

A Behavioral Architecture for Strategy Execution in the RoboFlag Game

Lyle Chamberlain¹, Japeck Tang¹, Megha Watugala¹, Julie A. Adams², and Michael Babish³

¹Division of Engineering and Applied Science
California Institute of Technology, Pasadena, CA 91125, USA
{lyle, japeck, megha@its.caltech.edu}

²Department of Computer Science
Rochester Institute of Technology, Rochester, NY 14623 USA
adamsj@ieee.org

³Mechanical and Aerospace Engineering
Cornell University, Ithaca, NY 14853 USA

Abstract

Strategy definition for teams of multiple mobile robots with fewer human operators than robots is a continuing area of research. Before strategies can be defined, the medium upon which they are implemented must be understood. This paper describes the development of a hybrid multi-agent behavioral control architecture used to implement the strategies for game play in the RoboFlag environment. Additionally, the graphical-based human-robotic interface is described. A framework has been built upon which systematic evaluation of strategy definition components can be performed.

1 Introduction

A significant amount of research has been conducted in the area of robot control, particularly concerning the best way to control m robots with n users where m is much greater than n , with or without a human-in-the-loop [1, 2, 3, 4, 5, 6]. When there are more robots than users, it becomes difficult to control the actions of the individual robots. Furthermore, the user may not be the best agent to control various aspects of the robot, such as low-level point-to-point motion behavior. Thus, it is desirable to find the optimal balance between autonomous behaviors as well as the level and manner of human interaction required to efficiently control one robot, a group of

robots, or a whole team of robots.

Human control depends largely on the method of strategy definition used to communicate with and program the robot team. In searching for a strategy definition technique, the difference between defining a strategy and implementing a strategy must be recognized. A strategy definition that does not take into account real world issues is useless; consequently, this work initially focused on strategy implementation. The intention was to provide a control architecture definition that indicates exactly what information is needed in order to complete a task. Classifying this information provides a basis to describe a strategy in RoboFlag.

Two teams were tasked with developing a control paradigm to address the issues of strategy definition, strategy execution, and human-in-the-loop control for the RoboFlag game [7]. The authors' team, team Ithaca, traveled from California Institute of Technology (CalTech) to conduct this research at Cornell University. Their opponents, team Pasadena, from Cornell University went to CalTech to work. The research was conducted as part of CalTech's ten-week Summer Undergraduate Research Fellowship (SURF) program [8]. At the end of the ten-week period, the teams participated in a final RoboFlag competition at Cornell [9].

2 Design Goals

It was determined that the architecture would not provide static central control, thus avoiding the temptation to use a central controlling agent readily available in the RoboFlag environment. It was recognized that coordinated play might be more difficult to achieve, but that the results would be more applicable to real world/combat system designs. With this main design criterion in mind, the following design goals were set:

- Create a distributed control architecture that:
 - Is robust to real-world situations,
 - Provides easy introduction of Human-in-the-Loop Control,
 - Provides inter-agent communication,
 - Provides team control distributed or dynamically assigned to a leader, and
 - Provides appropriate abstraction to describe RoboFlag game play.
- Develop competitive and useful robot behaviors and team strategies to run on the architecture.
- Develop a Human-Robotic interface.

3 Control Architecture

A control architecture was developed that follows the popular three-layer hybrid control technique. The three-level architecture provides a method to overcome the inherent weaknesses of several different control methods [4, 6, 10, 11, 12].

The lowest level of the architecture contains modules that are designed to handle mathematically continuous control problems. This level is implemented via the point-to-point navigation method. This navigation method incorporates the obstacle avoidance methodology provided within the RoboFlag environment. The architecture's second level handles discrete states and continuous time. This level activates one of many different behaviors appropriate to the current situation dependent upon the internal state. For purposes of this work, the first layer is treated as provided and unchangeable, thus the

behaviors reside in the second layer. The third layer permits the representation of both time and state in a discrete manner. This representation allows Artificial Intelligence (AI) agents that are less useful for real-time control, such as planners, to be employed. Essentially, this layer contains a representation of a series of events that must occur as well as communicates with the second layer to select actions that are executed in real-time.

This implementation of the three-layer architecture uses behavioral control located in the second and third layers, though the third layer is not strictly constrained to behavior-based computation. A behavior-based control methodology was chosen because of its inherent distributed nature [13]. If a system has inherently distributed control resulting in emergent behaviors in just one robot, we hypothesize that emergent team behaviors will follow similar design methods. The distributed team control is further modeled by representing inter-robot communications as inter-behavior connections across robots. Subsumption is employed as the behavior coordination mechanism [13, 14].

A cooperative multitasking environment was implemented that executes on each robot. This environment supports Simple Behaviors and Compound Behaviors. A Simple Behavior is defined by Pirjanian et al. [6] as:

"A behavior is a perception to action mapping module that, based on selective sensory information, produces (recommends for) actions in order to maintain or achieve a given, well specified task objective."

For example, a Simple Behavior executes a simple task such as *Stay On Circle*. The *Stay On Circle* behavior attempts to maintain a robot's position on a circle with a given center and radius. A Compound Behavior is a collection of Simple Behaviors arbitrated by a subsumption network. A Compound Behavior can be treated as a single behavior, but it provides several discrete states and is usually designed to provide a complex goal-achieving behavior. For example, the *Circle Defense* Compound Behavior causes

the robots to form a circle around their defense zone to protect their flag. Throughout the summer a library of nearly forty Simple Behaviors was developed that can be used to quickly develop goal-achieving Compound Behaviors.

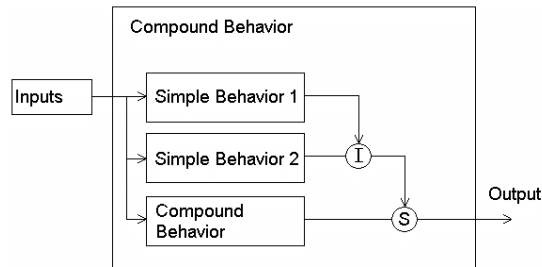


Figure 1. Behavior Organization. A Compound Behavior consists of one or more Simple or Compound behaviors arbitrated by a subsumption network.

Figure 1 shows an example of a Compound Behavior. The boxes represent behaviors, and the circles represent nodes in the subsumption network [13]. In this arbitrary example, the “I” node *inhibits* Simple Behavior 2’s signal if Simple Behavior 1 is active. The “S” node *subsumes* the Compound Behavior’s output with Simple Behavior 2’s signal if there is a conflict.

The Compound Behaviors can be employed as Competency Modules, in other words, a lower-level strategy. A single Competency Module executes at a time, and represents the second control layer. Compound Behaviors can also be employed as Strategy Modules. A Strategy Module represents a higher-level third control layer. Strategy modules determine which competency module should be executed based on the current game conditions, thus permitting the representation of more complex strategies. Strategy modules may include inter-robot communications, dynamic team leadership, and state-based strategy execution. Both the strategy module and competency module layers model communications as inter-behavior connections across agents.

Figure 2 provides the arrangement of the Strategy and Competency Modules into a three-layer architecture. This architecture

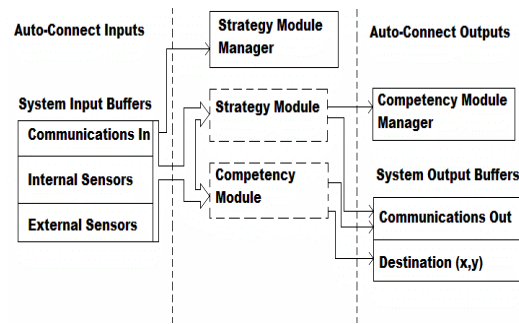


Figure 2. The Three-Layer Architecture Implementation. Destination connects to the first layer, Competency Module represents the second layer, and Strategy Module represents the third.

resides on each individual robot. The figure is divided into three sections (not to be confused with the three control layers). The middle section contains the second and third control layers in the form of the Competency and Strategy Modules, respectively. Dotted boxes indicate components that are dynamically instantiated by their corresponding Module Manager. The Competency Module Manager receives requests from the Strategy Module to instantiate specific Competency Modules. Currently, the Strategy Module Manager only receives requests to instantiate a Strategy Module from the human operator, who decides which strategy the robot should execute.

The Auto Connect Inputs section contains input buffers that represent a robot’s sensors and communications interface. Both the Strategy Module and the Competency Module can connect to these buffers. The Auto Connect Outputs section contains objects to which the Strategy Module and Competency Module can connect. The Strategy Module can connect *only* to the Competency Module Manager and the Communications buffer. Thus, it can only indirectly control the robot by requesting a certain type of goal-achieving behavior. The Competency Module connects to the Destination buffer. This buffer contains an (x, y) navigational coordinate and other navigational parameters for the navigation code provided in the RoboFlag environment.

4 Human Interface

An objective of the RoboFlag SURF program was the development of a human-robotic interface that would allow either a single user or multiple users to control the RoboFlag team during a game situation. This project chose to provide human control in the form of “Virtual Sensors.” Virtual Sensors are Simple Behaviors with a communications connection modeled as an inter-behavior connection to the human controller. Essentially, the human controller is treated the same as the robotic entities. This design permits the human(s) to interject control at any desired level simply by connecting the virtual sensor to the appropriate location in the subsumption network of either a Strategy Module and/or a Competency Module. Conversely, the robot can inform the human(s) of its situation and desires by establishing connections with the Virtual Sensors in the human interface client.

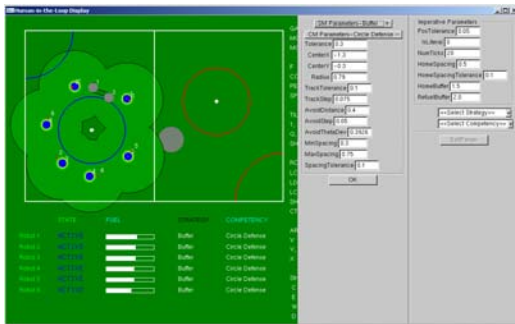


Figure 3. The human-robot interface with the *Circle Defense* Competency Module executing, parameter editing, line-of-sight *fog of war*, and fading obstacle.

In order to interact with the robots, a graphical-based human-robotic interface was developed, as shown in Figure 3. The interface provides various methods for selecting and controlling the robots, including selecting destinations. In addition, the human can select robot strategies as well as modify the strategy parameters during the game via the panel on the right side of the figure. Figure 3 demonstrates the robots executing the *Circle Defense* strategy and the *EditParam* button has been pressed. The interface also

provides a status panel that indicates each robot’s state, fuel level, as well as the currently running Strategy Module and Competency Module. The users are provided with the robot’s area of visibility, or *Fog of War*, that indicates sensory line-of-sight information. Furthermore, the interface provides a persistence display of enemy robots and obstacles after they have left the viewing regions of the team’s robots. This persistence is represented by a version of probabilistic fading.

5 Game Play

The final competition against team Pasadena provided insight into the usefulness of the system. From the beginning, our work focused primarily on developing a distributed control scheme; consequently, the human-robot interface was not as developed as the opposition’s. This was a calculated risk. The intention was that our robot team would require minimal human supervision, enabling it to perform complicated plays in which EVERY robot significantly contributed.

The primary problem was associated with obstacle avoidance. The obstacle avoidance employed for this work was that provided with the RoboFlag environment. It was incorporated as part of the First Control Layer. Behaviors in the Second Layer often contradicted the obstacle avoidance feedback. With no way to coordinate these opposing behaviors, the robots occasionally became unstable near obstacles, veering into them or rapidly oscillating in place. The obstacle avoidance routine necessitated constant human supervision of the entire robot team—negating any advantage the high level of autonomy should have provided. Future versions will incorporate obstacle avoidance in the Second Layer.

In all games, we made use of all three human teammates. Two were “operators,” each in charge of either offense or defense. The third human was an “overseer” who maintained overall team situational awareness and directed the two operators. In times of intense game play in one area of the field, both

operators would simultaneously control the affected robots.

An informal game was played without obstacles. It was found that suddenly the robot team's operation became more autonomous. The robots required much less "babysitting," and the team was able to simultaneously attack and defend on multiple fronts with minimal human supervision.

During the between-game periods, we were able to quickly modify the robots' behavior networks to adapt to game conditions. One behavior in particular was causing problems due to increased sensor noise on game day. With the library of nearly forty basic behaviors and the modular nature of the infrastructure, we were able to quickly prototype and test alternative control networks in the few minutes allotted between games.

6 Conclusions and Future Work

In conclusion, we ask "are we closer to defining a general method for strategy description?" The answer is "yes." Competency Modules provide an abstraction of the game concepts and goals. Our strategy and game play during the competition consisted of human selection of Competency Modules for each robot, as well as navigational coordinates used to guide a robot to a strategically useful destination. The third abstraction layer, the Strategy Module, contains one preliminary development. The limited project time (ten weeks with three people) hindered the team's ability to complete the Strategy Module. Therefore, it was more effective to allow the humans to select the appropriate Competency Modules. Future work should focus on developing the Strategy Modules that coordinate the positioning and Competency Module selection of cooperating teammates.

Due to time constraints, strategies and behaviors were developed in parallel with the overall infrastructure and architecture. Communications was one of the last infrastructure features implemented; consequently, most of the strategies and

behaviors were designed to operate without explicit communication. Balch and Arkin [15] demonstrate that useful communication between robots can be achieved implicitly through environmental modification and detection of peer robot position. Only one strategy employing explicit communication was developed. It is used to coordinate the refueling activities of peer robots, when the internal state (fuel level) of a peer robot cannot be ascertained from sensory data. For most tasks, implicit communication proved adequate.

Team Ithaca defined several strategies that work fairly successfully while creating a human-robotic interface with many useful features. Since most of the time was spent designing and developing the architecture, the human-robot interaction problems or the ease of strategy definition have not been fully explored. These represented goals of this project. However, we feel that the developed system is a useful tool for further research in both these domains.

Acknowledgments

The work described in this paper was supported in part by: the Defense Advance Research Projects Agency under cooperative agreement F30602-01-2-0577; the Air Force Research Laboratory, Information Directorate, PECASE award F49620-02-0388; and the Caltech SURF Program.

References

- [1] Adams, J.A. 1995. Human Management of a Hierarchical System for the Control of Multiple Mobile Robots. Ph.D. Dissertation. University of Pennsylvania.
- [2] Arkin, R.C. and Balch, T. 1998. Cooperative Multiagent Robotic Systems. *In Artificial Intelligence and Mobile Robots* edited by D. Kortenkamp, R.P. Bonasso, and R. Murphy. AAAI Press.
- [3] Mataric, M.J. 1997. Behavior-Based Control: Examples from Navigation, Learning, and Group Behavior. *Journal of Experimental and Theoretical Artificial*

- Intelligence, Special Issue on Software Architectures for Physical Agents.* 9(2-3): 323-336.
- [4] Parker, L.E. 1998. ALLIANCE: An Architecture for Fault Tolerant Multirobot Cooperation. *IEEE Transactions on Robotics and Automation.* 14(2): 220-240.
- [5] Parker, L.E. 1995. Heterogeneous Multi-Robot Cooperation. PhD Thesis, MIT.
- [6] Pirjanian, P.; Huntsberger, T.L.; Trebi-Ollennu, A.; Aghazarian, H.; Das, H.; Joshi, S.S.; and Schenker, P.S. 2000. CAMPOUT: A Control Architecture for Multi-robot Planetary Outposts. In the *Proceedings of SPIE Symposium on Sensor Fusion and Decentralized Control in Robotic Systems III*, Vol. 4196, Boston, MA: 221-230.
- [7] D'Andrea, R.; and Murray, R.M. 2003. The RoboFlag Competition. 2003 American Controls Conference, Invited Session. Denver, CO.
- [8] The CalTech Summer Undergraduate Research Fellowship Program (SURF): <http://www.its.caltech.edu/~surf>
- [9] Adams, J.A.; Hayes, A.T.; D'Andrea, R.; Murray, R.M. 2003. The RoboFlag SURF Competition: Results, Analysis, and Future Work. 2003 American Controls Conference, Invited Session. Denver, CO.
- [10] Connell, J.H. 1992. SSS: A Hybrid Architecture Applied to Robot Navigation. In the *Proceedings of the 1992 IEEE Conference on Robotics and Automation.* IEEE: 2719-2724.
- [11] Connell, J.H. and Viola, P. 1990. Cooperative Control of a Semi-Autonomous Mobile Robot. In the *Proceedings of the 1990 IEEE Conference on Robotics and Automation.* IEEE: 1118-1121.
- [12] Gat, E. 1998. Three-Layer Architectures. *AAAI Artificial Intelligence and Mobile Robots*, AAAI Press.
- [13] Brooks, R.A. 1986. A Robust Layered Control System for a Mobile Robot. *IEEE Journal of Robotics and Automation.* 2(1): 14-23.
- [14] Pirjanian, P. 2000. Multiple Objective Behavior-Based Control. *Journal of Robotics and Autonomous Systems.* 31(1-2): 53-60.
- [15] Balch, T. and Arkin, R.C. 1995. Communication in Reactive Multiagent Robotic Systems. *Autonomous Robots.* 1(1): 27-52.