# Development of CAMPOUT and its further applications to planetary rover operations: a multi-robot control architecture

Paolo Pirjanian*, Terry L. Huntsberger, Paul S. Schenker,
Jet Propulsion Laboratory, California Institute of Technology

## ABSTRACT

In this paper, we describe an architecture for the development of autonomy software for multi-robot distributed control and collective estimation. CAMPOUT, the Control Architecture for Multi-Robot Planetary OUTposts, provides communication facilities for sharing of state information across robots and it uses a behavior network for representation and execution of group activities as well as the activities of a single robot. In our research, we have shown that CAMPOUT provides a level of abstraction that enables us to develop multi-robot software in a manner much similar to what we use for single robot software development. We showcase the main architectural components by describing two multi-robot tasks for planetary construction and collective cliff-descent. For both tasks, we show how behavior networks can be used to describe group activities and how publish/subscribe and other communication mechanisms can be used to share state information across multiple robots.

**Keywords:** control architecture, multi-robot systems, tight coordination, planetary rover operations

## 1 INRODUCTION

Future planetary and space operations may require employment of multiple robots, which must cooperate to achieve a common task that is either impossible or impractical to accomplish using a single robot. These tasks may range from cooperative handling, transportation, assembly, and maintenance of large structures to collective surface exploration and mapping [1]. In our work to date, we have focused on two areas of multi-robot control issues. The first area pertains to cooperative grasp, handling, and transportation of large containers using Robotic Work Crews (RWC) [2]. The second area is the use of multi-robot systems to develop an aggressive mobility system for All Terrain Exploration (ATE) [3]. The ATE concept consists of a tethered 'cliff-bot' that is assisted by two 'anchor-bots' to reach a desired location on a cliff side, where high-value science targets can be accessed.

The advantages of using multiple versus a single robot for such tasks are many, including redundancy and hence fault-tolerance to single robot failures, complementary capabilities provided by a heterogeneous group of robots, parallel task execution, and increased work for the same launch mass [4]. But these benefits often come at the cost of an increased complexity of software and techniques for control and coordination of multiple robots, the need for inter-robot communications and more. For instance, a distinctive characteristic of ATE and RWC is the requirement for explicit coordination of the activities of distributed robotic entities in a tightly coupled fashion. Both systems consist of robots that are physically connected (with a container or tethers) so that the actions of a single robot directly influence the (physical) state of the others. E.g., in ATE, the velocity of the cliff-bot must be tightly coupled with the velocities at which the anchor-bots pay out the tethers. A significant mismatch between their velocities can either yank the cliff-bot or send it falling down the cliff, etc. Thus a major thrust of our work has been to develop a methodology along with a software architecture called CAMPOUT that supports the development of multi-robot systems. CAMPOUT (Control

* Paolo.Pirjanian@jpl.nasa.gov; phone 1 818 354-3169; fax 1 818)393-3254; http://robotics/people/paolop; Planetary Robotics Lab, Jet Propulsions Laboratory, California Institute of Technology 4800 Oak Grove Drive, Pasadena, CA 91109, USA

Architecture for Multi-robot Planetary OUTposts) essentially provides infrastructure for collective state estimation and distributed control. Robotics is a highly multidisciplinary field, and requires efficient integration of many components (perception, mapping, localization, control, learning, etc.) that use different representations, frameworks, and paradigms (classical control theory, AI planners, estimation theory, data fusion, computer vision, utility theory, decision theory, fuzzy logic, multiple objective decision making etc.). CAMPOUT provides the infrastructure, tools, and guidelines that consolidate a number of diverse techniques to allow the efficient use and integration of these components for meaningful interaction and operation. This is facilitated through a few elementary architectural mechanisms for *behavior representation*, *behavior composition*, and *group coordination*, and the interfaces between these. These mechanisms and a framework with guidelines for describing systems define the core of CAMPOUT. CAMPOUT is extensible and scales freely with regard to behavioral mechanisms and protocols that it can host and fuse, re-mappable inter-robot communications it can support, and the overall ability to functionally integrate heterogeneous, multi-purpose platforms.
In this paper, we describe the fundamental philosophy of CAMPOUT, the main underlying architectural mechanisms, and its application to two planetary rover tasks of coordinated transportation and collective cliff-descent. We conclude with lessons learned and directions for future work.

## 2 CAMPOUT

CAMPOUT consists of a number of key mechanisms and architectural components that facilitate development of multi-robot systems for cooperative and coordinated activities. These include the following:

1. **Modular task decomposition:** Following a behavior-based methodology, CAMPOUT provides fundamental building blocks for describing a system in terms of task-achieving modules known as a behavior-producing module or a behavior, for short. While a behavior provides a convenient and efficient architectural substrate to encapsulate perception and action, it is its interactions with other behavior-producing modules that generate the final behavior of the system. In its current implementation, task decomposition is done by hand and encoded in a script/plan, which is then executed by the agents. We are currently working towards extending CAMPOUT with automated planning of joint team activities.
2. **Behavior coordination mechanisms:** A system's behavior is described as a network of behaviors that interact with each other and with the environment through sensors and effectors. The behavior interactions are regulated through behavior coordination mechanisms (BCMs). The BCMs are used to restrict and control the behavioral interactions so that the system can operate according to its specifications. In other words, the BCMs are used to ensure that the behaviors interact in a desired and consistent manner.
3. **Group coordination:** CAMPOUT uses the same task decomposition scheme and representation to describe group activities, the deference being that the nodes (behaviors) of the network are distributed across the group of robots and connected through implicit (supported by extraprioceptive sensing) or explicit (radio) communication.
4. **Communications infrastructure:** Relying on sensing for communication is not a feasible solution because sensing can be unreliable, the sensory envelope is often more range-limited than radio communication, and it requires more computation for processing than is usually the case with radio communication. CAMPOUT provides a software infrastructure that allows transparent inter-robot communication, which enables the robots to share state information, sensors, actuators, etc. The communications infrastructure allows a behavior network to be seamlessly distributed across a network of robots.

### 2.1 Modular task decomposition

In CAMPOUT, the control and autonomy software is described as a network of modules known as behaviors. Essentially, a behavior is a specialized computational unit that generates a mapping from perception to action in order to achieve or maintain a goal such as obstacle avoidance, stable balance, compliance etc. Behaviors exploit specific knowledge about their task and the environment to achieve a remarkably efficient implementation of their control policy, while requiring limited computation [5].

In our architectural methodology we formalize a behavior, $b$, as a mapping, $b: P^* \times X \rightarrow [0; 1]$, that relates each percept sequence $p \in P^*$ and action $x \in X$ pair, $(p, x)$, 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 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 CAMPOUT, this representation is implemented using an N-dimensional array, which contains the desirability values recommended by a behavior.

### 2.2 Behavior coordination mechanisms

Behavior-producing modules or behaviors are the building blocks used for describing a system's behavior including its perceptual capabilities, decisions, actions, and reactions. Using behavior coordination mechanisms (BCMs) we combine a group of behaviors to achieve higher-level goals. A major issue in the design of behavior-based control systems is the formulation of effective mechanisms for coordination of the behaviors' activities into strategies for rational and coherent behavior. BCMs coordinate the activities of lower-level behaviors within the context of a high-level behavior's task and objective. An explicit design goal of CAMPOUT has been to support not one but an arbitrary number of BCMs. BCMs can be divided into two main classes: arbitration and command fusion. For a detailed overview, discussion, and comparison of behavior coordination mechanisms see [6].
If behaviors are viewed as operands, then BCMs are the operators used to combine behaviors into higher-level behaviors.
*Arbitration mechanisms* select one behavior, from a group of competing ones, and give it ultimate control of the system (the robot) until the next selection cycle. This approach is suitable for arbitrating between the set of active behaviors in accord with the system's changing objectives and requirements under varying conditions. It can focus the use of scarce system resources (sensory, computational, etc.) on tasks that are considered to be relevant. CAMPOUT implements the following arbitration mechanisms:

- Priority-based arbitration: which is a subsumptive-style, priority-based arbitration mechanism, where behaviors with higher priorities are allowed to suppress the output of behaviors with lower priorities.
- State-based arbitration: which is based on the Discrete Event Systems (DES) formalism [7], and is suitable for behvior sequencing.

*Command fusion* mechanisms combine recommendations from multiple behaviors to form a control action that represents their consensus. This approach provides for a coordination scheme that allows all behaviors to simultaneously contribute to the control of the system in a cooperative rather than a competitive manner, which makes them suitable for tightly coupled tasks that require spatio-temporal coordination of activities. CAMPOUT provides a number of complementary mechanisms for fusion:

- Voting techniques interpret the output of each behavior as votes for or against possible actions and the action with the maximum weighted sum of votes is selected. CAMPOUT implements a DAMN-style [8] voting algorithm based on BISMARC [9]
- Fuzzy command fusion mechanisms (see [10-11]) use fuzzy logic and inference to formalize the action selection processes. In addition, fuzzy approaches enable a new class of coordination mechanisms denoted context-dependent blending, introduced to robotics by Saffiotti, Ruspini, and Konolige in [10], which allow for weighted combination of behaviors. The implementation in CAMPOUT follows that described in [10].
- Multiple objective behavior fusion provides a formal approach to behavior coordination based on multiple objective decision theory [12]. Action selection consists of selecting an action that makes the best trade-off between the task objectives and which satisfies the behavioral objectives as much as possible.

In short, CAMPOUT supports both arbitration and command fusion, both of which are useful for describing complex activities. Arbitration is useful for describing sequences of actions/behaviors with conditionals. Command fusion is useful for cooperative and parallel execution of activities.

### 2.3 Group coordination

In order to cooperate and collectively contribute to a common task, the robots must cooperate and coordinate their activities. Behavior coordination is basically concerned with resolving or managing conflicts between mutually exclusive alternatives and between behavioral objectives. Group coordination in CAMPOUT (see Figure 1) is treated as *the coordination of multiple distributed behaviors, across a network of robots*, where more than one decision maker is present.

Behavior coordination in multi-robot systems has received relatively little attention. One approach proposed in [13] uses inhibition and suppression across a network of heterogeneous robots augmented with motivational behaviors that can trigger behavior invocation based on some internal parameters that measure progress. A similar approach was proposed in the AYLLU architecture [14], which uses port arbitration as the main mechanism for multi-robot behavior coordination. Both of these approaches can be viewed as the extension of subsumptive-style arbitration to multi-robot coordination.

Recently, work in progress is investigating the extension of the 3T architecture to multi-robot coordination [15]. The above approaches as well as most multi-robot architectures including ACTRESS, GOFER, SWARM [16, 17, 18] invariably have two things in common. First, multi-robot coordination mechanisms are limited to only one approach, and second this approach mostly tends to be arbitration rather than a command fusion scheme. Arbitration limits cooperation to execution of tasks that are either independent/parallel or loosely coupled, turn-taking tasks. We maintain that arbitration and command fusion mechanisms are complementary and a system implementation will typically make use of both.

The philosophy in CAMPOUT is that the architecture should support both arbitration and fusion. Further, we favor mechanisms that are based on formal theories to support a sound approach to description and validation of system behavior. This is an important characteristic of CAMPOUT, since it enables us to provide certain performance guarantees. We have chosen to support, but not limit the architecture to, arbitration using ALLIANCE and AYLLU's subsumptive-style and the discrete event system. Additionally, multi-objective behavior coordination is supported by CAMPOUT for command fusion [19].

The view taken in CAMPOUT is thus that multi-robot cooperation arises from coordination of multiple behaviors that reside on not one but a group of robots (see Figure 1). In order to support this view, BCMs must be extended to support multi-robot coordination. In [19] the multiple objective behavior coordination approach was extended to multi-robot applications.

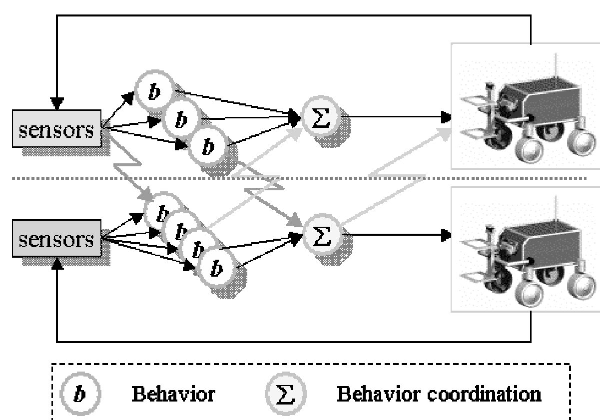


Figure 1 Networked robotics and resource sharing elements of CAMPOUT that enable definition of group coordination behaviors.

### 2.4 Communications infrastructure

CAMPOUT provides the infrastructure by which the distributed behaviors can interact through communication. The behaviors and hence the robots can communicate implicitly by interaction through the environment or explicitly using sensory feedback or explicit communication. The first two approaches, interaction through the environment and sensory feedback, do not require any explicit form for architectural support as long as the robots have the necessary sensing capabilities to facilitate such interaction. These forms for interaction can be difficult and often computationally demanding, which is why most multi-robot systems resort to a form of explicit communication. CAMPOUT provides a rich and efficient infrastructure for explicit communication to facilitate multi-robot cooperation. Using this infrastructure, behaviors on one robot can interact with behaviors on other robots. In general the infrastructure defines a network of resources that can be shared among the robots. These resources include behaviors, sensors, and actuators. Thus a behavior on one robot can be driven by a sensor on another robot or even contribute to the control of a different robot. This idea is depicted in Figure 1, where behavior composition can be achieved across several robots.

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. The primitive and 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. CAMPOUT provides a broad set of facilities to foster such collaborative effort by offering a communications infrastructure. The current implementation of communications in CAMPOUT is 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) are 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) are 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) are used to send and receive objective functions (multivalued behavior outputs) to and from a robot. This function encodes a multi-valued output into a message and transmits it to the destination where it is decoded into a multi-valued representation.
- **Publish/Subscribe:** Probably CAMPOUT's most useful/powerful facility for inter-robot communications is based on a publish/subscribe service where one robot can request any value on any robot. Basically, a robot can request subscription to a given attribute (variable, sensor, etc.) on a given robot at a given frequency. Once a subscription has been established using Subscribe(attribute, robot_ID, frequency), the subscriber can locally access the requested attribute.

These core set of communications facilities (and other convenience functions) support distributed sharing of resources such as sensors and state, as well as providing the necessary tools to form a network of behaviors spanning a group of physically distributed (but informationally connected) robots. The 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 applications for multi-robot planetary operations.

## 3 APPLICATION OF CAMPOUT TO ROVER COORDINATION

In this section, we describe how the architectural components of CAMPOUT can be used to describe tasks that represent group activities so that they can be executed by a distributed set of robots. We use two multi-robot applications which require close coordination between the activities of the robots and where most of the architectural facilities of CAMPOUT are used to accomplish this. Both tasks are characterized by their strict requirements for tight coordination because of the physical interactions between the robots.

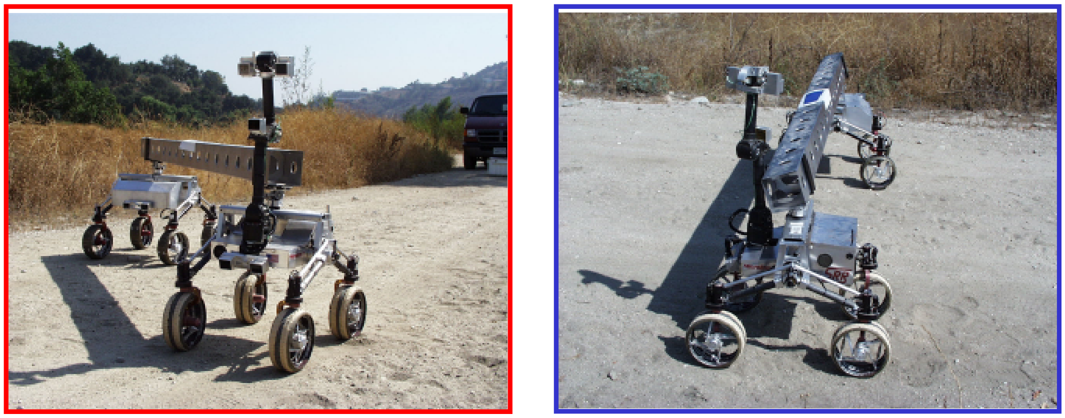

Figure 2 The Robotic Work Crews task showing transport of an extended object through tightly-coupled coordination of two robots in uneven terrain. (left) Column (diagonal) formation for long traverse. (right) Row formation for precision placement.

### 3.1 Coordinated Transportation

The first task consists of using two robots to transport a large, extended container over 10s of meters in rough terrain. The coordinated transport task in open, uneven terrain requires a tightly-coupled, close coordination of the activities of the two robots. This is accomplished by some 20 behaviors, organized in a hierarchy where lower-level behaviors are combined into higher-level behaviors (reported at last year's conference [20]). The behaviors form a network of modules distributed across the two robots and linked by means of communication. Key to the group behaviors is the notion of implicit communication through the shared container (payload carried by the two rovers) and explicit communication through communication facilities for distributed resource sharing. These behaviors are implemented and tied together using the mechanisms provided by CAMPOUT.

Coordinated transportation consists of four phase for grabbing and lifting the container, clearing the container storage unit and assuming a column transport formation facing the deployment site, traversing to the deployment site, and finally deploying and aligning the container. We have developed a finite state machine (FSM) description of the transport phases to emulate the planning level in CAMPOUT, since our first year task is not developing a planner. Here we only describe the behaviors for accomplishing the assume formation phase, where the robots find the deployment site and turn to face it in a column formation. The *Assume Formation* group behavior is invoked each time the heading error relative to the target is larger than a preset threshold. A behavior denoted *Find Target* uses a visual target finding algorithm based on color-segmentation to localize the rovers for heading adjustments during the traverse step in the sequence. Then an *Approach Target* group behavior is used to safely carry the container towards the deployment area. *Approach Target* uses a number of compliance behaviors to assure safe handling of the container during turn and carry operations by constraining and adjusting the movements of the two rovers.

A desired formation is defined by the relative angle between the two robots, $\alpha$, and the relative angle towards the target, $\gamma$ (see Figure 3). The *Find Target* behavior provides the angle to the target then a *Turn* group behavior reconfigures the formation to a desired one. Two constraints make this a challenging task. First, transformation between the current and target formations must ensure that the container is handled safely, i.e., the distance between the robots, *d*, should always remain within some tolerance margin, $d_{min} \leq d \leq d_{max}$, determined by the distance between the grip points of the rovers, L (250 cm), and the longitudinal translation in the gimbal $T_{gimbal}$ (±2 cm). I.e., $L-2T_{gimbal} \leq d \leq L+2T_{gimbal}$, which implies that the distance between the two rovers should be maintained within a margin of 8cm ($4T_{gimbal}$). A set of compliance behaviors monitor the state of the load and constrain the movement of the rovers to guarantee this requirement. Second, it is required that the container does not collide with the mast on the lead rover (see Figures 2 and 3), which could lead to damaging the mast, the gripper/gimbal, or the container, and/or dropping the container. The shaded area around the lead rover, in Figure 3, indicates the safety zone (-35° to +35°) where the container beam cannot enter because it will then collide with the mast.

### 3.1.1 Centralized motion planning

One approach is to formulate this problem as a constraint satisfaction search problem, with the following description:

- Configuration space is the possible states of the formation defined by ($\theta_L$, $\theta_F$, $\alpha$), where $\theta_L$ and $\theta_F$ are the absolute heading angle of the leader and follower respectively and $\alpha$ is the formation angle. The configuration space will exclude states where the beam intersects with the safety zone.
- The goal configuration is ($\theta_{target}$, $\theta_{target}$, $\alpha_{formation}$), where $\theta_{target}$ is the heading angle to the target and $\alpha_{formation}$ is the desired formation angle.
- The operators for search correspond to the actions that the robots can perform and include: TurnInPlace($\phi$) and Ackerman($\beta$) for each of the rovers. I.e., four types of operators exist, two for each rover. However, due to the strategy we have chosen for the compliance behaviors (see next section) we have constrained the motion of the leader to only *TurnInPlace* movements. Hence only three types of operators exist. *LeaderTurnInPlace*, *FollowerTurnInPlace*, and *FollowerAckerman*. Ackerman causes the follower to pivot around the lead rover.

Using some search algorithm, a centralized module/planner can generate the sequence of actions (operators) that bring the system to the goal configuration. The execution of the sequence must command and synchronize the motion of each of the robots. A main advantage of this approach is that it is complete and it can generate optimal (e.g., shortest sequence of movements) solutions. This is, however, outweighed by its many disadvantages including it's polynomial computational complexity due to a three-dimensional configuration space and a large branching factor determined by the number of operators. Further, this approach requires a centralized module/planner, which generates commands to control each of the robots and monitors their state during execution. While this centralized approach could easily be implemented in CAMPOUT, we preferred a decentralized approach that does not suffer from many of the disadvantages of its centralized counterpart.

### 3.1.2 Decentralized motion generation and coordination

By a careful inspection of results generated by the centralized search method described above, we observed a pattern in the action sequences, which was used to design a decentralized solution. The optimal solution generated by the search algorithm had the following pattern:

1. The lead rover turns as far as possible until either $\theta_{target}$ is reached or it cannot move further due to the safety zone constraint. It turns in the direction that minimizes the difference between its current heading angle and the desired heading, $\theta_{target}$.
2. The follower pivots until either $\alpha_{formation}$ is reached or it cannot move further due to the safety zone constraint. It pivots in the direction that minimizes the formation angle error.

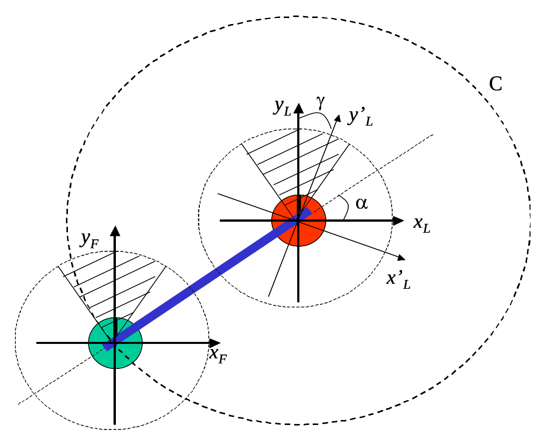

Figure 3 Formation between the two robots with follower on left and leader on right. The formation is defined by the angle $\alpha$ between the two robots. Desired heading is given by the relative heading angle $\gamma$**.** The shaded area on the lead rover is a safety zone where the container beam should not enter to prevent collisions with its mast.

This sequence will alternate the two robots until the goal configuration is reached. Note that once one rover moves, it also frees the other rover from being constrained by the safety zone. In this way, incremental progress is made towards the goal configuration. Using this observation we constructed a distributed solution to the problem where the lead and follower rovers alternate in turning in place and pivoting until the goal configuration is reached:

- Lead rover performs: TurnInPlace ($\max(\min(\alpha_{left}, \theta_{left}), \min(\alpha_{right}, \theta_{right}))$),

where $\alpha_{left}$ (-35 degrees) and $\alpha_{right}$ (35 degrees) are the limit angles of the safety zone and $\theta_{left}$ and $\theta_{right}$ are the relative angle to $\theta_{target}$ in clock-wise and counter-clock-wise direction respectively.

- Follower rover performs: Ackerman ($\max(\min(\alpha_{left}, \alpha_{formation}), \min(\alpha_{right}, \alpha_{formation}))$),

where $\alpha_{left}$ and $\alpha_{right}$ are the limit angles of the safety zone and $\alpha_{formation}$ is the desired formation angle.

It can be shown that this strategy is complete, i.e., it will reach a solution if one is found. However, the strategy does not guarantee an optimal solution (minimum steps) although its solutions are typically close to optimal.

The lead and follower rovers need to synchronize their activities for two purposes: 1) termination of formation configuration and 2) turn-taking between leader turning in place and follower pivoting. The communication behavior *Signal* is used to perform this synchronization. The termination condition is when $\theta_L = \theta_{target}$ and $\alpha = \alpha_{formation}$. The lead rover can measure $\alpha$ locally from the gimbal pots and its heading $\theta_L$ based on visual feedback and position encoders. The follower can access $\theta_L$ using communication constructs of CAMPOUT and it can measure $\alpha$ locally. This behavior is implemented using a discrete event system or finite state machine action selection mechanism as shown in Figure 4. Note that the *Turn* behavior in the figure is either a *TurnInPlace* or an *Ackerman* for the leader and follower, respectively.

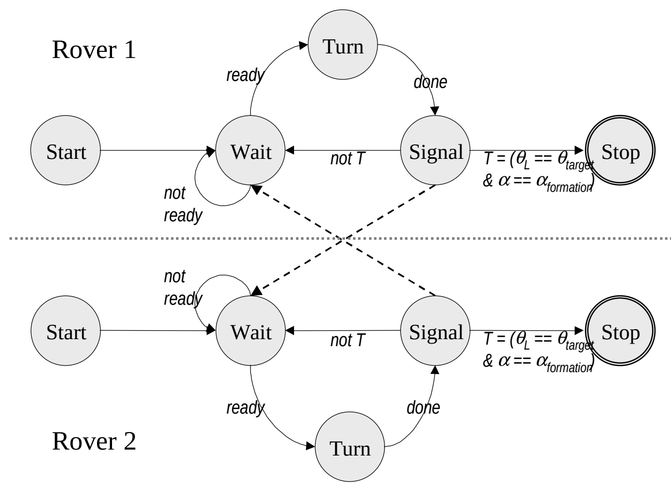


Figure 4 Distributed behavior network, used for the group Assume Formation task. The arrows represent events that cause transitions, and the dashed curves represent events caused by explicit communication of signals between remote components/behaviors.

The group turn behavior has two separate, distributed pieces: one that runs on the leader and one on the follower (see Figure 4). Each of these consists of the composition of a set of primitive behaviors on each of the rovers, which use local sensory feedback for control. These distributed pieces of the group turn behavior are synchronized in part using explicit communication for invoking either the leader part or the follower part. One rover turns (using the *Turn* behavior) until it is done (i.e., cannot turn further due to the safety constraint) then hands the token to the other rover by a signal. The *Wait* behavior in each of the rovers consists of a number of behaviors including compliance behaviors. For example, when the other rover starts moving/turning, the waiting rover monitors the state of the load (through the sensors of the gimbal) and then triggers a compliance behavior to assure that the container is handled safely in accordance with the distance constraint described above. This is accomplished by crabbing in the direction of the container in order to center the load (based on pot-meter readings) and to reduce the forces on the gimbals (based on the force-torque sensor readings). The behavior coordination and communication mechanisms provided by CAMPOUT enable a seamless integration and coordination of behaviors across the robots.

### 3.2 Collective Cliff-Descent

The objective of this work has been to develop a distributed mobility system for the cooperative traverse of a cliff-side wall --up to 75$^o$ grade-- where a 'cliff-bot' rappels down a cliff assisted by two semi-mobile railed robotic anchoring stations called 'anchor-bots'. This aggressive mobility platform includes capabilities for cooperative rappelling down a cliff and navigation to a designated way-point on the cliff (see Figure 5). This work enables access to high-value science at high-risk locations, such as escarpments, fissures, breakout channels and cliffs. The system test-bed consists of a tethered ensemble of three robotic entities; the *rappeller* or *cliff-bot*, and two anchoring assistants, *anchor-bots*, that cooperatively direct and safely guide the rappeller to descend to way-points which are on the cliff-side and which are within the workspace defined by the tether lengths and the anchoring points. In this work we are using CAMPOUT to demonstrate elements including collective fused state estimation and distributed controls. More generally, the underlying development of on-board autonomous "behavior-based" controls and state estimation technology will lead to a next generation of highly adaptive, survivable mobile systems for all-terrain exploration and multi-task planetary applications.

### 3.3 Collective way-point navigation

Without coordinated assistance from the anchor-bots, the cliff-bot will not be able to traverse the side of a cliff with slopes of up to 75 degrees, due to tip-over and loss of traction considerations. Most mobility platforms are not able to safely traverse moderate slopes of as low as 35-40 degrees without adaptive CG and traction control. In fact, we have previously developed such techniques for traversing slopes of up to 50 degrees [3].

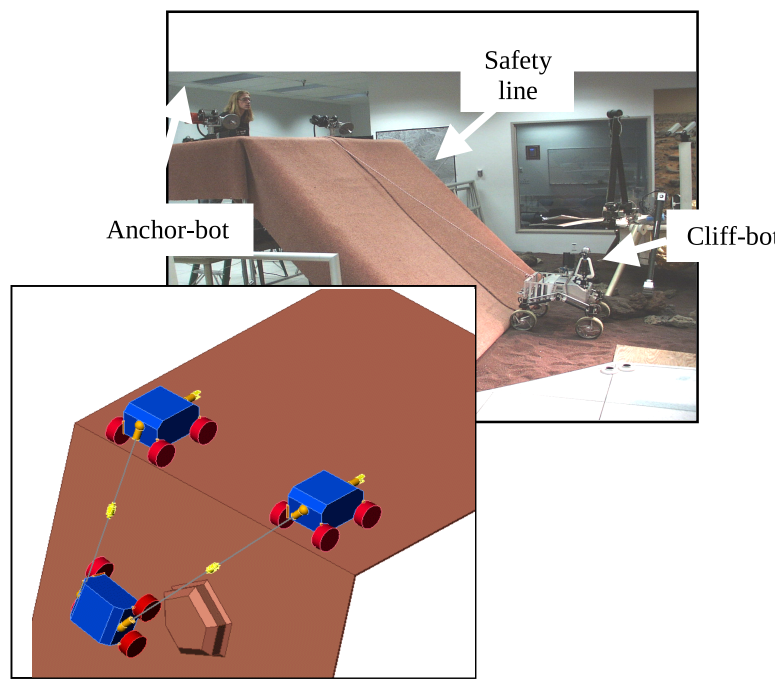


Figure 5 The cliff-bot concept for All Terrain Exploration, illustrated as a CAD drawing (lower-left) and the actual test-bed that we have constructed with the two semi-mobile winch assemblies emulating the anchor-bots (upper-right).

To navigate on a steep cliff-side wall, the cliff-bot and the two anchor-bots are required to tightly coordinate their activities. Each anchor-bot must adjust the velocity of its tether to the velocity of the cliff-bot. The relationship between the velocity of the tethers is derived from the kinematics constraints of the system and is determined by the projection of the cliff-bot's velocity, $\boldsymbol{v}$ onto the directional vector of each tether $\boldsymbol{r}_{left}$ and $\boldsymbol{r}_{right}$. The velocity vector $\boldsymbol{v}$ is easily estimated using the cliff-bot's position encoders and inertial navigation system but has to be shared with the anchor-bots. Also the tether vectors $\boldsymbol{r}_{left}$ and $\boldsymbol{r}_{right}$ are estimated based on sensor data on the cliff-bot, which is instrumented with resolvers to measure the tilt and pan (or pitch and yaw) angles of the attached tethers. So the yaw and pitch angles must be shared with the anchor-bots, which is transparently provided through CAMPOUT's communications infrastructure. Using CAMPOUT, each anchor-bot can access the sensor readings (tether vectors) or fused estimates (e.g., velocity vector) on any robot. To control the tether velocities each anchor-bot is instrumented with one encoder on the driving motor and another on the pay-out mechanism that measures the tether length.

In theory, the coordination of the velocities of anchor-bots with the cliff-bot boils down to a straight-forward projection of velocities onto the tether vectors. However, to realize this capability in practice the three robots are required to perform distributed control and estimation. This requires access to data that is distributed across the robots, which is facilitated by CAMPOUT. Further, there are a number of failure modes that are not handled by this straight-forward approach. One particular failure mode is when the cliff-bot attempts to ascend a steep slope but cannot create enough torque to create a momentum, which implies that its velocity never exceeds $\boldsymbol{0}$. This means that the projected velocities for the winches remain $\boldsymbol{0}$. When this occurs, we would like the anchor-bots to proactively, rather than reactively, assist the cliff-bot to initiate its velocity by hauling it for a short distance. Once the cliff-bot is in motion the anchor-bots will resume adjusting their velocities in a reactive manner. CAMPOUT's task decomposition and distributed task description capabilities facilitate the development of such distributed control strategies using powerful tools to simplify the problem. Many other failure modes exist that again require both collective estimation and distributed control. The architectural facilities provided by CAMPOUT support the development of such complex systems.

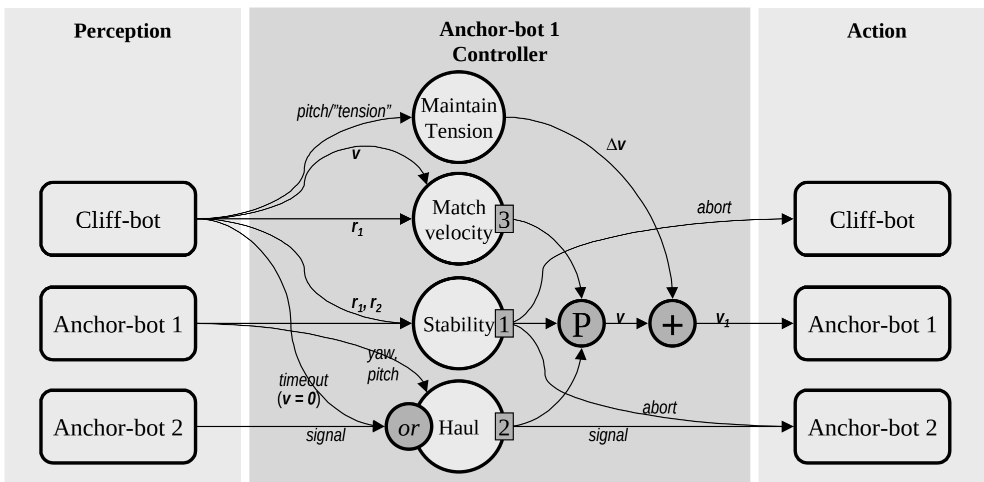


Figure 6 A subset of the behavior network for collective cliff-descent illustrating sub-system for controlling the velocity of anchor-bot 1. The arrows represent data links between local blocks as well as remote components (behaviors, sensors, actuators) thus spanning a behavior network across the team of robots.

The block diagram in Figure 6 shows a portion of the behavior network that is used for controlling the velocity of anchor-bot 1. The portion used for controlling anchor-bot 2 is the same but with different links between the behaviors, sensors, and actuators. The four main behaviors that control the velocity of the tether are *Match Velocity*, *Haul*, *Maintain (tether) Tension*, and *Stability*. *Match Velocity* adjusts the tether velocity based on the cliff-bot's velocity. *Haul* is triggered either when the other anchor-bot is triggered or when the velocity of the cliff-bot remains zero after a time-out period from the time where a navigation command was issued. When *Haul* is triggered, it gives the cliff-bot a "push" (or actually pull) in the right direction to help it initiate motion in the up-hill direction. *Maintain Tension* controls the tether velocity to maintain a safe tension on the tether. *Stability* consists of a number of modules that check various stability requirements (not covered here) and triggers a safe mode where all commands are aborted. These behaviors are in part coordinated using a priority-based arbitration mechanism (denoted with the circle labeled "P") and in part by a summation operator. The boxes with numbers 1, 2, and 3 indicate the priority of each behavior with *Stability* having the highest priority of 1. The links to the behaviors indicate exchange of sensory, perceptual, and other state information across the network of behaviors and robots, as supported by CAMPOUT's communications facilities. In fact, for the behaviors there is no perceivable distinction between local and remote data since all data transfer occurs transparently once the transfer has been initiated using the publish and subscribe methods.

## 4 CONCLUSIONS AND FUTURE WORK

Developing control software for multi-robot systems using the conventional tools used for single-robot systems can become rather tedious and challenging. One challenge stems from the lack of access to the state of a multi-robot system, which is required for decision-making and control. Another challenge is the limitations of conventional representations for the description of group activities for a set of distributed entities with independent computing platforms. A related challenge is need for coordination of the activities of individual robots to accomplish a desired group activity. These are hard challenges for multi-robot, in particular for tasks which require tight coordination of activities and where the robots' actions and states are interdependent.

CAMPOUT provides communication facilities for sharing of state information across robots and it uses a behavior network for representation and execution of group activities in the same way that it represents the activities of a single robot. In our research, we have shown that CAMPOUT almost bridges the gap between multiple robots and provides a level of abstraction that enables us to develop multi-robot software in a manner much similar to what we use for single robot software development.

For future work, we are interested in further bridging this gap by extending CAMPOUT with task planning capabilities and automation of group activity generation. Currently, we use CAMPOUT's facilities to hand-craft the behavior network that represents a group activity. We are investigating approaches to automate this process so that behavior networks can be generated and implemented automatically.

## ACKNOWLEDGEMENTS

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the Cross Enterprise Technology Development Program of the National Aeronautics and Space Administration. The authors would like to thank the many people that have provided valuable contributions to the described work including Chris Leger, Ashitey Trebi-Ollennu, Hari Das, Hrand Aghazarian, Brett Kennedy, Anthony Ganino, Mike Garrett, Sanjay S. Joshi, Lee Magnone, and Erik Mum.

## REFERENCES

1. T.L. Huntsberger, P. Pirjanian, and P.S. Schenker, "Robotic outposts as precursors to a manned Mars habitat," in Proceedings of Space Technology and Applications International Forum (STAIF-2001), Albuquerque, NM, Feb 2001.
2. P.S. Schenker, T.L. Huntsberger, P. Pirjanian, A. Trebi-Ollennu, H. Das, S. Joshi, H. Aghazarian, A.J. Ganino, B.A. Kennedy, and M.S. Garrett, "Robot work crews for planetary outposts: Close cooperation and coordination of multiple robots," in Proceedings of SPIE Symposium on Sensor Fusion and Decentralized Control in Robotic Systems III, Vol. 4196, Sensor Fusion and Decentralized Control in Robotic Systems III (Eds. G. T. McKee and P. S. Schenker), Boston, MA, Nov. 5-8, 2000.
3. P. S. Schenker, P. Pirjanian, B. Balaram, K. S. Ali, A. Trebi-Ollennu, T. L. Huntsberger, H. Aghazarian, B. A. Kennedy and E. T. Baumgartner, Jet Propulsion Laboratory; K. Iagnemma, A. Rzepniewski, and S. Dubowsky, Massachusetts Institute of Technology; P. C. Leger and D. Apostolopoulos, Carnegie Mellon University; G. T. McKee, University of Reading (UK), "Reconfigurable robots for all terrain exploration," in Proc. SPIE Vol. 4196, Sensor Fusion and Decentralized Control in Robotic Systems III (Eds. G. T. McKee and P. S. Schenker), 15 pp., Boston, MA, Nov. 5-8, 2000.
4. B. Wilcox, A. Nasif, and R. Welch, "Implications of Martian Rock Distributions on Rover Scaling," Planetary Society International Conference on Mobile Planetary Robots and Rover Roundup, Santa Monica CA, January 29 - February 1 1997.
5. P. Pirjanian, H.I. Christensen, and J.A. Fayman "Experimental Investigation of Voting Schemes for Fusion of Redundant Purposive Modules", 5th Symposium for Intelligent Robotic Systems, Stockholm, July 1997. pp 131-140.

6. P. Pirjanian, "Behavior Coordination Mechanisms - State-of-the-art", Tech-report IRIS-99-375, Institute for Robotics and Intelligent Systems, School of Engineering, University of Southern California, October, 1999.
7. J. Kosecka and R. Bajcsy, “Discrete Event Systems for autonomous mobile agents,” in Proc. Intelligent Robotic Systems '93 Zakopane, pages 21--31, July 1993.
8. J. Rosenblatt, “The Distributed Architecture for Mobile Navigation,” Journal of Experimental and Theoretical Artificial Intelligence, vol. 9, no. 2/3, pp.339-360, April-September, 1997.
9. T. L. Huntsberger and J. Rose, “BISMARC,” Neural Networks, vol. 11, no. 7/8, pp. 1497-1510, 1998.
10. A. Saffiotti, K. Konolige, and E.~H. Ruspini, “A multivalued logic approach to integrating planning and control,” Artificial Intelligence, vol. 76, pp. 481--526, March 1995.
11. J. Yen and N. Pfluger, “A fuzzy logic based extension to Payton and Rosenblatt's command fusion method for mobile robot navigation,” IEEE Transactions on Systems, Man, and Cybernetics, vol. 25, no. 6, pp. 971 -- 978, June 1995.
12. P. Pirjanian, “Multiple objective behavior-based control," Journal of Robotics and Autonomous Systems, vol. 31, no. 1-2, pp. 53-60, Apr 2000.
13. L. E. Parker, Heterogeneous Multi-Robot Cooperation, Massachusetts Institute of Technology Ph.D. Dissertation, January 1994. Available as MIT Artificial Intelligence Laboratory Technical Report 1465, February 1994.
14. B.B.Werger, “Ayllu: Distributed Port-Arbitrated Behavior-Based Control,” in Distributed Autonomous Robotic Systems 4, Lynne E. Parker, George Bekey, and Jacob Barhen (eds.), Springer, 2000:25-34.
15. R. Simmons, S. Singh, D. Hershberger, J. Ramos, T. Smith, “First Results in the Coordination of Heterogeneous Robots for Large-Scale Assembly”, In Proceedings of the International Symposium on Experimental Robotics (ISER), Honolulu Hawaii, December 2000.
16. P. Caloud, W. Choi, J.-C. Latombe, Le C. Pape, and M. Yin, “Indoor automation with many mobile robots,” in Proc. IEEE/RSJ IROS’90, 1990, pp. 67-72.
17. H. Asama, A. Matsumoto, and Y. Ishida, “Design of an autonomous and distributed robot system: ACTRESS,” in Proc. IEEE/RSJ IROS’89, 1989, pp. 283-290.
18. K. Jin, P. Liang, and G. beni, “Stability of synchronized control of discrete swarm structures,” in Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA’94), 1994, pp. 1033-1038.
19. P. Pirjanian and M. Mataric, “Multi-robot target acquisition using multiple objective behavior coordination,” in Proc. IEEE Intl Conf. on Robotics and Automation (ICRA2000), San Francisco, April 2000, pp 101-106.
20. P. Pirjanian, T.L. Huntsberger, A. Trebi-Ollennu, H. Aghazarian, H. Das, S. Joshi, and P.S. Schenker, "CAMPOUT: A control architecture for multi-robot planetary outposts," in Proc. SPIE Symposium on Sensor Fusion and Decentralized Control in Robotic Systems III, Vol. 4196, Boston, MA, Nov. 2000, pp. 221-230.