Tightly Coupled Cooperation Among Independent Agents

Daylond Hooper
Air Force Institute of Technology*
2950 Hobson Way
Wright-Patterson AFB, Ohio 45433

Problem Being Addressed

Coordination is defined as "the harmonious functioning of parts for effective results" (Mish 1996). When applied to multiple robot systems, coordination is classified by one of two categories: collaboration or cooperation. Each component contributes to the harmonious functioning aspect of Coordination. Collaboration occurs when multiple robots engage in separate independent tasks to fulfill a higher goal. Collaboration is limited to situations where there is no requirement for synchronization on the tasks. An example of this is in distributed mapping: each robot's progress on generating its portion of the map is unaffected by the progress of other robots. Cooperation, on the other hand, occurs when multiple robots engage in similar or identical tasks that are mutually dependent (coupled) in pursuit of a higher goal. These tasks require at least some degree of synchronization, and cooperative tasks fall on a spectrum that spans from loosely coupled to tightly coupled tasks. More tightly coupled tasks require a greater degree of synchronization. Cooperative box pushing is an example of a somewhat tightly coupled task, since the synchronization requirements are mostly limited to constraining the deviation from the intendended course. Cooperative lifting is a tightly coupled cooperative task because the robots must keep the item fairly level. This way, the robots avoid an uneven distribution of the weight and minimize the risk of damage to themselves and the item.

With highly independent autonomous systems, collaboration and loosely coupled cooperation are readily achieved. This is because each individual task completion is seen as a fulfillment of a subgoal pertaining to the overall goal. Though the apparent progress of a loosely coupled cooperative tasking may vary temporally, delaying the deactivation of the task until it remains complete for a minimum time period can ensure successful completion. However, it is much more difficult to construct a system that can perform tightly coupled cooperative tasks without increasing the constituents' mutual dependence. Many architectures designed to perform tightly coupled cooperative tasks have behavior activations that occasionally come from external sources, such as a task leader. These architectures (such as CAMPOUT (Huntsberger *et al.* 2003) and the layered multi-robot architecture (Simmons *et al.* 2000)) have a high degree of mutual dependence since each robot must not only know the abilities of others, but also be able to activate those abilities. Thus, to maintain independence yet still enable tightly coupled cooperative task completion, two questions should be answered:

- How do independent robots recognize tasks that require cooperation?
- How do independent robots participating in a tightly coupled cooperative task synchronize their behaviors?

The answers to these questions provides a system that allows for dynamic coalition formation and resizing while retaining individual independence, thus enabling a robust, expandable multi-robot system.

Proposed Plan for Research

Recognition of Cooperative Tasks

Short-term cooperation on a task is often employed by humans, even in adversarial conditions. Though the contributors may have end goals that are mutually exclusive, they each contribute to a particular task if their goals coincide. The short-term goals coincide for mutual benefit, regardless of the long-term goals. This is applicable to robotics, whether or not the conditions are adversarial. If a collection of robots is working towards a set of long-term goals, it often proves beneficial for one or more robots to contribute towards a common subgoal. Much of the ability to recognize cooperative tasks is traditionally predicated on the nature of the robot collective, such as robot size, speed, and the torque that the actuators can apply. To provide recognition of cooperative tasks in a collective-independent manner, more extensive modeling of the tasks associated with a goal is employed in the Hybrid Architecture for Multiple Robots (HAMR). This takes advantage of the utility value-generation aspects of HAMR for task allocation, since a single robot is assigned to the task. The robot measures

*This research was sponsored by an AFRL Lab Task 06SN02COR from the Air Force Office of Scientific Research, Lt Col Scott Wells, program manager. The views expressed in this article are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense or the U.S. Government.

Table 1: Procedure for moving a block from position (0, 0) to (5, 5).

Task	Conditions	Action	End Condition
T_1	At Start Position	Move to (0,0)	At (0,0)
T_2	At (0,0)	Lift Block	Block Lifted
T_3	Block Lifted	Move to (5,5)	At (5,5)
T_4	At (5,5)	Drop Block	Block At (5,5)

progress on the task, and provided the progress is changing, no assistance is needed (this is constrained to positive changes if the context of the task is appropriate). If the progress is static, the robot notifies the rest of the collective. Another robot is assigned the task (in addition to the original) then the two robots carry out the associated tasking. This mechanism is similar to the impatience mechanism in ALLIANCE (Parker 1999), but is more distributed and takes better advantage of the existence of previous calculations (from the utility value generation during goal assignment).

Synchronization Among Independent Robots

Before describing the mechanism for synchronizing cooperative task execution, the relationship between goals and tasks is defined. A goal is a desired final set of conditions, S. A task T_n is a condition set and action pair (S_n, A_n) that finishes at an end condition set, S_{n+1}. Thus, a series of tasks fulfills a goal. More sophisticated goals have subgoals, and the tasks employed to fulfill the subgoal are subtasks. For example, a procedure for moving a block (assuming it is lifted) from position (0, 0) to position (5, 5) is shown in Table 1.

The goal presentation shown in Table 1 measures overall goal progress through task completion, but does not measure progress on individual tasks. However, measuring progress on the tasks themselves enhance the ability to detect failure and to synchronize activities. Redefining the example from Table 1, the progress on the move action is measured as a vector amplitude. This provides a relative displacement from the start point to the end point, where the progress is given as the differential

$$1 - \frac{target\ position}{current\ position}$$

This measurement and others similar to it for different tasks provides the ability to report progress and detect failure. If the percentage value remains unchanged for a set period of time, the task has failed and the robot requires assistance. The assistance is provided by the robot from the remaining group that had the greatest utility value for the task. The second robot arrives and the two robots agree on certain task parameters, such as height to lift the block and transportation velocities. The robots adjust their task completion percentages accordingly, and communicate between themselves to synchronize their behaviors. These are passed as percentages: the first robot is the leader and passes values in set percentage increments between 1 and 20 until the task is complete. This mechanism provides high-level synchronization capabilities without increasing the mutual dependence of the involved agents and still allows for heterogeneity in the collective.

The research plan makes use of these tools for recognizing cooperative tasks and synchronizing task execution by implementing them in a heterogeneous multirobot team. This team consists of three ActivMedia Pioneer robots, two MiniWhegs robots, one Whegs robot, and two small tracked robots. The base software architecture is HAMR, a multirobot control architecture based upon the three-layer paradigm (Kortenkamp, Bonasso, & Murphy 1998). HAMR's cooperative mechanisms make use of individual utility calculations, which emphasize individual autonomy and mutual independence for task allocation and execution while also reducing communication overhead. The Coordinator layer in HAMR provides these mechanisms as an integrated aspect of the architecture.

Progress to Date

Current progress includes partial integration of HAMR into a simulated version of the Pioneer robots, with work pending on the Sequencer, Deliberator, and Coordinator aspects. A practical approach to solution of the presented problems has been developed, though certain associated details (such as methods for generating expiration times) are pending. HAMR, by design, works well for collaboration and loosely coupled cooperation. The task extension to cooperation is in progress.

References

Huntsberger, T.; Pirjanian, P.; Trebi-Ollennu, A.; Nayar, H. D.; Aghazarian, H.; Ganino, A. J.; and Garrett, M. 2003. Campout: A control architecture for tightly coupled coordination of multirobot systems for planetary surface exploration. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 33(5):550–558.

Kortenkamp, D.; Bonasso, R. P.; and Murphy, R., eds. 1998. *Artificial Intelligence and Mobile Robots*. AAAI. chapter 8: On Three-Layer Architectures.

Mish, F. C., ed. 1996. *Merriam-Webster's Collegiate Dictionary*. Merriam-Webster, 10th edition.

Parker, L. 1999. Alliance: An architecture for fault tolerant, cooperative control of heterogeneous mobile robots. *Neurocomputing* 28:75–92.

Simmons, R.; Singh, S.; Hershberger, D.; Ramos, J.; and Smith, T. 2000. First results in the coordination of heterogeneous robots for large-scale assembly. In *Proceedings of the International Symposium on Experimental Robotics(ISER)*.