components are described in detail in [Firby89].

The RAP library is the real heart of the system. The library is a collection of Reactive Action Packages (RAPs). Each RAP consists of a task description, and a list of alternative methods for achieving that task. Each method is annotated with a list of the circumstances under which it is applicable. The framework for specifying RAPs has been worked out in great detail, and includes ways of specifying arbitrary dependencies among task steps, deadlines, and resource constraints.

The library can be thought of as a sort of "playbook" which the system uses as its primary resource for deciding what to do. When an order is issued to a unit, it looks up the appropriate entry in the playbook, and then chooses a particular method based on the current situation. A RAP is roughly analogous to a behavior in the Behavior Control

paradigm. A RAP can invoke other RAPs, allowing complex RAPs to be constructed from simpler ones.

One of the unique features of this approach is that human guidance can be inserted at any point in the system. Ultimately, SAFORs respond at the top level to human-issued commands. However, human guidance can also be given to the system in the form of on-line plans, or as RAPs. Different sets of RAPs will produce different unit behaviors, reflecting different combat doctrines. Different RAP libraries can be swapped in and out in order to provide a variety of different training environments.

EXAMPLE

An example of a RAP is shown in figure 1. This example is adapted from Firby's original work. Since the RAP interpreter is written in LISP, the RAPs are written in a similar syntax. RAP variables are indicated by a prefixed question mark. The example ran in a domain which was vastly simpler than CBS, and where the focus was on logistics rather than combat. The system controlled a single simulated robotic delivery truck which could carry a single nondescript weapon which could be loaded or unloaded.

The purpose of this RAP was to tell the delivery truck how to deal with unexpected appearances of enemy troops. The RAP is a continuous-monitor, that is, it runs all the time in the background rather than being invoked in service of a specific task. The RAP has two methods, one to fight for a while (10 time units) and then retreat (unless the enemy retreats or is destroyed, in which case the monitor condition becomes false and the RAP becomes inactive), and the other to simply retreat. The first is applicable if the truck is carrying a loaded weapon.

There is a great deal of subtlety in the definition of this RAP which is beyond the scope of this paper. For more details the reader is referred to [Firby89].

4. A NOTE ON ADVERSARIAL ENVIRONMENTS

Wargaming is an adversarial environment. While at first glance this may appear to be the source of potential difficulty, it is really more a feature to be exploited than a problem to overcome. The adversarial nature of combat imposes a great deal of structure and predictability on the environment which can be used to simplify the design of an automated player. The automated player can be designed under the assumption that other agents in the game will behave according to certain doctrines, e.g. enemy units will place minefields around their defensive positions, friendly units will not attack when approached, etc. It is not necessary to attempt to divine the intentions of a unit based on its actions.

While it is possible to get deeply entangled in mathematical theories of adversarial games (which often lead to infinite regresses of arguments along the lines of, "If he knows that I know that he knows...") it is not necessary to do so in order to produce effective play. In fact, anecdotal evidence indicates that human military commanders do not usually engage in such reasoning. Instead, they make assumptions about the behavior of the enemy and proceed to make decisions on that basis. A similar strategy can be used to produce a competent (though probably not brilliant) automated player.

It should be noted that the adversarial nature of the environment has very little impact on the architecture for an automated player, and much more of an impact on the knowledge encoded within that architecture. The fundamental process is identical whether or not one incorporates a sophisticated adversarial theory: real-time control in an unpredictable, largely unknown environment. This means that it should be fairly straightforward to extend the simple assumption-based approach to a more sophisticated approach based on true adversarial planning if it is ever deemed desirable to do so.

```
(define-rap name: handle-enemy-troops
            type: continuous-monitor
            monitor-condition:
              (and (nearby ?enemy)
                   (class ?enemy enemy-unit))

  (method
    (context (and (location ?weapon weapon-bay)
                  (class ?weapon weapon)
                  (loaded ?weapon)))
    (task-network
      (task1 (shoot-at ?enemy ?weapon)
             (until-end task2))
      (task2 (retreat-from ?enemy)
             (until-end task1)
             (time-window 10 20)))

  (method
    (task-network
      (task1 (retreat-from ?enemy))))

)
```

Figure 1: An example of a RAP.

5. SYSTEM DESCRIPTION

The SAFOR will be part of an environment that includes the combat simulation (Corps Battle Simulation), the training audience, and a number of computer work stations that link the combat simulation with the training audience.

A SAFOR unit consists of a standard game unit and the controlling Automatic Player program (AP). Nominally, an AP should interact with a game unit in exactly the same way that a human player would. However, much of the human interface is devoted to presenting the huge quantity of data required to make decisions in a format which humans can easily assimilate. The AP requires no such aids, and can therefore interface with the game directly.

The AP will operate on a remote computer because the game consumes all of the computational resources available on the main CPU. In order to support future expansion, and to maintain compatibility with industry standards, the AP will interface to the game through a standard TCP/IP network connection. This will allow the AP to run on any computer connected to an ethernet. Modern, portable, object-oriented software development techniques will be used, allowing the AP program to run on just about any modern workstation, including Sun, IBM PC, and Macintosh computers.

A single AP program can control multiple units, and multiple AP's may be connected to the game at any time. An AP will appear to be a standard game unit in that it interacts with its human controllers by accepting orders and issuing reports. However, unlike standard game units, the AP will be able to accept orders at a very high level and be able to operate unsupervised for extended periods of time.

EXISTING CAPABILITIES

Some semi-automatic control already exists in CBS. Within the simulation itself game units have certain predefined behaviors that they exhibit in response to game orders and changes in the game environment. For example, a unit in motion along a road will stop and fight if it encounters an enemy unit, even if it was not given explicit orders to do so. More recently, the COAST system provides the ability to perform automatic infiltrations. The goal of the proposed work is to generalize and extend these capabilities in order to reduce the number of controllers required to run the simulation.

6. SUMMARY

We have described our approach to developing semi-automated forces (SAFORs) for the Corps Battle Simulation (CBS). We are adapting an approach which has been successfully applied in the past to the control of autonomous mobile robots. Our approach is based on the behavior control paradigm of task-wise decomposition, and Firby's Reactive Action Package system.

We expect SAFORs to allow substantial reductions in the personnel necessary to conduct CBS exercises, and to greatly increase the flexibility of exercise scheduling. SAFORs will also enable small-scale exercises which are not currently possible. We expect to have an initial prototype battalion commander SAFOR operational by third quarter FY93.

REFERENCES

[Agre90] Phil Agre, "What are Plans For?", *Robotics and Autonomous Systems*, vol. 6, pp. 17-34, 1990.

[Bonasso88] R. Peter Bonasso, "What AI can do for Battle Management: A Report of the First AAAI Workshop on AI Applications to Battle Management," *AI Magazine*, Fall 1988.

[Brooks91] Rodney A. Brooks, "New Approaches to Robotics," Science, 253, 1227-1254, September 1991.

[Firby89] R. James Firby, *Adaptive Execution in Dynamic Domains*, Ph.D. thesis, Yale University, 1989.

[Gat90] Erann Gat, et al., "Path Planning and Execution Monitoring for a Planetary Rover," *Proceedings of the IEEE Internation Conference on Robotics and Automation*, 1990.

[Gat91] Erann Gat, *Reliable Goal-directed Reactive Control of Autonomous Mobile Robots*, Ph.D. Thesis, Virginia Polytechnic Institute and State University, 1991.

[Gat92a] Erann Gat, "Integrating Planning and Reacting in a Heterogeneous Asynchronous Architecture for Controlling Real-world Autnomous Mobile Robots," *Proceedings of the Tenth National Conference on Artificial Intelligence* (AAAI), 1992.

[Gat92b] Erann Gat, "On the Role of Theory in the Control of Autonomous Mobile Robots," Presented at the AAAI Fall Symposium on the Applications of AI to Autonomous Mobile Robots, 1992.

[Gat92c] Erann Gat, et al., "Simple Sensors for Performing Useful Tasks Autonomously in Complex Outdoor Terrain," Proceedings of the SPIE Conference on Sensor Fusion, 1992.

[Gat93] Erann Gat, "On the Role of Internal State in the Control of Autonomous Mobile Robots," *AI Magazine*, Spring 1993 (to appear).

[Hendler90] James Hendler, Austin Tate and Mark Drummond, "AI Planning: Systems and Techniques," *AI Magazine*, Summer 1990.

[JPL91] CBS 1.3 System Description, JPL publication D-7850, Jet Propulsion Laboratory, California Institute of Technology, 1991.

[Miller84] David P. Miller, "Planning by Search Through Simulations", Technical Report YALEU/CSD/RR423, Yale University, 1984.

[Miller92] David P. Miller, et al., "Reactive Navigation Through Rough Outdoor Terrain: Experimental Results," *Proceedings of the Tenth National Conference on Artificial Intelligence* (AAAI), 1992.

[Suchman87] Lucy Suchman, *Plans and Situated Actions*, Cambridge, 1987.

[Wilcox92] Brian Wilcox, et al., "Mobile Robots for Planetary Exploration," *Proceedings of the IEEE Internation Conference on Robotics and Automation*, 1992.