43rd IEEE Conference on Decision and Control
December 14-17, 2004
Atlantis, Paradise Island, Bahamas

Receding Horizon Control of Multi-Vehicle Formations: A Distributed Implementation[1]

William B. Dunbar[2] Richard M. Murray[3]

Abstract: *We consider the control of interacting subsystems whose dynamics and constraints are decoupled, but whose state vectors are coupled non-separably in a single cost function of a finite horizon optimal control problem. For a given cost structure, we generate distributed optimal control problems for each subsystem and establish that a distributed receding horizon control implementation is stabilizing. The implementation requires synchronous updates and the exchange of the most recent optimal control trajectory between coupled subsystems prior to each update. Key requirements for stability are that each subsystem not deviate too far from the previous open-loop state trajectory, and that the receding horizon updates happen sufficiently fast. The venue of multi-vehicle formation stabilization is used to demonstrate the distributed implementation and simulations are provided.*

Keywords: receding horizon control; cooperative control; distributed control.

1 Introduction

We are interested in the control of a set of dynamically decoupled subsystems that are required to perform a cooperative task. An example of such a situation is a group of vehicles cooperatively converging to a desired formation, as explored in Dunbar and Murray [5], Ren and Beard [15] and Leonard and Fiorelli [10]. One control approach that accommodates a general cooperative objective is receding horizon control. In receding horizon control, the current control action is determined by solving online, at each sampling instant, a finite horizon optimal control problem. In continuous time formulations, each optimization yields an open-loop control trajectory and the initial portion of the trajectory is applied to the system until the next sampling instant. A survey of receding horizon control is given by Mayne *et al.* [11]. For the problem of interest here, cooperation between subsystems can be incorporated in the optimal control problem by including coupling terms in the cost function, as done in [5]. In this paper, subsystems that are coupled in the cost function are referred to as *neighbors*. A drawback of the receding horizon control approach to our problem is that currently only a centralized solution and implementation can guarantee asymptotic stability theoretically. When the subsystems are operating in a real-time distributed environment, a centralized implementation may not be viable due to the computation and communication requirements of solving the centralized problem at every receding horizon update. In this paper, a *distributed implementation* of receding horizon control is presented in which each subsystem is assigned its own optimal control problem, optimizes only for its own control at each update, and exchanges information only with neighboring subsystems. It is assumed that neighboring subsystems can directly communicate with one another. The motivation for pursuing such a distributed implementation is to enable the autonomy of the individual subsystems while reducing the computation and communication requirements of a centralized implementation.

Previous work on distributed receding horizon control include Jia and Krogh [7], Motee and Sayyar-Rodsaru [14] and Acar [1]. All of these papers address coupled LTI subsystem dynamics with quadratic separable cost functions. State and input constraints are not included, aside from a stability constraint in [7] that permits state information exchanged between the subsystems to be delayed by one update period. In another work, Jia and Krogh [8] solve a min-max problem for each subsystem, where again coupling comes in the dynamics and the neighboring subsystem states are treated as bounded disturbances. Stability is obtained by contracting each subsystems state at every sample period, until the objective set is reached. As such, stability does not depend on information updates between neighbors, although such updates may improve performance. More recently, Keviczky *et al.* [9] have formulated a distributed model predictive scheme where each subsystem optimizes locally for itself and every neighbor at each update. By this formulation, feasibility becomes difficult to ensure, and no proof of stability is provided.

This paper summarizes the results by Dunbar and Murray [6]. We begin in Section 2 by defining the system dynamics and an integrated cost function. Both are specific to a multi-vehicle formation objective, so subsystems are henceforth referred to as *vehicles*. The results of this paper hold for more general subsystem dynamics and performance objectives, as detailed in [4]. In Section 3, the integrated cost is decomposed into distributed integrated costs and a distributed optimal control problem is defined for each vehicle

[1]Partial support for this work was provided by the DARPA SEC program under grant number F33615-98-C-3613 and by AFOSR grant number F49620-01-1-0361.

[2]Department of Computer Engineering, University of California, 1156 High Street, Santa Cruz, CA, 95064, USA. Corresponding Author: dunbar@soe.ucsc.edu

[3]Control and Dynamical Systems, California Institute of Technology, Mail Code 107-81, 1200 E California Blvd, Pasadena, CA 91125, USA.

in the formation. The distributed receding horizon control algorithm is then defined, and the stability results are given in Section 4. Two key requirements for stability are that the receding horizon updates happen sufficiently fast, and that each distributed optimal state trajectory satisfy a *compatibility constraint*. Loosely speaking, the compatibility constraints ensure that the actual state trajectory of each vehicle is not too far from the trajectory that each neighbor assumes for that vehicle, from one receding horizon update to the next. While the compatibility constraints used here incur some conservatism in the closed-loop response, a fact quantified in Section 4, the numerical results in Section 5 show that good closed-loop performance is achieved when these constraints are relaxed. Section 6 discusses conclusions and extensions.

2 System Description and Objective

In this section, we define the system dynamics and pose an integrated cost function relevant for multi-vehicle formation stabilization. The states of the vehicles are coupled in the cost function, while each vehicle is modelled by decoupled dynamics subject to input constraints. The cases where vehicles are subject to coupled and decoupled state constraints are addressed in [4]. We make use of the following notation. The symbol $\|\cdot\|$ denotes any vector norm in $\mathbb{R}^n$, and dimension n follows from the context. For any vector $x \in \mathbb{R}^n$, $\|x\|_P$ denotes the P-weighted 2-norm, defined by $\|x\|_P^2 = x^T P x$, and P is any positive-definite real symmetric matrix. Also, $\lambda_{\max}(P)$ and $\lambda_{\min}(P)$ denote the largest and smallest real eigenvalues of P, respectively. The set $B(x; r)$ denotes a closed ball in $\mathbb{R}^n$ with center x and radius r. When x is a curve, we sometimes abuse the notation $\|x\|$ to mean $\|x(t)\|$ at some instant of time $t \in \mathbb{R}$.

Our objective is to stabilize a group of vehicles toward an equilibrium point in a cooperative way using receding horizon control. For each vehicle $i \in \{1, ..., N_a\}$, the state and control vectors are denoted $z_i(t) = (q_i(t), \dot{q}_i(t)) \in \mathbb{R}^{2n}$ and $u_i(t) \in \mathbb{R}^m$, respectively, at any time $t \geq t_0 \in \mathbb{R}$. The vectors $q_i(t) \in \mathbb{R}^n$ and $\dot{q}_i(t) \in \mathbb{R}^n$ are the position and velocity, respectively, of each vehicle i. The *decoupled* second-order, time-invariant nonlinear system dynamics for each vehicle $i \in \{1, ..., N_a\}$ are given by $\ddot{q}_i(t) = g_i(q_i(t), \dot{q}_i(t), u_i(t))$, which we shall write in the equivalent form

$$\dot{z}_i(t) = f_i(z_i(t), u_i(t)), \quad t \geq t_0, \tag{1}$$

where $f_i(z_i(t), u_i(t)) = (\dot{q}_i(t), g_i(q_i(t), \dot{q}_i(t), u_i(t))) \in \mathbb{R}^{2n}$. It is assumed that there is no model error. While the system dynamics can be different for each vehicle, the dimension of every vehicles state (control) is assumed to be the same, for notational simplicity and without loss of generality. Each vehicle i is also subject to the decoupled input constraints $u_i(t) \in \mathcal{U}$, $t \geq t_0$. The set $\mathcal{U}^N$ is the N-times Cartesian product $\mathcal{U} \times \cdots \times \mathcal{U}$. The concatenated vectors are denoted $q = (q_1, ..., q_{N_a})$, $\dot{q} = (\dot{q}_1, ..., \dot{q}_{N_a})$, $z = (z_1, ..., z_{N_a}) \in \mathbb{R}^{2nN_a}$ and $u = (u_1, ..., u_{N_a}) \in \mathcal{U}^{N_a}$.

In concatenated vector form, the system dynamics are

$$\dot{z}(t) = f(z(t), u(t)), \quad t \geq t_0, \quad \text{given } z(t_0), \tag{2}$$

where $f(z, u) = (f_1(z_1, u_1), ..., f_{N_a}(z_{N_a}, u_{N_a}))$. The desired equilibrium point is denoted $z^c = (z_1^c, ..., z_{N_a}^c)$. Since the dynamics are second-order and time-invariant, the desired equilibrium velocity $\dot{q}_i^c = 0$ for every vehicle i, and the desired constant equilibrium position values are denoted $q^c = (q_1^c, ..., q_{N_a}^c)$. We now make some standard assumptions regarding the system (2) and the set $\mathcal{U}$ (e.g., see (A1)–(A3) in [3]).

Assumption 1 The following holds: (**a**) $f : \mathbb{R}^{2nN_a} \times \mathbb{R}^{mN_a} \to \mathbb{R}^{2nN_a}$ is twice continuously differentiable, $0 = f(z^c, 0)$, and f linearized around $(z, u) = (z^c, 0)$ is stabilizable; (**b**) the system (2) has a unique, absolutely continuous solution for any initial condition $z(t_0)$ and any piecewise right-continuous control $u : [t_0, \infty) \to \mathcal{U}^{N_a}$; (**c**) $\mathcal{U}$ is a compact subset of $\mathbb{R}^m$ containing the origin in its interior.

Let $u_{\max}$ be the positive scalar constant $u_{\max} = \{\max \|v(t)\| \mid v(t) \in \mathcal{U}^{N_a},\ t \geq t_0 \in \mathbb{R}\}$. The integrated cost function relevant for multi-vehicle formation stabilization is defined as

$$L(z, u) = \sum_{(i,j) \in \mathcal{E}_0} \omega \|q_i - q_j + d_{ij}\|^2 + \omega \|q_\Sigma - q_d\|^2 + \nu \|\dot{q}\|^2 + \mu \|u\|^2,$$

given the positive weighting constants $\omega, \nu, \mu \in \mathbb{R}$, and where $\omega \|q_\Sigma - q_d\|^2$ is the *tracking cost*, defined by $q_\Sigma = (q_1 + q_2 + q_3)/3$ and $q_d = (q_1^c + q_2^c + q_3^c)/3$. The set $\mathcal{E}_0$ is the *set of all pair-wise neighbors* that defines the formation in the following way. First, if $(i, j) \in \mathcal{E}_0$, then $(j, i) \notin \mathcal{E}_0$, and $(i, i) \notin \mathcal{E}_0$ for every vehicle $i \in \{1, ..., N_a\}$. Next, for every vehicle i there is at least one pair (i, j) or (j, i) in $\mathcal{E}_0$, i.e., every vehicle has at least one neighbor. Finally, associated with $\mathcal{E}_0$ is the set of constant relative vectors $\mathcal{D} = \{d_{ij} \in \mathbb{R}^n | (i, j) \in \mathcal{E}_0\}$, each of which connects the desired equilibrium positions of a pair of neighboring vehicles, i.e., for any two neighbors i and j, $q_i^c + d_{ij} = q_j^c$. Additionally, the relative vectors in $\mathcal{D}$ are consistent with one another in the sense that, e.g., if (i, j), (j, k) and (i, k) are all in $\mathcal{E}_0$, then $d_{ij} + d_{jk} = d_{ik}$. It is assumed at the outset that $\mathcal{E}_0$ and $\mathcal{D}$ are provided by some supervisory mechanism. Note that $L(z, u) = 0$ if and only if $(z, u) = (z^c, 0)$. Also, while the tracking cost is here defined with vehicles 1, 2 and 3, different and fewer (or more) vehicles can be included in this term without loss of generality (see discussion in Chapter 6 of [4]). The set of pair-wise neighbors of any vehicle $i \in \{1, ..., N_a\}$ is defined as $\mathcal{N}_i = \{j \in \{1, ..., N_a\} \mid (i, j) \text{ or } (j, i) \in \mathcal{E}_0\}$. When we refer to the *neighbors* of any vehicle $i \in \{4, ..., N_a\}$, we mean the set $\mathcal{N}_i$, and the *neighbors* of any vehicle $i \in \{1, 2, 3\}$ refers to the set $\mathcal{N}_i \cup \{1, 2, 3\} \setminus \{i\}$. The integrated cost can be equivalently written as

$$L(z, u) = \|z - z^c\|_Q^2 + \mu \|u\|^2, \tag{3}$$

where $Q = Q^T > 0$ (Proposition 6.1 in [4]). In the next section, $L(z, u)$ is decomposed into distributed integrated cost functions. Then, distributed optimal control problems and corresponding distributed receding horizon control algorithm are stated.

3 Distributed Receding Horizon Control

In this section, we introduce notation, define N_a separate optimal control problems and the distributed receding horizon control algorithm. For any vehicle $i \in \{1, ..., N_a\}$, let $z_{-i} = (z_{j_1}, ..., z_{j_k})$ and $u_{-i} = (u_{j_1}, ..., u_{j_k})$ denote the vectors of the states and controls of the neighbors of i, respectively, where the ordering of the sub vectors is arbitrary but fixed. Also, $\dot{z}_{-i} = f_{-i}(z_{-i}, u_{-i})$ represents the collective decoupled dynamics of the neighbors of any vehicle i. The *distributed integrated cost* in the optimal control problem for any vehicle $i \in \{1, ..., N_a\}$ is defined as

$$L_i(z_i, z_{-i}, u_i) = L_i^z(z_i, z_{-i}) + \gamma\mu\|u_i\|^2 + L_d(i), \quad \text{where}$$
$$L_i^z(z_i, z_{-i}) = \sum_{j \in \mathcal{N}_i} \frac{\gamma\omega}{2}\|q_i - q_j + d_{ij}\|^2 + \gamma\nu\|\dot{q}_i\|^2$$
$$\text{and} \quad L^d(i) = \begin{cases} \gamma\omega\|q_{\Sigma} - q_d\|^2/3, & i \in \{1,2,3\} \\ 0, & \text{otherwise,} \end{cases}$$

and $\gamma \in \mathbb{R}$ is a positive constant. By construction, $\sum_{i=1}^{N_a} L_i(z_i, z_{-i}, u_i) = \gamma L(z, u)$. Note that the terms that couple the positions of vehicles are equally weighted in the decomposition, although such a choice is not necessary for the stability results to hold.

In every distributed optimal control problem, the same constant prediction horizon $T \in (0, \infty)$ and constant update period $\delta \in (0, T]$ are used. In practice, the update period $\delta \in (0, T]$ is typically the sample interval. By our distributed implementation, an additional condition on δ is required, namely that it be chosen sufficiently small, as quantified in the next section. At each receding horizon update, every optimal control problem is solved synchronously, i.e., at the same instant in time. The common receding horizon update times are denoted $t_k = t_0 + \delta k$, where $k \in \mathbb{N} = \{0, 1, 2, ...\}$. At each update, every vehicle optimizes only for its own open-loop control, given its current state and that of its neighbors. Since each cost $L_i(z_i, z_{-i}, u_i)$ depends upon the neighboring states z_{-i}, each vehicle i must presume some trajectories for z_{-i} over each prediction horizon. To that end, prior to each update, each vehicle i *receives* an *assumed* control trajectory from each neighbor. Then, using the model, the current state and the assumed control for that neighbor, the assumed state trajectories are computed. Likewise, vehicle i transmits an assumed control to all neighbors prior to each optimization. By design, the assumed control for any vehicle is *the same* in every distributed optimal control problem in which it occurs, i.e., every neighbor of i will assume the same trajectories for i over each prediction horizon. To distinguish all of the different trajectories, we introduce the following notation. Recall that $z_i(t)$ and $u_i(t)$ are the actual state and control, respectively, for each vehicle $i \in \{1, ..., N_a\}$ at any time $t \geq t_0$. Over any prediction interval $[t_k, t_k + T]$, $k \in \mathbb{N}$, associated with current time t_k, for each vehicle $i \in \{1, ..., N_a\}$ we denote

$u_i^p(\tau; t_k)$:	the predicted control trajectory,
$u_i^*(\tau; t_k)$:	the optimal predicted control trajectory,
$\hat{u}_i(\tau; t_k)$:	the assumed control trajectory,

where $\tau \in [t_k, t_k + T]$. The corresponding state trajectories are likewise denoted $z_i^p(\tau; t_k)$, $z_i^*(\tau; t_k)$ and $\hat{z}_i(\tau; t_k)$, and at time $\tau = t_k$, all of these trajectories are equal to the initial condition $z_i(t_k)$. Let $u^p(\tau; t_k)$, $u^*(\tau; t_k)$ and $\hat{u}(\tau; t_k)$ be the concatenated predicted, optimal and assumed control vectors for all vehicles, respectively, with similar notation for the concatenated state vectors. Consistent with z_{-i}, also let $\hat{u}_{-i}(\tau; t_k)$ and $\hat{z}_{-i}(\tau; t_k)$ be the assumed control and state trajectories of the neighbors of i, corresponding to current time t_k. The collection of distributed optimal control problems is now defined.

Problem 1 For each vehicle $i \in \{1, ..., N_a\}$ and at any update time t_k, $k \in \mathbb{N}$: Given $z_i(t_k)$, $z_{-i}(t_k)$, and $\hat{u}_i(\tau; t_k)$ and $\hat{u}_{-i}(\tau; t_k)$ for all $\tau \in [t_k, t_k + T]$, find

$$J_i^*(z_i(t_k), z_{-i}(t_k)) = \min_{u_i^p(\cdot; t_k)} J_i(z_i(t_k), z_{-i}(t_k), u_i^p(\cdot; t_k)),$$

where $J_i(z_i(t_k), z_{-i}(t_k), u_i^p(\cdot; t_k))$ is equal to

$$\int_{t_k}^{t_k+T} L_i(z_i^p(s; t_k), \hat{z}_{-i}(s; t_k), u_i^p(s; t_k))\, \mathrm{d}s + \gamma\|z_i^p(t_k + T; t_k) - z_i^c\|_{P_i}^2,$$

subject to

$$\begin{aligned} &\dot{z}_i^p(\tau; t_k) = f_i(z_i^p(\tau; t_k), u_i^p(\tau; t_k)) \\ &\dot{\hat{z}}_i(\tau; t_k) = f_i(\hat{z}_i(\tau; t_k), \hat{u}_i(\tau; t_k)) \\ &\dot{\hat{z}}_{-i}(\tau; t_k) = f_{-i}(\hat{z}_{-i}(\tau; t_k), \hat{u}_{-i}(\tau; t_k)) \\ &u_i^p(\tau; t_k) \in \mathcal{U} \\ &\|z_i^p(\tau; t_k) - \hat{z}_i(\tau; t_k)\| \leq \delta^2\kappa \end{aligned} \tag{4}$$

for all $\tau \in [t_k, t_k + T]$, with $z_i^p(t_k; t_k) = \hat{z}_i(t_k; t_k) = z_i(t_k)$ and $\hat{z}_{-i}(t_k; t_k) = z_{-i}(t_k)$, and terminal constraint

$$z_i^p(t_k + T; t_k) \in \Omega_i(\varepsilon_i),$$

given the constants $\kappa, \varepsilon_i \in (0, \infty)$, weighting matrix $P_i = P_i^T > 0$, and terminal set $\Omega_i(\varepsilon_i) = \{z \in \mathbb{R}^{2n} \mid \|z - z_i^c\|_{P_i}^2 \leq \varepsilon_i\}$. ■

As part of the optimal control problem, the optimized state for i is constrained to be at most a distance of $\delta^2\kappa$ from the assumed state in Equation (4). We refer to Equation (4) as the state *compatibility constraint*. The constraint is a means of enforcing a degree of consistency between what a vehicle plans to do and what neighbors believe that vehicle will plan to do, proportional to the square of the update period. The optimal control solution to each distributed optimal control problem (assumed to exist) is $u_i^*(\tau;t_k)$, $\tau \in [t_k, t_k+T]$. The closed-loop system for which stability is to be guaranteed is

$$\dot{z}(\tau) = f(z(\tau), u_{\mathrm{RH}}(\tau)), \quad \tau \geq t_0, \tag{5}$$

with the applied *distributed receding horizon control law*

$$u_{\mathrm{RH}}(\tau) = (u_1^*(\tau;t_k), ..., u_{N_a}^*(\tau;t_k)),$$

for $\tau \in [t_k, t_{k+1})$ and any $k \in \mathbb{N}$. The receding horizon control law is updated when each new initial state update $z(t_k) \leftarrow z(t_{k+1})$ is available. Before stating the control algorithm formally, which in turn defines the assumed control for each vehicle at every update, a decoupled terminal controller associated with each terminal cost and constraint set is defined. The linearization of the ith subsystem (1) at $(z_i, u_i) = (z_i^c, 0)$ is denoted $A_i = \frac{\partial f_i}{\partial z_i}(z_i^c, 0)$, $B_i = \frac{\partial f_i}{\partial u_i}(z_i^c, 0)$. By assuming stabilizability for each vehicle i (Assumption 1 (a)), a feasible local linear feedback $u_i = K_i(z_i - z_i^c)$ which stabilizes each nonlinear subsystem (1) in $\Omega_i(\varepsilon_i)$ can be constructed (see [3], [12]). To that end, we make the following assumption.

Assumption 2 For every vehicle $i \in \{1, ..., N_a\}$, the largest positive constant $\varepsilon_i > 0$ is chosen such that $\Omega_i(\varepsilon_i)$ is a positively invariant region of attraction for the closed-loop nonlinear system $\dot{z}_i = f_i(z_i, K_i(z_i - z_i^c))$, and such that for all $z_i \in \Omega_i(\varepsilon_i)$, the stabilizing feedback is feasible, i.e., $u_i = K_i(z_i - z_i^c) \in \mathcal{U}$. In addition, each terminal weighting matrix $P_i = P_i^T > 0$ is chosen to satisfy the Lyapunov equation

$$(A_i + B_iK_i)^T P_i + P_i(A_i + B_iK_i) = -(Q_i + \mu K_i^T K_i),$$

with $Q_i = \lambda_{\max}(Q) I \in \mathbb{R}^{2n\times 2n}$.

Existence of each $\varepsilon_i \in (0, \infty)$ is guaranteed by Lemma 1 in [3]. By construction, $\widehat{Q} := \mathrm{diag}(Q_1, ..., Q_{N_a})$ satisfies $\widehat{Q} \geq Q$, where Q is the weighting for the integrated cost (3). Defining $K = \mathrm{diag}(K_1, ..., K_{N_a})$, $P = \mathrm{diag}(P_1, ..., P_{N_a})$, $A = \mathrm{diag}(A_1, ..., A_{N_a})$ and $B = \mathrm{diag}(B_1, ..., B_{N_a})$, it is trivial that $P = P^T > 0$ satisfies

$$(A + BK)^T P + P(A + BK) = -(\widehat{Q} + \mu K^T K). \tag{6}$$

For each $i \in \{1, ..., N_a\}$, let $z_i^K(t; z(t_0))$ denote the closed-loop solution to

$$\dot{z}_i^K(t; z(t_0)) = f_i\left(z_i^K(t; z(t_0)), K_i\left(z_i^K(t; z(t_0)) - z_i^c\right)\right),$$

with $t \geq t_0$, given initial condition $z_i(t_0)$. The decoupled linear feedbacks are referred to as *terminal controllers*. In the quasi-infinite horizon approach in [3], the (single) terminal controller is never actually employed, as the receding horizon control law is applied for all time. In the *dual-mode* approach in [12], receding horizon control is employed until the state reaches the terminal constraint set, at which point the terminal controller is employed for all future time. The distributed implementation algorithm defined below is based on the quasi-infinite horizon approach, while a dual-mode version is discussed in Section 4. Let $Z_\Sigma \subset \mathbb{R}^{2nN_a}$ denote the set of initial states $z(t)$ which can be steered to $\Omega_1(\varepsilon_1) \times \cdots \times \Omega_{N_a}(\varepsilon_{N_a})$ by a piecewise right continuous control $u^p(\cdot;t) : [t, t+T] \rightarrow \mathcal{U}^{N_a}$. To achieve convergence, the update period must satisfy $\delta \leq \delta_{\max}$, where the constant $\delta_{\max} \in (0, T]$ is defined in the next section. When results apply for any constant $\delta \in (0, T]$, we set $\delta_{\max} = T$. Following the presentation in [12], we now state the control algorithm.

Algorithm 1 At time t_0 with $z(t_0) \in Z_\Sigma$, the *Distributed Receding Horizon Controller* for any vehicle $i \in \{1, ..., N_a\}$ is as follows:

Data: $z_i(t_0)$, $z_{-i}(t_0)$, $T \in (0, \infty)$, $\delta \in (0, \delta_{\max}]$.

Initialization: At time t_0, solve Problem 1 for vehicle i, setting $\hat{u}_i(\tau; t_0) = 0$ and $\hat{u}_{-i}(\tau; t_0) = 0$ for all $\tau \in [t_0, t_0 + T]$ and removing the constraint (4).

Controller:

1. Over any interval $[t_k, t_{k+1})$, $k \in \mathbb{N}$:
   (a) Apply $u_i^*(\tau; t_k)$, $\tau \in [t_k, t_{k+1})$.
   (b) Compute $\hat{u}_i(\tau; t_{k+1}) = \hat{u}_i(\tau)$ as

$$\hat{u}_i(\tau) = \begin{cases} u_i^*(\tau; t_k), & \tau \in [t_{k+1}, t_k + T) \\ K_i\left(z_i^K(\tau; z_i^k) - z_i^c\right), & \tau \in [t_k + T, t_{k+1} + T] \end{cases}$$

   where $z_i^k := z_i^*(t_k + T; t_k)$.
   (c) Transmit $\hat{u}_i(\cdot; t_k)$ to every neighbor and receive $\hat{u}_j(\cdot; t_k)$ from every neighbor j.
2. At any time t_k, $k \in \{1, 2, ...\}$:
   (a) Measure current state $z_i(t_k)$ and measure or receive the current states $z_{-i}(t_k)$.
   (b) Solve Problem 1 for vehicle i, yielding $u_i^*(\tau; t_k)$, $\tau \in [t_k, t_k + T]$. ■

At initialization of Algorithm 1, Problem 1 is solved for each vehicle without enforcing the compatibility constraint (4) and assuming that every neighbor applies zero control over the prediction interval $[t_0, t_0+T]$. The choice of $\hat{u}(\tau; t_0) = 0$ at initialization is motivated in [6]. When $z(t_0) \in Z_\Sigma$, Problem 1 is feasible at initialization, in that the input and terminal constraints are satisfied and every distributed value function $J_i(\cdot)$ is bounded. At every subsequent update t_k, $k \geq 1$, the compatibility constraints are enforced, and each vehicle assumes all neighbors will continue along their previous open-loop plans, finishing with their decoupled linear control laws. Although Algorithm 1 requires the solution to Problem 1 instantaneously at each update time t_k, a predictive version could be stated to account for non-trivial computation times, as discussed in [4]. Also, the algorithm relies on computing the optimal solution to Problem 1 at every update, although the optimal need not be unique. To relax this requirement, a version akin to that in [12] could be stated, wherein each distributed value function $J_i(\cdot)$ satisfies an improvement property from one update to the next. The assumed control trajectories would then be defined in terms of the previous (suboptimal) control.

4 Analysis

In this section, we state the stability results, assess the distributed implementation and discuss alternative formulations. To save space, all proofs are omitted, but can be found in [6]. The main result is that by applying Algorithm 1, the closed-loop state $z(t)$ converges to a neighborhood of the objective state z^c, for a sufficiently small upper bound on the update period $\delta_{\max}$. At any time t_k, $k \in \mathbb{N}$, the sum of the optimal distributed value functions is denoted

$$J^*_\Sigma(z(t_k)) = \sum_{i=1}^{N_a} J^*_i(z_i(t_k), z_{-i}(t_k)).$$

We begin by stating a feasibility property, following the standard arguments in [3] and [12].

Lemma 1 *Suppose Assumptions 1 and 2 hold and $z(t_0) \in Z_\Sigma$. Then, by application of Algorithm 1 with $\delta_{\max} = T$, Problem 1 has a feasible solution at any update time t_k, $k \in \{1, 2, ...\}$. Moreover, the set Z_Σ is a positively invariant set for the closed-loop system* (5).

In the analysis that follows, we require that the optimal and assumed state trajectories remain bounded.

Assumption 3 There exists a constant $\rho_{\max} \in (0, \infty)$ such that $\|z^*(t; t_k) - z^c\| \leq \rho_{\max}$ and $\|\hat{z}(t; t_k) - z^c\| \leq \rho_{\max}$, for all $t \in [t_k, t_k + T]$ and any $k \in \mathbb{N}$.

The following lemma gives a bounding result on the decrease in $J^*_\Sigma(\cdot)$ from one update to the next. Since the compatibility constraints are enforced for update times t_k with $k \geq 1$, the result holds for $k \in \{1, 2, ...\}$.

Lemma 2 *Suppose Assumptions 1-3 hold and $z(t_0) \in Z_\Sigma$. Then, by application of Algorithm 1 with $\delta_{\max} = T$, and for the positive constant ξ defined by*

$$\xi = \gamma\kappa\omega T \left(4\rho_{\max} + T^2\kappa\right) [|\mathcal{E}_0| + 1], \tag{7}$$

the function $J^*_\Sigma(\cdot)$ satisfies

$$J^*_\Sigma(z(t_k + \delta)) - J^*_\Sigma(z(t_k)) \leq$$
$$- \int_{t_k}^{t_k+\delta} \sum_{i=1}^{N_a} L^z_i \left(z^*_i(s; t_k), \hat{z}_{-i}(s; t_k)\right) \mathrm{d}s + \delta^2 \xi,$$

for all $k \in \{1, 2, ...\}$.

Denote the compact level sets as

$$\Omega_\beta = \{z \in \mathbb{R}^{2nN_a} \mid J^*_\Sigma(z) \leq \beta\},$$

with constant $\beta \in (0, \infty)$. The set Ω_β is in the interior of Z_Σ if $\beta > 0$ is sufficiently small. Now, for any $\beta \in (0, \infty)$ such that $\Omega_\beta \subset Z_\Sigma$, we can choose a constant $r = r(\beta) \in (0, \rho_{\max})$ with the following properties:

$$B(z^c; r) \subseteq \Omega_{\beta/2} \quad \text{and} \quad r^2 \leq \frac{8\beta}{\gamma\lambda_{\min}(Q)}. \tag{8}$$

Our main result demonstrates that, for any $\beta \in (0, \infty)$, the closed-loop state trajectory converges to Ω_β, provided that the update period bound $\delta_{\max}$ in Algorithm 1 is proportional to r^2 as defined below, and r satisfies the properties in Equation (8). We first make the following assumptions, and then state the main theorem of the paper.

Assumption 4 The following holds: (**a**) the update period is sufficiently small that the following first-order Taylor series approximation is valid:

$$\sum_{i=1}^{N_a} L^z_i(z^*_i(s; t_k), \hat{z}_{-i}(s; t_k)) \approx \gamma\|z(t_k) - z^c\|^2_Q$$
$$+ 2\gamma(s - t_k)(z(t_k) - z^c)^T Q f(z(t_k), u^*(t_k; t_k)),$$

for all $s \in [t_k, t_k + \delta]$ and any $k \in \mathbb{N}$; (**b**) there exists a Lipschitz constant $\mathcal{K} \in [1, \infty)$ such that for any $z, z' \in Z_\Sigma,\ u, u' \in \mathcal{U}^{N_a}$,

$$\|f(z, u) - f(z', u')\| \leq \mathcal{K}\Big(\|z - z'\| + \|u - u'\|\Big).$$

Theorem 1 *Suppose Assumptions 1-4 hold, $z(t_0) \in Z_\Sigma$ and for a given constant $\beta \in (0, \infty)$ with $\Omega_\beta \subset Z_\Sigma$, the constant $r = r(\beta) \in (0, \rho_{\max})$ is such that the properties in Equation* (8) *are satisfied. Then, by application of Algorithm 1 with*

$$\delta_{\max} = \frac{\gamma(r/2)^2\lambda_{\min}(Q)}{\xi + \gamma\mathcal{K}\rho_{\max}(\rho_{\max} + u_{\max})\lambda_{\max}(Q)}, \tag{9}$$

and ξ given by Equation (7)*, the closed-loop state trajectory enters $B(z^c; r)$ in finite time and remains in Ω_β for all future time.*

The theorem guarantees that, by application of Algorithm 1 with $\delta_{\max}$ given by Equation (9), the closed-loop state trajectory enters the the closed ball $B(z^c; r)$ in finite time and remains in the level set Ω_β for all future time. Moreover, the size of the set Ω_β can be made arbitrarily small provided the positive constant r satisfies the conditions in Equation (8). The price for a smaller set of convergence, i.e., by choosing r smaller, is a smaller bound on the update period $\delta_{\max}$, which in turn results in a tighter bound in the compatibility constraints (4). Still, the conditions above for convergence are only sufficient, and the simulation results in Section 5 demonstrate that good closed-loop performance and convergence is achieved with an update period larger than required by the theory. Observe that the update period bound $\delta_{\max}$ in Equation (9) is proportional to $1/\xi$, which in turn is proportional to $1/\kappa$. So, the compatibility constraint in Equation (4) cannot be independently relaxed by increasing κ, since this results in a smaller bound on δ

As stated in the introduction, our motivation for pursuing a distributed implementation is to enable the autonomy of the individual subsystems while reducing the computation and communication requirements of a centralized implementation. Since each vehicle is computing its own control locally, the autonomy objective is satisfied. Regarding the cost of computation, the distributed implementation is computationally scalable in the sense that the size of every local optimization problem is independent of the total number of subsystems, as well as the number of neighbors of any vehicle. This is a key advantage over a centralized implementation. Comparing the cost of communication is less straightforward, as the distributed implementation requires the transmission of *trajectories*, as opposed to just current state information, at each update. A qualitative analysis comparing the cost of computation and communication of the distributed and centralized implementations is given in [4].

We can also assess the closed-loop performance of the distributed implementation. More than in centralized implementations, the *closed-loop performance depends largely on the optimal open-loop trajectories computed at initialization*. One reason for this dependence is that vehicles update their controls under the assumption that neighbors will act on what was previously optimal. As such, the effect of the initial response is propagated into subsequent responses in a more direct way than in centralized implementations. This is true to the extent that a vehicles performance objective is affected by its neighbors, which is a function of the relative weighting between coupling and non coupling terms. For example, if the initial response is sluggish and coupling terms in the cost are heavily weighted, each vehicles subsequent control will likely be sluggish. Another reason for the dependence of performance on initialization is the compatibility constraints, a fact that we now quantify. Let $N_{\text{RH}} \in \mathbb{N}$ be some number of receding horizon updates after time t_0 such that $N_{\text{RH}} \cdot \delta \approx T$. At the optimum, the state compatibility constraint for each vehicle i is

$$\|z_i^*(t;t_k) - \hat{z}_i(t;t_k)\| \le \delta^2\kappa, \quad t \in [t_k, t_k+T].$$

Over the subinterval of time $[t_k, t_{k-1}+T]$, we have

$$\|z_i^*(t;t_k) - z_i^*(t;z_i(t_{k-1}))\| \le \delta^2\kappa, \quad t \in [t_k, t_{k-1}+T].$$

For $k=1$ and at time $t = t_{N_{\text{RH}}}$, where $t_{N_{\text{RH}}} = t_0 + N_{\text{RH}} \cdot \delta \approx t_0 + T$, we therefore have that

$$\|z_i^*(t_{N_{\text{RH}}}; z_i(t_1)) - z_i^*(t_{N_{\text{RH}}}; z_i(t_0))\| \le \delta^2\kappa.$$

For $k=2$ and at time $t = t_{N_{\text{RH}}}$, we also have that

$$\|z_i^*(t_{N_{\text{RH}}}; z_i(t_2)) - z_i^*(t_{N_{\text{RH}}}; z_i(t_1))\| \le \delta^2\kappa.$$

Applying this recursively up to $k = N_{\text{RH}}$, each at time $t = t_{N_{\text{RH}}}$, summing up both sides of the inequalities and applying the triangle inequality gives

$$\|z_i(t_{N_{\text{RH}}}) - z_i^*(t_{N_{\text{RH}}}; z_i(t_0))\| \le N_{\text{RH}} \cdot \delta^2\kappa \approx T\delta\kappa,$$

where we use the fact that $z_i^*(t_{N_{\text{RH}}}; z_i(t_{N_{\text{RH}}})) = z_i(t_{N_{\text{RH}}})$. After N_{RH} iterations, the current state deviates from the original optimal state, at the appropriate point in time, by at most $T\delta\kappa$. Thus, when the update period is small enough to satisfy the theoretical conditions for convergence, the compatibility constraints imply the closed-loop trajectory must remain relatively close to the trajectory computed at initialization; therefore, the transient response will only be as good as the initial response. If the compatibility constraints are relaxed by choosing a larger update time, one might expect a more optimal transient response at the price of poorer convergence. In fact, simulation experiments show good convergence even in the absence of the compatibility constraints (Chapter 6, [4]). Keep in mind that the convergence conditions in Theorem 1 are sufficient and therefore conservative. Two alternative formulations detailed in [4] include a dual-mode version of the distributed receding horizon control law, and replacing the state compatibility constraints with *control* compatibility constraints.

5 Simulation Example

A simulation of a four vehicle formation is presented in this section. For simplicity, the dynamics of each vehicle are given by $\ddot{q}_i(t) = u_i(t)$, for each $i \in \{1,2,3,4\}$. The objective is a formation that tracks a reference trajectory. To be consistent with the stabilization objective of Section 2, the velocity of the reference trajectory is constant over two time intervals and the receding horizon control law stabilizes the error dynamics over each such time interval. The reference trajectory $(q_{\text{ref}}(t), \dot{q}_{\text{ref}}(t)) \in \mathbb{R}^4$ is defined as

$$q_{\text{ref}}(t) = \begin{cases} (t,0), & t \in [0,10) \\ (10, 10-t), & t \in [10,\infty) \end{cases}, \qquad (10)$$

where $t_0 = 0$ in the notation of the previous sections. The error system for any vehicle i has state $(q_i - q_{\text{ref}}, \dot{q}_i - \dot{q}_{\text{ref}})$ and dynamics $\ddot{q}_i = u_i$. The jump in the reference velocity at $t = 10$ seconds serves to examine how well the error dynamics are stabilized when the desired formation is *reconfigured*. The control constraint set is defined as $\mathcal{U} = \{(v^1, v^2) \in \mathbb{R}^2 \,:\, -1 \le v^j \le 1,\ j = 1,2\}$. To eliminate any offset between the center of geometry of the formation and the reference trajectory, we set $q_d = (0,0)$. The set of pair-wise neighbors defining the desired formation is $\mathcal{E}_0 = \{(1,2),(1,3),(2,4)\}$. The desired relative vectors are constant for the two legs of the reference trajectory, defined as $d_{12} = d_{24} = (-2,1)$ and $d_{13} = (-2,-1)$ for $t \in [0,10)$, and $d_{12} = d_{24} = (1,2)$ and $d_{13} = (-1,2)$ for $t \in [10,\infty)$. The common rotation in the vectors at time 10 seconds is to match the desired heading of the formation with the heading of the reference trajectory. The initial conditions for each vehicle are given as $q_1(0) = (-1,2)$, $q_2(0) = (-4,0)$, $q_3(0) = (-2,0)$ and $q_4(0) = (-7,-1)$, with $\dot{q}_i(0) = (0,0)$ for each vehicle i. In both centralized and distributed receding horizon implementations, a horizon time of $T = 5$ seconds and update period of $\delta = 0.5$ seconds are used. Also, the following weighting parameter values are consistent in both implementations: $\omega = 2.0$, $\nu = 1.0$ and $\mu = 2.0$. To solve the optimal control problems numerically, we employ the Nonlinear Trajectory Generation (NTG) software developed at Caltech. A detailed description of NTG as a real-time trajectory generation package for constrained mechanical systems is given in [13]. For the centralized receding horizon control law, parameter values in the optimal control problem satisfy sufficient conditions for stability [6]. The formation response is shown in Figure 1.

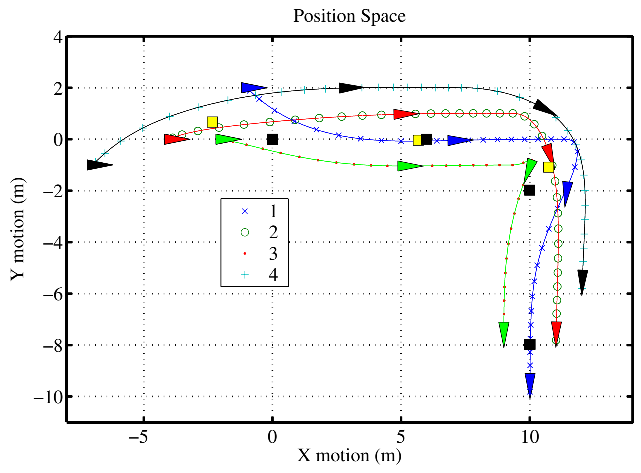

Figure 1: Four vehicle formation using centralized receding horizon control. The light yellow square is $q_\Sigma(t)$ and the black square is $q_{\text{ref}}(t)$, for $t = 0, 6, 12, 18$ seconds.

The four closed-loop position trajectories of the vehicles are shown, with each vehicle depicted by a triangle. The heading of any triangle shows the direction of the corresponding velocity vector. The symbols along each trajectory mark the points at which the receding horizon updates occur. The legend identifies a symbol with a vehicle number for each trajectory. The vehicles are shown at the snapshots of time 0, 6, 12 and 18 seconds. Also shown at these instants of time are the reference trajectory position $q_{\text{ref}}(t)$, identified by the dark square, and the average position of the core vehicles $q_\Sigma(t)$, identified by the light square. The tracking part of the cooperative objective is thus achieved when the two squares are perfectly overlapping. At time 6 seconds, the vehicles are close to the desired formation, and at time 8 seconds the objective has been met with good numerical precision. At time 12 seconds, the snapshot shows the formation reconfiguring 2 seconds after the change in heading of the reference trajectory. At time 18 seconds, the objective has again been met with good numerical precision.

Now, the distributed receding horizon control law is employed. In [6], we show that $\varepsilon_i = 0.33$ for each $i \in \{1, 2, 3, 4\}$ guarantees that the conditions in Assumption 2 hold. We set $\gamma = 2$, and when the compatibility constraints are enforced we set $\kappa = 2$. The compatibility constraints are enforced at every update time except at initialization, as specified by Algorithm 1, and time 10 seconds. Due to the reconfiguration of the desired formation at 10 seconds, the terminal constraints are redefined, in the error dynamics, relative to the new desired location and heading of the formation. Consequently, if the compatibility constraints are enforced at time 10 seconds, each optimization problem no longer becomes feasible. Removing the compatibility constraints at 10 seconds allows each vehicle to find a feasible path the new terminal constraints, and the compatibility constraints are then subsequently enforced. The formation response is shown in Figure 2. The performance is very close to that

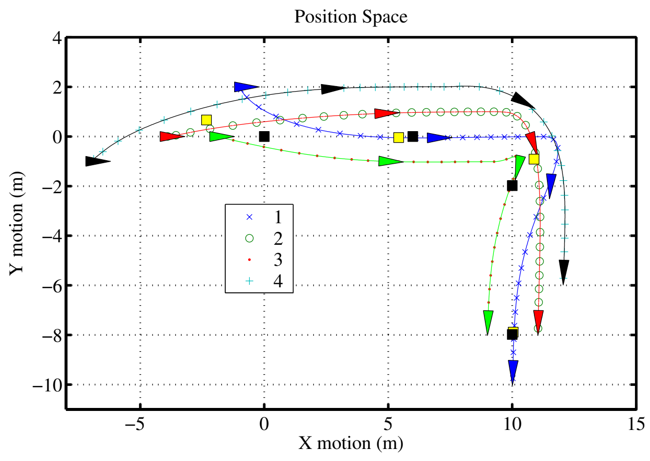

Figure 2: Four vehicle formation using distributed receding horizon control.

of the centralized implementation. At time 6 seconds, the formation is slightly lagging the reference compared to the centralized version. At 18 seconds, the formation objective is close to being met, and for slightly more time the same precision as the centralized implementation is achieved. For the chosen values of δ and κ, the compatibility constraints in fact never become active. This suggests that, based on the assumption that neighbors continue along their previously optimal paths and when coupling occurs in the cost functions, good performance and convergence is achievable, even without enforcing the compatibility constraints. On the other hand, compatibility constraints are imperative to maintain feasibility when coupling constraints are present, as discussed in Section 6. Using the same simulation parameters, we note that *control* compatibility constraints do become active over both transient phases of the formation response, as detailed in [6].

A more naive approach than assuming neighbors continue along their previously optimal paths is to assume neighbors apply *zero* control *at every update*, and the compatibility constraints are never enforced. This was explored in simulations in a previous paper [5], as well as in [6]. In this case, the response is characterized by overshoot, as vehicles believe neighbors will continue with constant velocity over the entire optimization horizon at each update. If the horizon time T is shortened, overall performance improves, as the zero control assumption becomes more valid. The reason is that a straight line approximation of the actual open-loop path is valid locally, under some mild smoothness conditions, and so the assumption is a better one for smaller T. In the formulation in [9] a similar effect is observed. Since vehicles here are relying on the assumption that neighbors keep doing what they *were* doing, and the compatibility constraint ensures that the assumption is not too far off, practical stability is ensured. Moreover, the sensitivity to horizon time, as when neighbors are assumed to continue with constant velocity and as observed in the formulation in [9], is not present here. Regarding the communication requirements of transmitting assumed controls to neighboring vehicles, in the NTG formulation corresponding to the simulations above, 14 B-spline coefficients specified the two-dimensional assumed control trajectory of each vehicle. Polynomial representations of trajectories in the optimization problem, when valid, can aid in keeping the communication requirements closer to that of traditional decentralized schemes.

6 Conclusions and Extensions

In this paper, a distributed implementation of receding horizon control is formulated. An integrated cost function relevant for multi-vehicle formation stabilization that couples the states of a set of dynamically decoupled subsystems is first defined. One aspect of the generality of our approach is that the subsystem dynamics are nonlinear and heterogeneous. Another aspect is that more general coupling functions can also be admitted, as detailed in [4]. The coupling cost is decomposed and distributed optimal control problems are then defined. Each distributed problem is augmented with a *compatibility constraint*, which is a central element in the stability results by ensuring that actual and assumed responses of each vehicle are not too far from one another. Convergence to a neighborhood of the desired equilibrium point is proven in the absence of explicit uncertainty and for sufficiently fast receding horizon updates. We note that a sufficiently fast update period is also required in [12], which addresses robustness to model error. In contrast, we require sufficiently small δ to mitigate an *engineered* uncertainty, namely due to the discrepancy between the assumed and actual controls of every vehicle, which is present even though there is no model error.

Since every optimal control problem is solved globally synchronously, the distributed receding horizon control law is not decentralized, as this requires centralized clock keeping [2]. A locally synchronous, and consequently decentralized, version is explored in [4]. In our distributed approach, no communication is required between vehicles while the distributed optimal control problems are being solved. This is an advantage over parallelization methods [2], where every distributed optimization must communicate with neighboring optimizations while iterating. Thus, our distributed implementation would generally incur a lower communication cost than an approach using receding horizon control with parallelization methods. A tradeoff is that for problems that admit parallelization, convergence to the centralized solution is guaranteed. While the distributed implementation here is stabilizing, it will perform differently in general than the centralized implementation. In fact, the distributed implementation corresponds to the solution of a modified centralized problem, detailed in [4]. As part of our ongoing work, we hope to make quantitative comparisons between the distributed implementation here and parallel implementations of the corresponding centralized problems. Another important extension of our distributed implementation is the ability to handle coupling state constraints, e.g., collision avoidance constraints. The dual-mode receding horizon approach by Michalska and Mayne [12] addresses robustness to model uncertainty in the presence of generic state constraints by making the constraints more conservative. In the same way, coupling state constraints can be incorporated in our distributed implementation, when such constraints can be made more conservative, as detailed in [4].

References

[1] L. Acar. Boundaries of the receding horizon control for interconnected systems. *Journal of Optimization Theory and Applications*, 84(2), 1995.

[2] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.

[3] H. Chen and F. Allgöwer. A quasi-infinite horizon nonlinear model predictive scheme with guaranteed stability. *Automatica*, 14(10):1205–1217, 1998.

[4] W. B. Dunbar. *Distributed Receding Horizon Control of Multi-Agent Systems: Theoretical and Numerical Algorithms (In Preparation)*. PhD thesis, California Institute of Technology, 2004.

[5] W. B. Dunbar and R. M. Murray. Model predictive control of coordinated multi-vehicle formations. In *Proceedings of the IEEE Conference on Decision and Control*, Las Vegas, NV, 2002.

[6] W. B. Dunbar and R. M. Murray. Distributed receding horizon control with application to multi-vehicle formation stabilization. Accepted to *Automatica*, June, 2004, `http://www.ce.ucsc.edu/~dunbar/docs/DRHC.pdf`.

[7] D. Jia and B. H. Krogh. Distributed model predictive control. In *Proceedings of the American Control Conference*, 2001.

[8] D. Jia and B. H. Krogh. Min-max feedback model predictive control for distributed control with communication. In *Proceedings of the American Control Conference*, 2002.

[9] T. Keviczky, F. Borrelli, and G. J. Balas. Model predictive contorl for decoupled systems: A study of decentralized schemes. In *Submitted to the American Control Conference*, Boston, MA, 2004.

[10] N. E. Leonard and E. Fiorelli. Virtual leaders, artificial potentials and coordinated control of groups. In *2001 Conference on Decision and Control*, Florida, 2001.

[11] D.Q. Mayne, J.B. Rawlings, C.V. Rao, and P.O.M. Scokaert. Contrained model predictive control: Stability and optimality. *Automatica*, 36:789–814, 2000.

[12] H. Michalska and D. Q. Mayne. Robust receeding horizon control of contrained nonlinear systems. *IEEE Trans. Auto. Contr.*, 38:1623–1632, 1993.

[13] M. B. Milam, K. Mushambi, and R. M. Murray. A new computational approach to real-time trajectory generation for constrained mechanical systems. In *Proceedings of the Conference on Decision and Control*, 2000.

[14] N. Motee and B. Sayyar-Rodsari. Optimal partitioning in distributed model predictive control. In *Proceedings of the American Control Conference*, 2003.

[15] W. Ren and R. W. Beard. A decentralized scheme for spacecraft formation flying via the virtual structure approach. *Journal of Guidance, Control and Dynamics*, 27(1):73–82, January 2004.