CAMPOUT: A Control Architecture for Multi-robot Planetary Outposts

P. Pirjanian, T. L. Huntsberger, A. Trebi-Ollennu, H. Aghazarian, H. Das, S. S. Joshi, P. S. Schenker

Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Drive/MS 82-105
Pasadena, California 91109-8099
Paolo.pirjanian@jpl.nasa.gov

ABSTRACT

A manned Mars habitat will require a significant amount of infrastructure that can be deployed using robotic precursor missions. This infrastructure deployment will probably include the use of multiple, heterogeneous, mobile robotic platforms. Delays due to the long communication path to Mars limit the amount of teleoperation that is possible. A control architecture called CAMPOUT (Control Architecture for Multirobot Planetary Outposts) is currently under development at the Jet Propulsion Lab in Pasadena, CA. It is a three layer behavior-based system that incorporates the low level control routines currently used on the JPL SRR/FIDO/LEMUR rovers. The middle behavior layer uses either the BISMARC (Biologically Inspired System for Map-based Autonomous Rover Control) or MOBC (Multi-Objective Behavior Control) action selection mechanisms. CAMPOUT includes the necessary group behaviors and communication mechanisms for coordinated/cooperative control of heterogeneous robotic platforms. We report the results of some ongoing work at the Jet Propulsion Lab in Pasadena, CA on the transport phase of a photovoltaic (PV) tent deployment mission.

Keywords: behavior-based control, multiple mobile robots, robot outposts

1. INTRODUCTION

A control architecture provides a structured approach to design, specification, and hopefully implementation and validation of a complex control system and its subsystems. It usually defines a paradigm or philosophy for structuring the problem and imposes constraints that guide the way the control problem can be solved. We define a control architecture as the abstract design of a class of agents: the set of structural components in which perception, reasoning, and action occur; the specific functionality and interface of each component, and the interconnection topology between components. This definition identifies a number of architectural issues (such as perception action components, interfaces and topology between components etc.) that are useful in specifying and describing a particular architecture.

Robot control architectures can be broadly characterized as deliberative (based on planning), reactive (direct link between sensing and actuation), or a hybrid blend of the two. This hierarchy is shown in **Figure 1** with some representative examples of each type. CAMPOUT is a distributed, hybrid, behavior-based system in that it couples reactive and local deliberative behaviors without the need for a centralized planner. The control architectures that are closest to CAMPOUT in design are ALLIANCE [Parker (1994)], DAMN,[13] BISMARC,[4] and MOBC.[10] We have previously examined the use of behavior-based control for autonomous robotic outposts.[5,6]

There are numerous paradigms that can be followed for a control architecture design. CAMPOUT is characterized as hybrid within the realm of the behavior-based approach. Hybrid architectures provide the most general type of control due to combining low-level reactive components with high-level deliberative planners. Due to its generality, significant amount of demonstrated success, and available expertise within our group we select to realize CAMPOUT using the behavior-based paradigm. The behavior-based approach is mostly suited for the low-level reactive control aspects of an architecture. However, having planning behaviors is gaining more and more acceptance within behavior-based systems and will be included in the next generation of CAMPOUT. The approach used in CAMPOUT is highly distributed. First, behaviors within a single robot operate in a distributed manner thus allowing for concurrent and/or parallel execution of several tasks. Second, each robot can operate on its own, independently of other robots, based on its faculty of perception and action facilities. Cooperation between multiple robots occurs through active collaboration and with no centralized planning or decision-making component to dictate explicit commands.

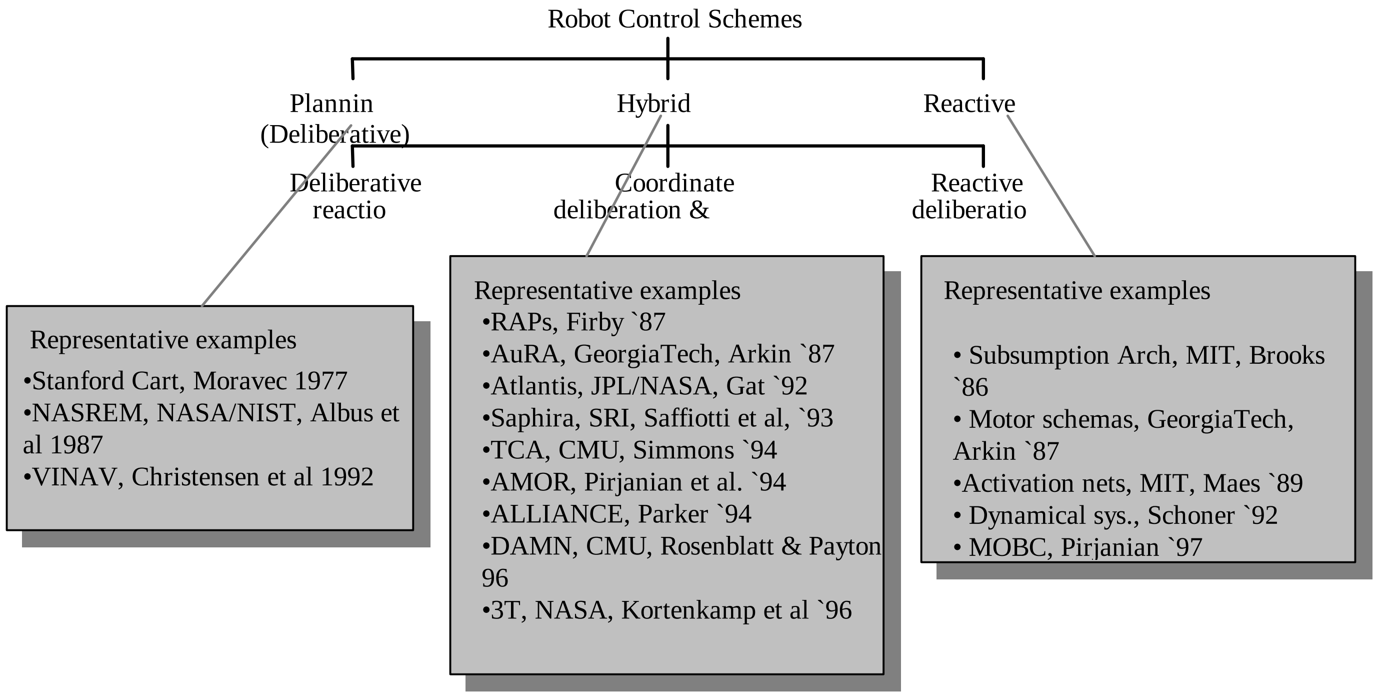

Robot Control Schemes
Plannin (Deliberative)
Hybrid
Reactive
Deliberative reactio
Coordinate deliberation &
Reactive deliberatio
Representative examples
•Stanford Cart, Moravec 1977
•NASREM, NASA/NIST, Albus et al 1987
•VINAV, Christensen et al 1992
Representative examples
•RAPs, Firby `87
•AuRA, GeorgiaTech, Arkin `87
•Atlantis, JPL/NASA, Gat `92
•Saphira, SRI, Saffiotti et al, `93
•TCA, CMU, Simmons `94
•AMOR, Pirjanian et al. `94
•ALLIANCE, Parker `94
•DAMN, CMU, Rosenblatt & Payton 96
•3T, NASA, Kortenkamp et al `96
Representative examples
• Subsumption Arch, MIT, Brooks `86
• Motor schemas, GeorgiaTech, Arkin `87
•Activation nets, MIT, Maes `89
• Dynamical sys., Schoner `92
• MOBC, Pirjanian `97

Figure 1. Robot control scheme hierarchy with representative examples.

The advantages of distributed control and coordination are: efficient use of system resources, parallel execution of multiple tasks, reliability and fault-tolerance to failure of individual components (including failure of single robots).

The following list constitutes the main characteristics that CAMPOUT includes as guidelines for design:

1. **Cognizant of failure:** components are designed so that they can determine success or failure of their task
2. **Fault tolerant:** no single point of failure, graceful degradation
3. **Distribute control:** to avoid single-point-of-failure and support scalability
4. **Scalable:** easily scale to more complex problems and larger number of components, robots etc.
5. **Ease of integration:** robotics is a highly multidisciplinary field, and requires efficient integration of many components (e.g., perception, mapping, localization, control, learning, etc.) that use different techniques, frameworks, and paradigms (e.g., classical control theory, AI planners, estimation theory, data fusion, computer vision, utility theory, decision theory, fuzzy logic, multiple objective decision making etc.). The architecture should provide the infrastructure, tools, and guidelines that allow the efficient use and integration of these components for meaningful interaction and operation.
6. **Rational decision making:** Satisficing vs. Optimal decisions, being realistic about resource limitations, multiple objective nature of the problem
7. **Explicit knowledge:** knowledge, formal, heuristic or otherwise, should have explicit representation to support easy maintenance, documentation, etc.
8. **Uncertainty handling:** its components should be able to operate reliably in face of uncertain and/or incomplete information, noisy sensors, and imperfect actuators
9. **Adaptivity:** stable and insensitive to perturbations in environment
10. **Learning.** The architecture should allow incorporation of learning capabilities. Learning enables the system and its capabilities to improve and evolve.
11. **Reactive <u>and</u> deliberative:** provide tight perception-action feedback loops to react promptly to unexpected situations and plan ahead of time for efficient use of resources. Plans should guide not control reactive components.

12. **Formal framework:** to enable the design of systems according to requirements, specifications, etc.
13. **Framework, tools, guideline, methods:** for supporting the design and implementation of all the above and evaluation of same. Given system task, mission requirements, resources, limitations etc. the framework should guide one to efficiently design, implement, and evaluate a system.
14. **Small overhead:** many "general architectures" introduce a prohibitive amount of overhead cost to a system. A key feature of the CAMPOUT architecture implementation is low overhead. This implementation may not be as general as desired but will be efficient. In particular, an architecture should find the right balance between generality and efficiency.

The next section discusses the overall organization of CAMPOUT, followed by implementation details and some preliminary experimental studies.

2. CAMPOUT

A high-level overview of CAMPOUT is shown in **Figure 2**. It is basically a three-layer architecture, which may be a derived from the types of environments in which planetary rover systems are expected to operate and survive.[3] A long duration mission such as a robot outpost on a planetary surface has a wide range of needs from the low level reactive components for local navigation and manipulator control to the higher level planning for large-scale construction tasks. The three-layer architecture spans these requirements through drivers directly tied to the actuators receiving commands from a behavior-based control hierarchy that is driven by a higher task planning layer.

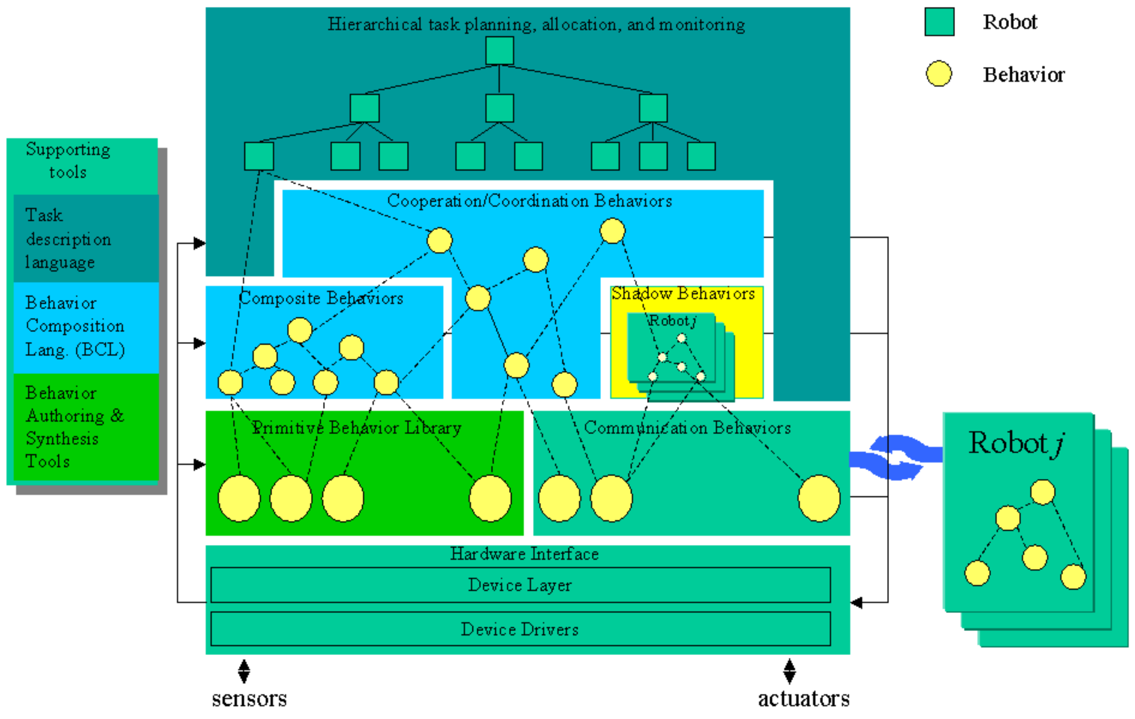

Hierarchical task planning, allocation, and monitoring
Robot
Behavior
Supporting tools
Task description language
Behavior Composition Lang. (BCL)
Behavior Authoring & Synthesis Tools
Cooperation/Coordination Behaviors
Composite Behaviors
Shadow Behaviors
Robot j
Primitive Behavior Library
Communication Behaviors
Robot j
Hardware Interface
Device Layer
Device Drivers
sensors
actuators

Figure 2. A logical block diagram of the Control Architecture for Multi-Robot Planetary Outposts, its components, interaction between components, interfaces, and tools.

The middle layer in CAMPOUT has a number of behaviors that are specifically tailored for not only cooperative but also tightly coordinated tasks.

2.1 Primitive Behavior Library

The main architectural substrate in CAMPOUT consists of a *behavior producing module* (commonly known as a *behavior*). A behavior is a perception to action mapping module that based on selective sensory information produces (recommendations for) actions in order to maintain or achieve a given, well specified task objective. For example, for safe navigation the system will often require a minimum of two behaviors: *AvoidObstacle* for safety and *GotoTarget* for navigation. Note that the *AvoidObstacle* behavior is only concerned with obstacle avoidance and obstacle avoidance only. Similarly, the *GotoTarget* behavior is only concerned with controlling the robot towards a target and is not concerned with obstacle avoidance at all, nor is it aware of the existence of the obstacle avoidance behavior. This limited-responsibility approach enables a remarkably efficient implementation of the behaviors. However, the behaviors can have conflicting objectives and hence an efficient behavior coordination mechanism (BCM) is required to resolve such conflicts and produce a useful combination of the behaviors into higher level behaviors, also know as composite behaviors (c-behaviors).

2.2 Composite Behaviors

Composite behaviors are constructed by careful combination of lower-level behaviors. At the lowest level of such a behavioral hierarchy we find the primitive behaviors (p-behaviors), which constitute a library of core capabilities for a robot. By coordinating the activities of the primitive behaviors we can construct a composite behavior that enhances the skill set of the robot. Composite behaviors can however also be constructed from other (lower-level) composite behaviors or a mix of primitive and composite behaviors. For instance a composite behavior, *SafeNavigation* can be constructed from the primitive behaviors *AvoidObstacle* and *GotoTarget* by a simple fixed priority-based coordination of the two that enables the *AvoidObstacle* behavior when the robot is close to obstacles and the *GotoTarget* behavior when the path of the robot is obstacle free.

2.3 Behavior Coordination Mechanisms

A central issue in many architectures is how to integrate its components in a well-understood way. Behavior coordination mechanisms (BCMs) provide tools for integration of behaviors to achieve higher-level goals. Priority-based behavior coordination represents a very primitive (but some times useful) type of coordination. CAMPOUT is an open architecture in the sense that any other behavior coordination mechanism can easily be integrated. Basically a BCM will be implemented as an operator (analogous to the logical AND or OR operators) and used to compose behaviors. These operators will also be provided in the Behavior Composition Language, in one form or another, to allow a high-level description language for behavior composition. We have done a detailed and extensive study of behavior coordination mechanisms.[11]

2.4 Communication Behaviors

The primitive and the composite behaviors constitute the skill set that enable a robot to interact with and accomplish tasks in its environment. The skill set of the robot can be augmented by adding new primitive and/or composite behaviors. For cooperation and interaction with each other the robots are required to communicate thus they must have a set of basic behaviors for communication. Communication is not necessarily limited to explicit exchange of information via some soft of a data link but can also include visual, auditory, tactile, and other types of communication. For instance a robot can determine the relative position of another robot using cameras. Alternatively, the other robot could explicitly transmit its position within a global coordinate system. Each approach might render appropriate for a specific task, system, or environment. CAMPOUT provides the methodology and infrastructure that support all such approaches.

2.5 Shadow Behaviors

The communication behaviors provide the information necessary to facilitate cooperation between a team of robots. This information is encoded in form of shadow behaviors (s-behaviors) that basically represent a remote behavior, including state information etc, running on a separate robot. Thus s-behaviors enable the behaviors to span a network of distributed behaviors over the robots. This will allow the behaviors to have access to remote sensing and actuation capabilities. Thus s-behavior need not be treated any different than the robot's own p-behaviors and c-behaviors.

2.6 Cooperation/Coordination Behaviors

In order to cooperate and collectively contribute to a common task the robots will have to cooperate and coordinate their activities. S-behaviors facilitate the composition of high-level behaviors that can achieve such coordination. As mentioned earlier, s-behaviors need not be treated any different than p- or c-behaviors (indeed their interfaces are exactly the same) thus coordination among several robots can be achieved by coordination of the activities of (a subset of) s-behaviors and the robot's behaviors using suitable behavior coordination mechanisms.

3. SYSTEM IMPLEMENTATION

In this section, we describe the current implementation of the CAMPOUT architectural components and infrastructure and showcase the use of its facilities and power of capabilities within the scope of a cooperative transport task. The challenging task of coordinated transport, as we will see, is particularly interesting because it involves the full use of CAMPOUT's facilities. In its current implementation, CAMPOUT supports no high-level planning capabilities, although planning components could be incorporated into the system but there are no specific capabilities that provide the infrastructure to support such a development. The following section will hence focus on the middle layers (the behavior-based layers) of the architecture.

3.1 CAMPOUT Architectural Components

Currently, CAMPOUT provides the following facilities for behavior representation, behavior generation, behavior coordination, and communications infrastructure for distributed robot coordination:

- **Behavior representation:** a set of abstract data types (known as objective functions) and related operations to describe the output of a general behavior as a multivalued preference.
- **Behavior prototyping toolkit:** provides a set of tools for rapid-prototyping of primitive as well as composite behaviors, i.e., facilities that can be used to easily develop behaviors.
- **Behavior coordination mechanisms:** that provide a repertoire of mechanisms that can be used to coordinate the activities of lower-level behaviors to form higher-level composite behaviors.
- **Communications infrastructure:** provides a set of tools and functions for interconnecting a set of robots and/or behaviors for sharing resources (e.g., sensors or actuators), exchanging information (e.g., state, percepts), synchronization, rendezvous etc.

3.1.1 Behavior representation

In our architectural methodology we formalize a behavior, b, as a mapping, b: $P \times X \rightarrow [0; 1]$, that relates each percept p and action x pair to a preference value that reflects the action's desirability. The percept describes possible (processed or raw) sensory input and the N-dimensional action space is defined to be a finite set of alternative actions. The described mapping assigns to each action $x \in X$ a preference, where the most desired actions are assigned 1 and undesired actions are assigned 0, from that behaviors point of view. Note that this definition of a behavior does not dictate how the mapping is to be implemented but provides a general recipe for a behavior with a well-defined interface (useful when composing behaviors regardless of their roles or positioning in a behavior hierarchy). This representation does not exclude implementation using a look-up-table, a finite state machine, a neural network, an expert system, control laws (such as PID etc.), or any other approach for that matter. Note also that this representation does not restrict us to reactive behaviors since it could have internal state. In that sense, each behavior can be implemented using whichever approach is appropriate. Finally, traditional, single-valued behaviors fall within this representation because, b: $P \rightarrow X$ can be represented by a multivalued output where all x are associated with 0 but the single x which is selected by the behavior. In CAMPOUT this representation is implemented using an N-dimensional array, which will contain the desirability values recommended by a behavior.

3.1.2 Behavior prototyping toolkit

The general behavior representation used in CAMPOUT does not suggest or prohibit any particular style or approach of implementation, but it does however provide a set of tools for developing behaviors, currently using rule-based and state-machine representations. CAMPOUT features a toolkit for synthesizing behaviors using fuzzy control. In fuzzy behavior-based control, each behavior is synthesized by a rule-base controlled by an inference engine to produce a multivalued output. E.g., an obstacle avoidance behavior can be encoded using to simple IF-THEN rules. One rule recommends (expresses the desire) turning away from a close obstacle and the other suggests moving straight forward if the obstacle is at a safe distance. Using standard fuzzy inference, e.g., max-prod in this example, the rules are combined into a multivalued output that encodes the (grade of) desirability of each action from the behavior's point of view. Finite state machines, have been shown to be very efficient for synthesizing behaviors with fixed action patterns and provide a formal approach to behavior encoding.[1,8] This scheme of behavior development is used extensively in CAMPOUT.

3.1.3 Behavior coordination mechanisms

In order to compose higher-level behaviors, CAMPOUT provides a complementary set of coordination mechanisms that can be used for action sequencing, conflict resolution, priority-based behavior invocation, and context-dependant behavior invocation. Specifically, CAMPOUT provides behavior arbitration mechanisms including finite-state machines, and subsumption-style arbitration. Further, it provides command fusion mechanisms using multivalued logic and multiple objective decision making approaches.

Behaviors that compete for the control of the robot must be coordinated to resolve potential conflicts. Fuzzy behavior coordination is performed by combining the fuzzy outputs of the behaviors using an appropriate operator such as a triangular conorm (which corresponds to the fuzzy set union), e.g., the max operator. Then defuzzification (e.g., center of gravity, COG) is used to select a final crisp action ultimately used for control.[14] This scheme should lead to the selection of an action that somehow represents the consensus among the behaviors and thus comprises the action that best satisfies the decision objectives that they encode. However, in situations where there is conflict between the active behaviors, this approach leads to inappropriate results. One approach to dealing with conflicts is based on *context-dependent blending,*[14] where context dependence is encoded using a number of fuzzy meta rules such as:

IF obstacle is close THEN avoid collisions
IF NOT (obstacle is close) THEN follow target

If the robot is close to an obstacle, then the contribution from the collision avoidance will be higher than the contribution from target following. This corresponds to weighted decision making, where the weight values are determined by the truth values of the rule antecedents and is used to scale the behaviors' fuzzy outputs accordingly. Another approach, used in the Saphira architecture,[7] is to resolve conflicts by allowing higher priority behaviors to suppress/dominate behaviors with lower priorities. These approaches provide partial solutions to resolving conflict between behaviors in specific situations and systems.

Defuzzification can also produce inappropriate results since it selects an action that does not have support from any of the behaviors.[15] In summary, due to the unclear semantics and unpredictable characteristics of fuzzy behavior coordination, alternative approaches should be considered. One alternative approach, that avoids the common pitfalls of fuzzy behavior coordination, is multiple objective behavior coordination (MOBC) which fits well into the framework of fuzzy behavior coordination and can easily replace standard fuzzy inference and defuzzification techniques.

3.1.4 Communication infrastructure

In order to facilitate a group of robots to coordinate their activities and cooperate towards the accomplishment of a common task they may be required to communicate to share resources (e.g., sensors or actuators), exchange information (e.g., state, percepts), synchronize their activities etc. CAMPOUT provides a broad set of facilities to foster such collaborative effort by offering a communications infrastructure. The current implementation of communications in CAMPOUT are provided using UNIX-style sockets. Another approach would be to base the communications on some general-purpose message-passing package such as MPI. However, such generality comes at significant overhead cost in efficiency which we intend to avoid for the types of applications that CAMPOUT is designed for. The communications facilities consist of the following core functions:

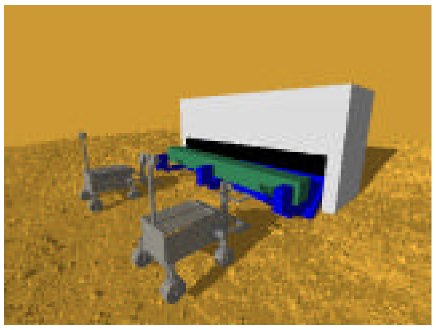
- **Synchronization:** two main functions Signal (destination, sig) and Wait (source, sig) can be used to send and wait for a signal to and from a given robot. This pair constitutes the facilities for synchronizing the activities of robots and/or behaviors.
- **Data exchange:** SendEvent (destination, event) and GetEvent (source, event) can be used to send and receive an event structure to and from a particular robot. The event structure can contain arbitrary data packages as contracted between the sender (source) and receiver (destination). For instance, it can be used to transmit a percept or raw sensor data from one robot to the other etc. E.g., robot 2 will be able to have a behavior that is being fed by the position of robot 1 (to, e.g., follow it).

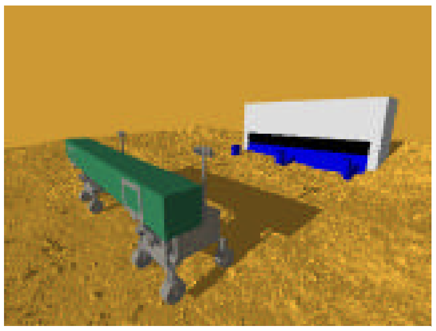
- **Behavior exchange:** SendObjective (destination, objective) and GetObjective (source, objective) can be used to send and receive objective functions (multivalued behavior outputs) to and from a robot. Using these set of functions one can form a network of behaviors across a distributed group of robots.

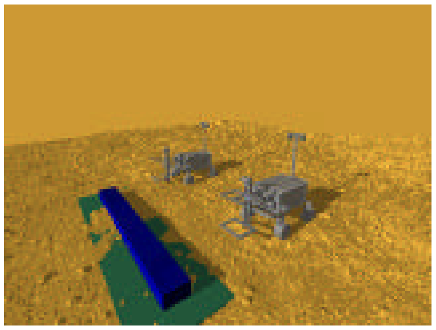
These core set of communications facilities (and other convenience functions) support distributed sharing of resources such as sensors and state as well as provide the necessary tools to form a network of behaviors spanning a group of physically distributed (but informationally connected) robots. State of one robot (e.g., sensor readings or output from a behavior) can be used to affect/determine the behavior of another robot. All these facilities are showcased in the following coordinated transport task.

4. EXPERIMENTAL STUDIES

We have selected a photovoltaic (PV) tent deployment scenario as our experimental testbed for CAMPOUT. A study was done on the viability of a PV tent array for the power needs of a human habitat on Mars.[2] Each PV tent is capable of generating 4 kW under Martian conditions, so an array of 25 tents will produce enough to meet the requirement of 100 kW. The individual


1. Unload container from Container Storage Unit (CSU)


2. Traverse to deployment site


3. Position and open container

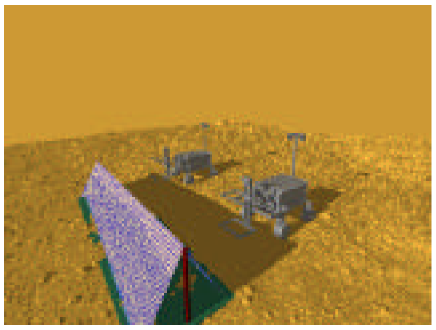

4. Deploy PV tent

Figure 2. Four step process for deploying a PV container. The storage container is 5 m in length and is initially stored at a centralized cache site about 50-100 m from the deployment zone.

containers of the PV tent elements are 5 m in length, so it would be difficult for a single mobile platform to manipulate and transport to a deployment site. A four step process for the deployment of a single PV tent by two rovers is shown in **Figure 3**. Our studies this year have concentrated on Step 2, the traverse to the deployment site. We have retrofitted two of our Sample Return Rovers (SRR&SRR2K) with a gimbal mounted on a cross-brace between the shoulders. The gimbal is not actuated but is

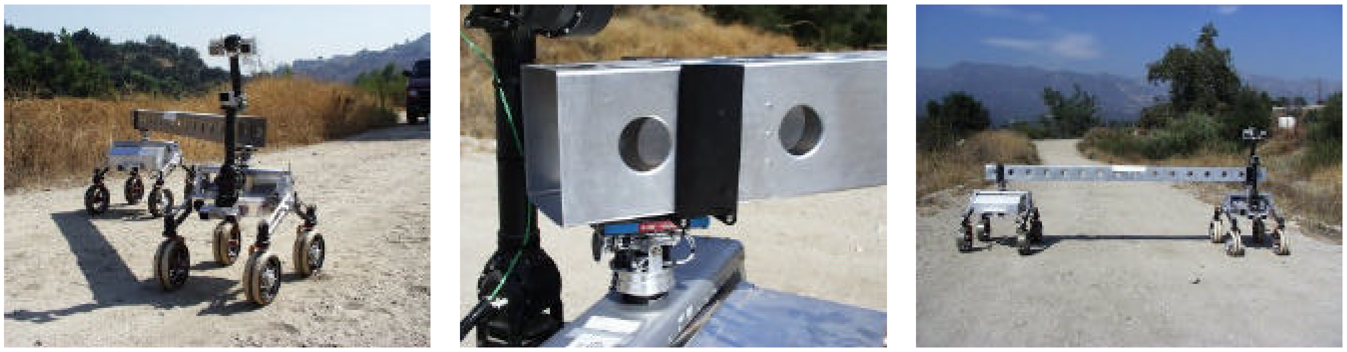


Figure 4. Transport of an extended container by two rovers in the arroyo at JPL. Left: Rovers in a column (offset) transport formation; Middle: Closeup of instrumented gimbal on one of the rovers; Right: Rovers in a row (side-by-side) formation.

fully instrumented with 3DOF force sensors and pots. The gimbal arrangement and two of the coordinated transport formations are shown in **Figure 4**.

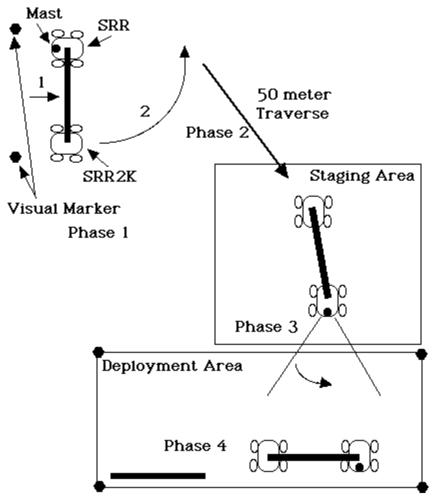


Figure 5. Four phase sequence for transport step in the PV tent deployment scenario. See text for details of steps.

We have developed a finite state machine (FSM) description of the transport phase to emulate the planning level in CAMPOUT, since our first year task is not developing a planner. The four phase sequence for transport is shown in **Figure 5**. The four phases are: clear the container storage unit and assume the column transport formation, traverse to the staging area, survey the deployment area for a clear site, and traverse to the deployment site and align the container.

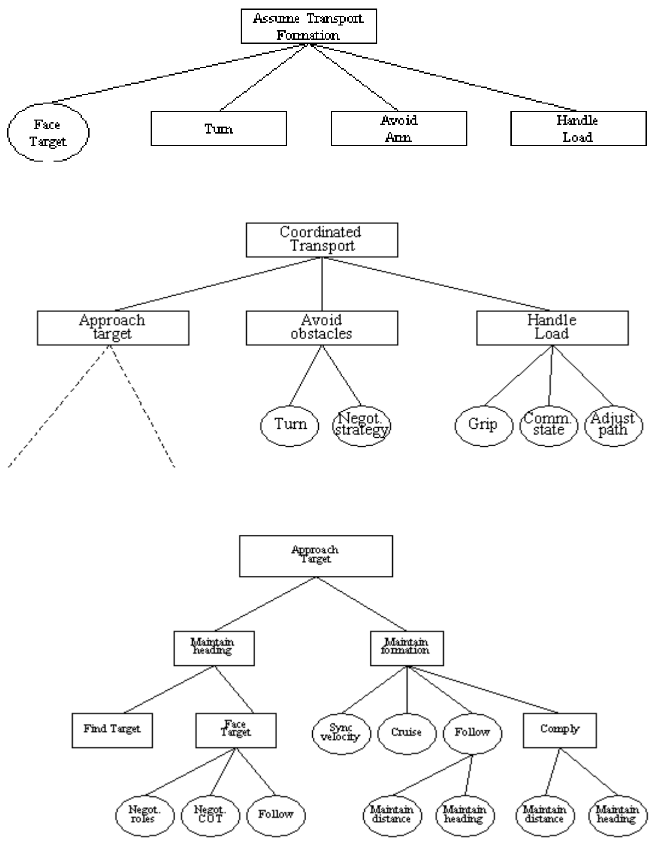


Figure 6. Assume Transport Formation and Coordinated Transport group behaviors used under CAMPOUT for the execution of the sequence shown in Figure 5.

There are two main group behaviors used in CAMPOUT for control: *Assume Transport Formation* and *Coordinated Transport*. The *Assume Transport Formation* group behavior is used in Phases 1 and 4, and the *Coordinated Transport* group behavior is used in Phases 2 and 4. These two behaviors are implemented under CAMPOUT using the behavior fusion framework described in Section 3. They are shown in **Figure 6**, where the main group behavior is shown at the root of the hierarchy. Key to coordinated transport is the notion of compliance by implicit communication through the shared container. The *Comply* group behavior is designed to minimize the amount of explicit communication and only uses point-to-point communication for syncing before movement. We use a target finding algorithm based on color to localize the rovers for heading adjustments during the traverse step in the sequence. The algorithm extracts the corner features of a target and returns the pose and orientation of the target relative to the rovers. This algorithm is used in the *Find Target* group behavior, which then is fused with the *Face Target* group behavior in order to *Maintain Heading*.

We are currently running trials in the arroyo at JPL in order to verify the repeatability, accuracy, and suitability of the CAMPOUT approach to the coordination of multiple rovers for transport of an extended container.

5. DISCUSSION

We have presented a control architecture called CAMPOUT for the system level coordination of multiple mobile robots. The design is three-layer, with a behavior-based middle layer. The behavior hierarchy is built on earlier work by Pirjanian.[11,12] It uses a multiple objective decision making paradigm in order to select actions that are satisficing in the Pareto optimal sense. The lowest level in CAMPOUT is built using legacy device drivers from previous rover tasks such as SRR and FIDO. During the next fiscal year we will concentrate on the development of the grasp and manipulate behaviors that are necessary for the first, third and fourth steps in the PV tent deployment scenario.

ACKNOWLEDGEMENTS

This work was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration

REFERENCES

1. R. A. Brooks, "A robust layered control system for a mobile robot," *IEEE Journal of Robotics and Automation*, **2**(1), 14-23, 1986.
2. A. Colozza, "Design and optimization of a self-deploying PV tent array," NASA CR187119, Lewis research Center Group, Book Park, OH, 1991.
3. E. Gat, "Three-layer Architectures," in *Artificial Intelligence and Mobile Robots*, eds. D. Kortenkamp, R. P. Bonasso, and R. Murphy, MIT Press, Cambridge, MA, 1998.
4. T. L. Huntsberger and J. Rose, "BISMARC: A Biologically Inspired System for Map-based Autonomous Rover Control," *Neural Networks*, **11**(7/8), 1497-1510, 1998.
5. T. L. Huntsberger, M. Mataric, P. Pirjanian, "Action selection within the context of a robotic colony," in *Proc. SPIE Symposium on Sensor Fusion and Decentralized Control in Robotic Systems II*, Vol. 3839, Boston, MA, Sept. 1999.
6. T. L. Huntsberger, H. Aghazarian, E. Baumgartner, P. S. Schenker, "Behavior-based control systems for planetary autonomous robot outposts," in *Proc. Aerospace 2000*, Albuquerque, NM, 2000.
7. K. Konolige and K. Myers, The Saphira Architecture for Autonomous Mobile Robots," in *Artificial Intelligence and Mobile Robots*, eds. D. Kortenkamp, R. P. Bonasso, and R. Murphy, MIT Press, Cambridge, MA, 1998.
8. J. Kosecká, H. I. Christensen, and R. Bajcsy, "Experiments in behavior composition," *Robotics and Autonomous Systems*, **19**, 287-298, 1997.
9. L. Parker, "ALLIANCE: An Architecture for Fault Tolerant Multi-Robot Cooperation," ORNL TM12920, Oak Ridge National Laboratory, Oak Ridge, TN, 1995.
10. P. Pirjanian, "Multiple Objective Behavior-Based Control," to appear in *Journal of Robotics and Autonomous Systems, Special Issue*, 2000.
11. P. Pirjanian, "*Multiple Objective Action Selection & Behavior Fusion using Voting*", Ph.D.-thesis, Faculty of Technical Sciences, Aalborg University Denmark. April 1998.

12. P. Pirjanian and M. Mataric, "Multiple Objective vs. Fuzzy Behavior Coordination," to appear in *Lecture Notes in Computer Science on Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, eds. D. Drainkov and A. Saffiotti, 2000.
13. J. K. Rosenblatt, "*DAMN: A Distributed Architecture for Mobile Navigation,*", PhD Thesis, The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 1997.
14. A. Saffioti, E. Ruspini, and K. Konolige, "Using Fuzzy Logic for Mobile Robot Control," *Handbook of Fuzzy Sets and Possibility Theory*, Kluwer Academic, 1997.
15. J. Yen and N. Pfluger, "A fuzzy logic based extension to Payton and Rosenblatt's command fusion method for mobile robot navigation," IEEE Trans on Systems, Man, & Cybernetics, **25**(6), 971-978, 1995.