# Adaptive Cooperative Manipulation with Intermittent Contact

Todd D. Murphey and Matanya Horowitz
Electrical and Computer Engineering
University of Colorado at Boulder
Boulder, Colorado 80309
murphey@colorado.edu

*Abstract*— **Cooperative manipulation with multiple, independent agents can be complicated by changing dynamics as the agents come in and out of contact with the object they are manipulating. This effect, combined with uncertainty in the environment, leads to nontrivial issues in terms of guaranteeing convergence and task completion. Here we illustrate how these effects can be mitigated using a decentralized adaptive control technique based on hybrid control. Results are verified in an experiment using three agents.**

## I. Introduction

Although manipulation using multiple contacts has been studied a great deal over the past two decades, it is typically assumed that the contact itself is well-characterized and that the "contact state" of a system (that is, when it is in or out of contact and whether that contact is sticking or slipping) can be pre-determined by the designer of the manipulation task. In realistic deployment situations, this is rarely the case, and one must be careful in designing a system that is robust to unmodeled contact dynamics and unmodeled disturbances.

The past several decades have seen tremendous effort in the development of autonomous, multi-agent systems. Major advances in the theory of distributed cooperative control have emerged that encourage the deployment of large scale systems that will be scalable, flexible, and robust to unanticipated complexity in their environments. Using a distributed, as opposed to a centralized, control method eliminates problems due to computational complexity, scalability, and having a single point of failure. Nearest neighbor laws for such distributed systems are reasonably well understood [1], and maintaining a formation has been achieved in numerous simulations and in several experiments (see as examples [3], [9], [13], [6], [4]).

In general, a system consisting of mechanical agents that come in and out of contact with the manipulated object consists of both *continuous* dynamics and *symbolic* dynamics. The continuous dynamics represent the rigid body dynamics and the structural dynamics of the agent. The symbolic dynamics represent the contact state of the agent with the object as it evolves over time (i.e., as the agent goes in and out of contact and as it transitions between sticking and slipping contact). The sources of uncertainties described here (i.e., uncertain contact states of the agents) are described purely in terms of the symbolic dynamics–hence, the uncertainty is largely decoupled from the continuous state of the coordinated manipulation system.

Coordinated manipulation using many agents to manipulate a single object has just recently become of interest to the robotics community [9], [5]. The basic ideas in these macro-scale analyses are to understand how to "cage" some passive object using a set of robots. The technique used in [9] relies on the notion of "object closure" to ensure that an object will be successfully manipulated. This approach allows for the design of coordinated motions for the purpose of moving a collection of robots to a desired final location while maintaining object closure. The key insight in [9] is that the notion of object closure is intrinsically robust–it states that an object is in object closure if it cannot escape the robots surrounding it. This means that occasional loss of contact does not violate the object closure requirement.

We are interested in treating a meta-level problem. Assuming we can successfully cage an object, can we reliably command a sequence of actions or modal behaviors that are invariant to the particulars of the contact state and other environmental factors? We encode this as a condition on the proximity graph, so that as the modes changes the graph changes as well. When extern disturbances occur, they only have an effect insofar as they change the proximity conditions. We present a technique first developed in [8], [12] that uses a locally-defined energy function that leads to a globally converging coordinated control algorithm in the face of arbitrarily changing proximity graph structures. This is helpful because the dynamics change discretely when the agents come in and out of contact with the manipulated object and allows one to specify discrete stages in the manipulation process (such as swarm, cage, move). The adaptive control system weighs the changing network and changing modes of operation with internal models of the system in order to provide converging responses with acceptable performance. This is accomplished through an internal estimate of the stability margin along with the use of consensus for purposes of performance improvement.

This work is in contrast to other approaches that focus on nonsmooth analytical approaches to showing stability and convergence. The advantage of the approach we show here is that it does not depend on any particular proximity graph structure, and correspondingly does not depend on any particular potential function for potential-based methods. In particular, no common Lyapunov function is required. Proofs of stability have been produced for such systems (e.g., [7], [2]), but typically these proofs impose constraints on the dynamics of the system and the proximity graph. For example, the results in [7] apply only to a specific potential function on the unit-disk graph, and the results in [2] apply to another particular potential function on a Voronoi graph. The difficulty associated with these prior works is that the stability results leave little room for task specification; tasks must be framed in terms of what can be achieved in a stable manner and may therefore be limited to stable area coverage or "flocking" through a series of obstacles. Moreover, the task specification will likely change over time, thus introducing discrete changes into the equations of motion. The key point is that the control mechanism should dictate task specification to the minimum extent possible. To this end, we have developed a more general method of providing convergence in cooperative tasks, focusing on ease of implementation and genericity of proximity graphs to which it is applicable.

This paper is organized as follows. Section II describes the problem in which we are interested, as well as a high-level description of the approach we use. Section III provides the specifics of the techniques we use. Section IV provides a detailed example including both simulation and experiments. Section V includes our conclusions and future directions.

## II. Problem Description

Let $R$ be the set of agents. Let $\mathcal{G}$ be the set of graphs over the vertices $R$. Let the *sensor graph* $G_S$ be a graph where $R$ is the vertex set, and there is an edge (or "link") between two vertices $r_1$ and $r_2 \in R$ iff agents $r_1$ and $r_2$ can both sense each other. Let the *control graph* (also referred to as the *neighbor graph*) $G_N$ be a graph where $R$ is the vertex set, and there is an edge between two vertices $r_1$ and $r_2 \in R$ iff agents $r_1$ and $r_2$ are interacting for control purposes. To simplify notation, we will understand $S$ to be the edge set of $G_S$, $N$ to be the edge set of $G_N$, and $S_i$ and $N_i$ to the corresponding set for a given agent $i$. The graph $G_N$ will be defined by a time-varying switching function $\sigma$, which we will describe in terms of a graph construction algorithm. Note that $N$ (the set of neighbors used in the control law) is necessarily a subset of $S$ (the set of sensed neighbors).

In our approach to showing convergence, instead of computing a limit on the switching frequency explicitly, we use a notion of a global "energy reserve" (first introduced in [12], [11]) to create a convergence-guaranteeing limiting effect on the switching rate. (The idea behind this name is that if a switch will increase the value of the Lyapunov function, there must be enough energy reserve to compensate for this increase.) We find this approach intuitive and moreover straightforward to implement in our distributed scenarios, in which switching events are detected locally. Although any global quantity can be problematic, we will demonstrate that a local estimate of this quantity based upon a zero sum consensus algorithm is sufficient to establish convergence.

The general problem we wish to address is how one takes a sequence of proximity graph definitions useful for coordinated manipulation and implement them on an underlying dynamic, cooperative system. The method presented here allows one to specify arbitrary proximity graph rules, hence potentially moving the correctness question into the graph design domain.

The primary difficulty is that a control $u(G, \mathbf{x})$ (where $G \in \mathcal{G}$ is a graph and $\mathbf{x}$ is the state) has no information about low-level convergence characteristics that may have to be *modified* to preserve convergence. Hence, we will require a mapping $\begin{array}{rcl} \chi : \mathcal{G} & \rightarrow & \mathcal{G} \\ G & \mapsto & G_E \end{array}$ that maps a *desired* proximity graph $G$ to a stably implementable proximity graph $G_E$. An example of such a motivating scenario is discussed in the next section.

We would like to have a system that has known high-level properties (e.g., successful manipulation of an object) while maintaining low-level characteristics such as stability (of the physical system), stability margin, and performance metrics. The basic approach is to translate the proximity graph $G$ to an alternative $G_E$ by using a mapping $\chi$ that is essentially a dynamically updated guard condition that protects the stability of the system. Hence, $\chi$ may be thought of as a means of "filtering" the effects of $G$ based on a stability condition.

If one has a stable cooperative system for each possible network state $G_N$, then one may use an adaptive control strategy to guarantee stability in a decentralized manner. In particular, the idea is to associate with each agent $i$ a value $E_i$ which is defined as the solution to the following differential equation, where $E_i$ has an arbitrarily chosen nonnegative initial value.

$$\begin{aligned} \dot{E}_i(t) &= -k_e d_i(t) \text{ if no switch in graph occurs} \\ E_i(t) &= \lim_{\tilde{t} \to t^-} E_i(\tilde{t}) - \Delta V \text{ otherwise} \end{aligned}$$

where $k_e > 0$ is the same constant for all agents, $0 < k_e < 1$ (this will be formalized shortly in Eqs. (5) and (6)). The value $d_i$ is a local conservative estimate of the stability margin of the system, and is critical to maintaining stability. The value $E_i$ is initialized to a nonnegative value and then evolves according to Equation 5 as long as the network topology is not changing. Whenever there is a switch, $E_i$ is re-initialized to the value given in Equation 6 by subtracting $\Delta V$. We call $E_i$ the *local energy reserve*, and it should be thought of as a local estimate of stability margin relative to the *hybrid* system. (Moreover, replacing $E_i$ with an estimate of $E_i$ can be shown to provide global stability so long as the estimate is conservative.)

This brings us to the simple change necessary to stabilize the system. The modified control $u(\chi(G_N), x(t))$ is identical to $u(G_N, x(t))$, except for the added condition that any switch in the control graph that would cause $\hat{E}_i < 0$ is prohibited. This result provides guaranteed convergence, and one is guaranteed to eventually be able to implement any graph $G$. It is worth noting that the evolution of $E$ is only used in the calculation of $\chi$–it doesn't affect which controls $u$ are admissible for the system. Additionally, this computation is decentralized: agents only need access to local values $E_i$, $d_i$, and local estimates of changes in the Lyapunov functions as the network topology changes.

The key idea is that we are using the evolution of the energy reserve $E_i$ to systematically *block* changes in proximity graph if they will lead to instability (that is, $\chi$ blocks new graphs until stability can be ensured). However, *typically* $u(\chi(G), x) = u(G, x)$ in systems that do not have aggressive controller gains [12]. Hence, $\chi$, though a conservative approach to preserving stability, often does not come into play.

## III. General Technique

Consider a set of agents $R$ and a time-varying switching signal $\begin{array}{rcl}\sigma:\mathbb{R} & \rightarrow & \mathcal{G}\\ t & \mapsto & G_N\end{array}$ that determines the proximity graph and is constant except for discrete changes at times $t_1....t_n$ on the interval $[t_0, t_f]$. Assume that the state for each agent $i$ is $\mathbf{x} \in M$, the governing equations are $\dot{\mathbf{x}} = f(\mathbf{x})$, and that the switching function changes $f$ over time, $\sigma : (\mathbf{x}, t) \longrightarrow f$. The equations of motion of interest are as follows:

$$\begin{aligned} \ddot{\mathbf{x}}_i &= u_i && (1)\\ u_i &= \left\{ \begin{array}{cc} -\dot{\mathbf{x}} & \tau(\sigma) < T \\ \overline{u}_i(\chi(G), x(t)) & \tau(\sigma) \geq T \end{array} \right. && (2) \end{aligned}$$

where $\overline{u}_i$ stabilize $x$ for each choice of $G$, $\tau(\sigma)$ is the length of time since the last change in the network topology $N_i$, and $T$ is a time-delay before $u$ can decrease the Lyapunov function. The filter $\chi$ will be defined shortly. We assume that for each time interval $(t_j...t_{j+1})$ (we will call this interval $\tau_j$), there exists a global potential function $\mathbf{V}_{\sigma(\tau_j)}$ such that $\mathbf{V}_{\sigma(\tau_j)}$ is positive-definite, $\dot{\mathbf{V}}_{\sigma(\tau_j)}$ is negative semi-definite, and $\ddot{\mathbf{V}}_{\sigma(\tau_j)}$ is bounded. (This is satisfied, for instance, under the conditions on the graph Laplacian discussed in [7].) We define the overall potential function $\mathbf{V}_{\sigma(t)}$ to be equal to $\mathbf{V}_{\sigma(\tau_j)}$ on the interval $(t_j...t_{j+1})$, for all $j$.

Define the quantity $s_i$ such that:

$$s_i(t) = \frac{1}{2}\left(\sum_{j\in N_i}\left(\lim_{t\to\tilde{t}^+} P(x_i, x_j) - \lim_{t\to\tilde{t}^-} P(x_i, x_j)\right)\right), \quad (3)$$

where $P(x_i, x_j)$ is the potential between agent $i$ and $j$. Moreover, each agent can determine an estimate $\hat{s}_i$ such that $\sum_{i\in R}\hat{s}_i \geq \sum_{i\in R} s_i$ (often for our purposes $\hat{s}_i = s_i$). This quantity captures the instantaneous change in potential due to the link switching. The factor of 1/2 is present because each link connects to two agents, and thus will be counted twice. It is thus easy to show that the following holds:

$$\sum_{i\in R} s_i = \lim_{t\to\tilde{t}^+}(\mathbf{V}_{\sigma(t)}) - \lim_{t\to\tilde{t}^-}(\mathbf{V}_{\sigma(t)}) \quad (4)$$

Associate with each agent $i$ a value $E_i$ which is called the *local energy reserve*, and is defined as the solution to a differential equation. $E_i$ has an arbitrarily chosen nonnegative initial value and evolves according to the following:

$$\begin{aligned} \dot{E}_i(t) &= \left\{ \begin{array}{cl} 0 & \text{if } s_i(t) = 0 \text{ and } \tau(\sigma) < T \\ -k_e d_i(t) + w_i & \text{if } s_i(t) = 0 \text{ and } \tau(\sigma) \geq T \end{array} \right. && (5)\\ E_i(t) &= \lim_{\tilde{t}\to t^-} E_i(\tilde{t}) - s_i(t) \text{ otherwise} && (6) \end{aligned}$$

where $k_e$ is a global constant, $0 < k_e < 1$ and $\sum_i w_i = 0$ (which will show up as a zero-sum consensus [10] term later). Notice that $E_i$ is initialized to a nonnegative value and then evolves according to Equation 5 as long as $s_i$ is zero (that is, on intervals with no switches). Whenever $s_i \neq 0$ (there is a switch), $E_i$ is re-initialized to the value given in Equation 6.

Each agent maintains a local estimate $\hat{E}_i$, which is initially greater than zero and evolves as follows:

$$\begin{aligned} \dot{\hat{E}}_i(t) &= \left\{ \begin{array}{cl} 0 & \text{if } \hat{s}_i(t) = 0 \text{ and } \tau(\sigma) < T \\ -k_e d_i(t) + w_i & \text{if } \hat{s}_i(t) = 0 \text{ and } \tau(\sigma) \geq T \end{array} \right. && (7)\\ \hat{E}_i(t) &= \lim_{\tilde{t}\to t^-} \hat{E}(\tilde{t}) - \hat{s}_i(t) \text{ otherwise} && (8) \end{aligned}$$

Let the global values $E$ and $\hat{E}$ be defined such that

$$E = \sum_{i\in R} E_i \quad (9)$$

$$\hat{E} = \sum_{i\in R} \hat{E}_i \quad (10)$$

We will call $E$ the *global energy reserve*.

This brings us to the graph filter definition that provides convergence, defined by

$$\chi(G(t), x(t), t) = \left\{ \begin{array}{ll} G & \text{if } \hat{E}_i > 0 \\ \lim_{\tilde{t}\to t^-} \chi(G(t), x(t), \tilde{t}) & \text{otherwise.} \end{array} \right.$$

The filter $\chi$ is an identity on $\mathcal{G}$, except for the added condition that any switch that would cause $\hat{E}_i < 0$ for any agent $i$ is prohibited. Note that the value of $\hat{E}_i$ cannot decrease in the absence of switching if $d_i \leq 0$ for all $i$ (this can be thought of as a conservative estimate of the stability margin of the system for a graph at time $t$). Also, this computation is decentralized; the agents only need access to the local values $E_i$, $d_i$, and $s_i$.

The immediate consequence of modifying $\sigma$ in this way is that $\hat{E} \geq 0$, since it is the sum of all nonnegative terms. It follows from Equations 9 and 10 and the definitions of $s_i$ and $\hat{s}_i$ that $E \geq \hat{E}$. Thus if $\hat{E} \geq 0$, then $E \geq 0$ as well. This allows us to prove the following statement (from [8]).

*Theorem 3.1:* The states in the system in Eq (1) and (2) all converge to a state of of unchanging potential for any sequence of graphs $G(t)$.

Applying this result to cooperative manipulation tasks allows two advantages. First, if a sequence of proximity graphs correspond to different behavioral modes, we can guarantee that switching between modes does not destabilize the formation. Moreover, if adjustment is necessary to implement the next mode (e.g., if a robot is nonholonomic and must reorient itself to successfully change from `swarm` or `cage` mode to `move` mode), then the agents will not increase their energy reserve until they are ready to start the next mode.

Now we may state the algorithm for ensuring convergence in the face of arbitrary time-varying proximity graph topologies.

**Algorithm for Filtering Proximity Graphs**

```
Given a proximity graph $G(x(t))$:
  1) Choose a set of initial values $\hat{E}_i$ greater than zero;
  2) Update $\hat{E}_i$ using Eq. (7) and (8);
  3) Apply $\chi$ to $G(x(t))$ using $\hat{E}_i$;
  4) Calculate the control law $\overline{u}$ using $\chi(G(x(t)))$.
```

Note that the algorithm is completely decentralized and only adds one state ($\hat{E}$) to each vehicle that needs to be maintained.

## IV. Example: Connected Target Tracking with Underactuated Dynamics Using Filtered Gabriel Graphs Interactions

We now introduce an example that takes advantage of Thm. 3.1. We assume we have each agent $i$ with the normalized nonholonomic vehicle kinematics:

$$\begin{array}{rcl} \dot{x}^i & = & \cos(\theta)u_1^i \\ \dot{y}^i & = & \sin(\theta)u_1^i \\ \dot{\theta}^i & = & u_2^i. \end{array} \tag{11}$$

We have the control from Eq.(1) be defined by a quadratic potential function so that we have

$$\overline{u}_i = \big[\sum_{j\in N_i} k_s(\|\mathbf{x}_i - \mathbf{x}_j\| - l_0)\hat{\mathbf{v}}_{ij}\big] - k_d\dot{\mathbf{x}}_i \tag{12}$$

where $\mathbf{x}_i$ represents the Cartesian coordinates describing the agent's position, $\ddot{\mathbf{x}}_i$ is the agent's acceleration, $\dot{\mathbf{x}}_i$ is the agent's velocity, $N_i$ is the set of links connected to this agent, and $\hat{\mathbf{v}}_{ij}$ is the unit vector from agent $i$ to agent $j$. Control constants are the natural length ($l_0$), the stiffness ($k_s$), and the damping coefficient ($k_d$). We require that the system be symmetric: if an agent $a$ has a link connected to agent $b$, then agent $b$ must have a link connected to agent $a$.

In this case, we get a system that trivially satisfies the requirements of Thm 3.1 using a time delay of $T = argmin(\theta(t) - \angle(\sum_j \nabla P(x_i, x_j))$, where $\angle(\sum_j \nabla P(x_i, x_j))$ is the angle of the vector the control law from the graph creates. In fact, any differentially flat system where $\nabla P$ is the gradient of $P$ with respect to $\mathbf{x}$ in the vector space created by the differentially flat outputs will converge using the approach given here. The agents use the command from the graph structure as a differentially flat output to track. If they are oriented improperly, it will take them some finite amount of time to reorient so that their state is equal to the initial condition of the desired differentially flat output. Hence, by Eq.(7), they will not allow $E_i$ to increase for that amount of time. Other circumstances where a switch in network topology may be prohibited include when mechanical contact changes or the agent enters the sensing range of another agent. Although the decision to prohibit a switch is made by each agent based on its local energy reserve, it may be desirable to allow switches to occur whenever the *global* energy reserve is sufficiently large. That is, we do not want to prevent a switch due to low energy reserves in one part of the system, when there are sufficient energy reserves unused somewhere else. Thus, we need some mechanism for sharing information about the energy reserve levels between agents.

We will take advantage of the *average-consensus* algorithm described by Olfati-Saber and Murray [10]. This algorithm allows a distributed set of agents to reach a consensus on a common global value, while sharing information only with their local neighbors. If an agent $i$ has a set of neighbors $S_i$ that it can sense,

$$\bar{u}_i = \sum_{l\in S_i}(E_l - E_i). \tag{13}$$

With the energy reserved defined, we now need to generate the manipulation strategy. Our implementation consists of three modes, `swarm`, `cage`, and `move`. For a given manipulation task, we assume that the path may be described by a series of waypoints, either in $\mathbb{R}^2$ or in $SE(2)$. The mode `swarm` is simply the quadratic interaction rule described before with a logarithm potential from [7] for purposes of collision avoidance. After the agents have surrounded the object, the natural length $l_0$ is reduced until the object is successfully caged. (We do not deal with the geometry of this problem here.) Then, in the `move` mode the agents move the object to the new waypoint location. If the caging condition is not met after some time, the agents re-enter the `cage` mode.At each change in mode, the agents use the energy reserve to filter the mode changes. If the agents leave contact unexpectedly, $\chi$ filters the resulting mode change. In this way, unanticipated changes do not cause instability, even if we make the swarm of robots reactive to the manipulation task.

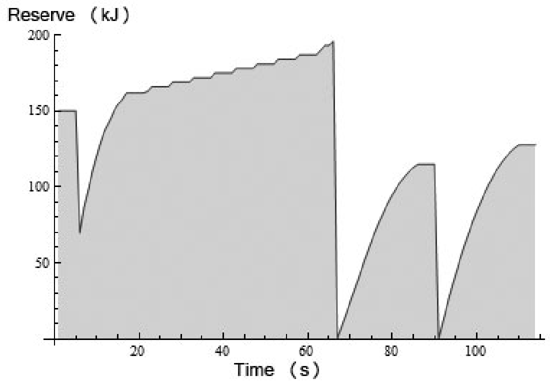

Fig. 2. The energy reserve versus time.

The system evolves somewhat differently than it would with a global value of $E$, as the times when we must prohibit a switch have changed due to the differing *local* values of $E$, but the system meets all the conditions necessary for the proof in Section III because the *global* behavior of $E$ still has the required properties. However, as described in [10], all of the local energy reserves will now converge to a single value.

The consensus strategy found in [10] is just one example of a valid consensus function. In fact, any consensus

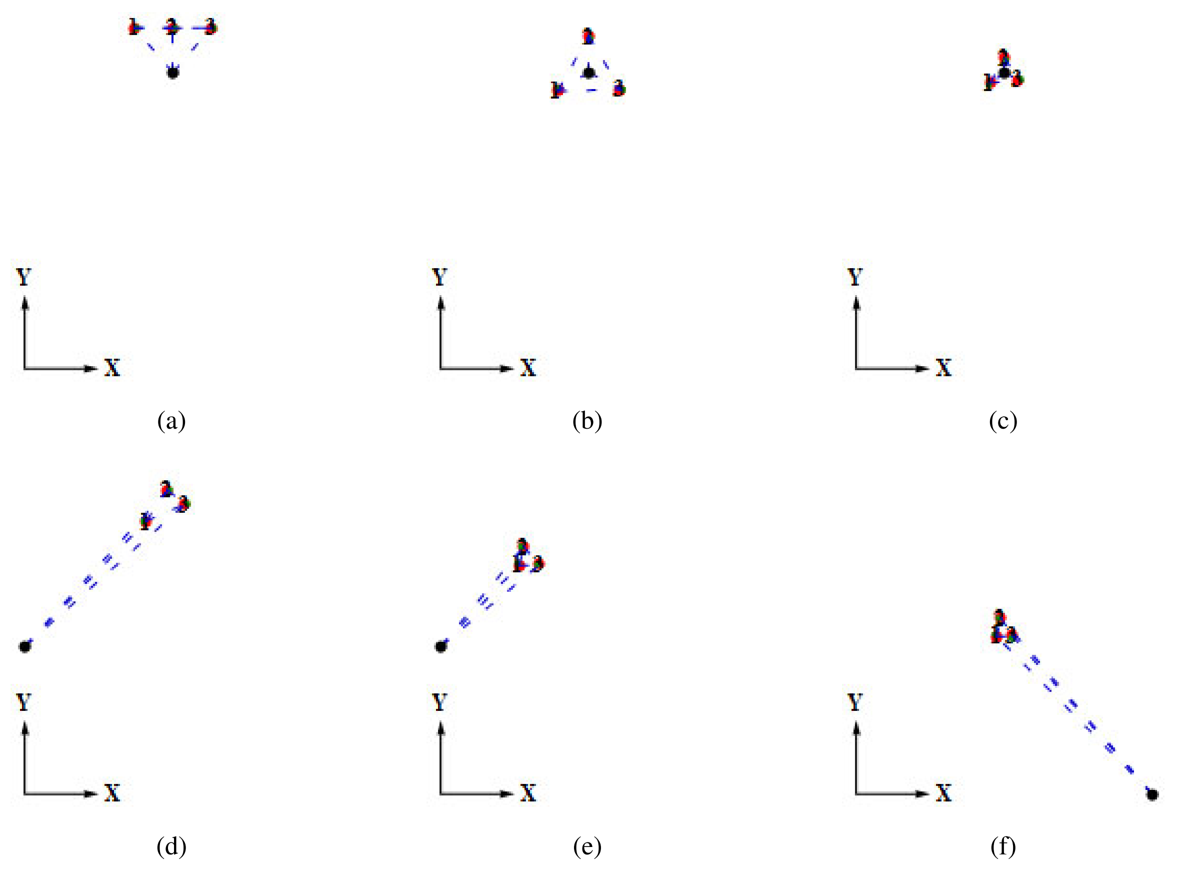

Fig. 1. A simulation of a cooperative manipulation task is shown above.(Agent orientation is indicated by the small black line pointing in the direction the agent is facing.) In (a) the agents are to the side of a box that is to be manipulated. In (b) and (c) they swarm around it and then switch to a caging mode in (d). They then push the object in (e) until a new waypoint is introduced and they move the object someplace else in (f).

algorithm with the zero-sum property is acceptable [8]. The consensus on $E$ is independent of the normal control of the system, although a faster consensus will improve performance in terms of convergence rate.

Simulations of the system in Eq.(11) are seen in Fig.1. In it we see three agents manipulating an object using the modes `swarm`, `cage`, and `move`. In Fig.1(a) we see three agents above the object. They `swarm` around it, and after their average distance to the object is less than $0.5$ m, they proceed to `cage` (in Fig.1(b) and (c)) the object by decreasing the natural length $l_0$ used in their control law from Eq.(12). The transition to `cage` does not require much reorientation, so they are able to do it quickly, as is seen in Fig.2 at time $t \approx 10s$, where the energy eventually drops as they change to `cage` mode. After the object is successfully caged a waypoint is introduced, so they must all reorient. The energy reserve prevents them from continuing with `move` until they are all ready, at which point they start tracking the waypoint in Figs.2(d) and (e). This is seen in the energy reserve in Fig.2 at time $t \approx 65s$, where the energy eventually drops as the change to `move` mode. A new waypoint is introduced, and again the agents have enough energy reserve at time $t \approx 90s$ to start tracking it.

The authors have developed a wireless, decentralized cooperative control testbed. This testbed is adequate for proof-of-concept experiments for all the algorithms that are discussed here. The robot-agents in the test-bed are built using the Roomba robotic vacuum cleaner manufactured by iRobot as a base. This provides for an inexpensive hardware platform that can integrate sensors and distributed computing. The Roomba measures 32cm across and uses a simple differential drive system. (Hence, its dynamics are underactuated, but it can emulate a fully-actuated system reasonably well.) The drive system can be controlled via an externally accessible serial port, called the Roomba Serial Command Interface (SCI). It is also possible to read the Roomba's built-in sensors, including shaft encoders, via the SCI. Thus, a controller board can be mounted on top of the Roomba and connect via the SCI, without needing access to the internals of the base. We are using a Linux-based board (the TS-7260 from Technologic Systems) as the deployed computer. It has wireless, can access sensors, and moreover can use vision and other sensors if necessary. So far, it has been sufficient for testing our algorithms in a completely decentralized wireless testbed.

In the experiment seen in Fig.3, a group of three mobile robots with the dynamics in Eq.(11) are given initial conditions to the side of a box (seen in Fig.3(a)) that they are supposed to manipulate. A potential is used to get the robots to `swarm` around the box (Fig.3(b) and (c)), and then they are supposed to switch to a `cage` mode. This change in mode is filtered by $\chi$, leading to a pause while the

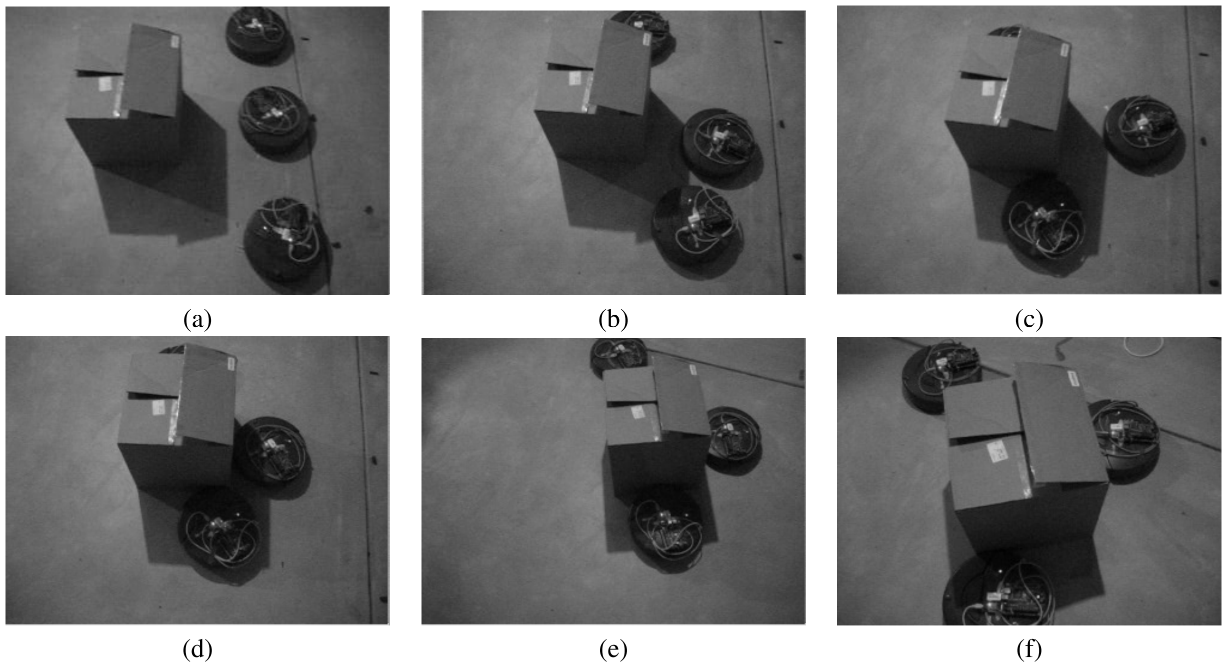

Fig. 3. Video snapshots of cooperative manipulation testbed are shown above. In (a) the robots are to the side of a box that is to be manipulated. In (b) and (c) they swarm around it and then switch to a caging mode in (d). They then push the object in (e) until their odometry readings get off in (f) where they start to drift from the desired formation.

agents adjust their positions and prepare for the next phase. After they have caged the object in Fig.3(d), they switch to a `move` mode, during which they move the object in Fig.3(e). Eventually the agents stop manipulating the object well because of odometry errors accumulating (Fig.3(f)). We are currently implementing an overhead tracking system for dead-reckoning purposes.

## V. Conclusions

In this paper we have applied an adaptive control technique to a relatively simple coordinated manipulation task. We have introduced an approach to cooperative manipulation that focuses on monitoring and filtering the admissible changes in network graph topology according to a stability criterion. This technique can be thought of as filtering modal commands so that the agents do not continue with their task until the other agents are ready. We do not include geometric aspects of guaranteeing caging for irregularly shaped objects. Instead, we focus on robustness aspects of coordinated manipulation. The technique appears to work well both in simulation and in experiment. However, future experiments will need to involve more complex manipulation tasks. In order for this to be feasible, we need to avoid the limited-odometry problems seen in the experiment in Fig.3. We will achieve this using an overhead tracking system. Future analytical work includes proving that objects that have become "uncaged" will become caged again in finite time. This will provide the necessary piece for showing completeness of a manipulation strategy that only uses the three modes `swarm`, `cage`, and `move`.

## References

[1] J. Cortes and F. Bullo. From geometric optimization and nonsmooth analysis to distributed coordination algorithms. In *Proc. of Conf. on Decision and Control*, pages 3274–3280, Maui, Hawaii, Dec. 2003.

[2] Jorge Cortes, Sonia Martinez, Timur Karatas, and Francesco Bullo. Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255, 2004.

[3] Lars Cremean, William Dunbar, David van Gogh, Jason Hickey, Eric Klavins, Jason Meltzer, and Richard M. Murray. The caltech multi-vehicle wireless testbed. In *Conference on Decision and Control (CDC)*, 2002.

[4] A.K. Das, R. Fierro, V. Kumar, J.P. Ostrowski, J. Spletzer, and C.J. Taylor. A vision-based formation control framework. *IEEE Tran. on Robotics and Automation*, 18(5):813–825, Oct. 2002.

[5] M. G. Feemster, J. M. Esposito, and J. Nicholson. Manipulation of large objects by swarms of autonomous marine vehicles: Part i – rotation. In *Proc. IEEE SE Symp. on Systems Theory*, pages 255–259, March 2006.

[6] M. Gillen, A. Lakshmikumar, D. Chelberg, C. Marling, M. Tomko, , and L. Welch. A hybrid, hierarchical schema-based architecture for distributed autonomous agents. In *AAAI symposium on Intelligent, Distributed and Embedded Systems*, 2002.

[7] A. Jadbabaie, Jie Lin, and A.S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988 – 1001, 2003.

[8] T. D. Murphey. Filtering of interaction rules in cooperation. In *Submitted to American Controls Conference (ACC)*, 2008. Submitted.

[9] G.A.S. Pereira, M.F.M. Campos, and V. Kumar. Decentralized algorithms for multirobot manipulation via caging. *International Journal of Robotics Research*, 23(7/8):783–795, 2004.

[10] Reza Olfati Saber and Richard M. Murray. Consensus protocols for networks of dynamic agents. In *American Control Conference*, 2003.

[11] B. Shucker, T. D. Murphey, and J. Bennett. Switching control without nearest neighbor rules. In *Proc. American Controls Conference (ACC)*, pages 5959–5965, 2006.

[12] B. Shucker, T. D. Murphey, and J. Bennett. Switching rules for decentralized control with simple control laws. In *American Controls Conference (ACC)*, pages 1485–1492, 2007.

[13] H.G. Tanner, G.J. Pappas, and V. Kumar. Leader-to-formation stability. *IEEE Tran. on Robotics and Automation*, 20(3):443– 455, June 2004.