Proceedings of the 2006 American Control Conference
Minneapolis, Minnesota, USA, June 14-16, 2006

Decentralized Robust Receding Horizon Control for Multi-vehicle Guidance

Yoshiaki Kuwata, Arthur Richards, Tom Schouwenaars, and Jonathan P. How

Abstract— **This paper presents a decentralized robust Model Predictive Control algorithm for multi-vehicle trajectory optimization. The algorithm is an extension of a previous robust safe but knowledgeable (RSBK) algorithm that uses the constraint tightening technique to achieve robustness, an invariant set to ensure safety, and a cost-to-go function to generate an intelligent trajectory around obstacles in the environment. Although the RSBK algorithm was shown to solve faster than the previous robust MPC algorithms, the approach was based on a centralized calculation that is impractical for a large group of vehicles. This paper decentralizes the algorithm by ensuring that each vehicle always has *a* feasible solution under the action of disturbances. The key advantage of this algorithm is that it only requires local knowledge of the environment and the other vehicles while guaranteeing robust feasibility of the entire fleet. The new approach also facilitates a significantly more general implementation architecture for the decentralized trajectory optimization, which further decreases the delay due to computation time.**

Keywords: Model Predictive Control (MPC), Constrained, Decentralized, Invariant set, Robust Feasibility

I. INTRODUCTION

Model Predictive Control (MPC) or Receding Horizon Control (RHC) has been successfully applied to trajectory optimization problems for unmanned vehicles [1]–[2] because it can systematically handle constraints such as vehicle dynamics, flight envelope limitations, and no-fly zones. Recent research has focused on *robust MPC*, which is robust to external disturbances or inherent discrepancies between the model and the real process, and numerous techniques have been proposed in the past decade [3]–[4].

Recent work [8] extended a new form of the constraint tightening approach in Ref. [9] to address the computational complexity of the on-line optimization. The main improvement was that the new algorithm did not explicitly require that the system states reach the target over the planning horizon. Instead, the controller only requires that the states can be driven to a robust control invariant set, which is updated as the system evolves. This enables the use of much shorter planning horizons and the online optimizations can be solved much faster. The robust MPC approach was also combined with a *cost-to-go* function that provides a good estimate of the path beyond the planning horizon to the goal [2] to create a *Robust Safe But Knowledgeable* (RSBK) algorithm. While the RSBK algorithm was shown to have a significant computational improvement over the original RMPC algorithm, a centralized planning approach is impractical for a team of multiple vehicles. Using a technique similar to Ref. [10], this paper develops a distributed planning algorithm (DRSBK) that guarantees robust feasibility of the vehicle paths without iterating, which is crucial for real-time implementation.

The primary computational benefit of the DRSBK algorithm over RSBK is that each vehicle only calculates its own trajectory. The interactions with all other vehicles are handled as constraints. We also define a local neighborhood of each UAV that includes all other vehicles that could have a direct conflict with that vehicle. By limiting the number of vehicles considered to only those within a local region of each UAV, the number of constraints in each subproblem is reduced, which further simplifies the DRSBK computation. Using only local communication, DRSBK is shown to maintains the robust feasibility of the entire fleet. The algorithm generalizes the implementation approaches of Refs. [9], [11] to have several of the vehicles computing their trajectories simultaneously. This greatly reduces the delay incurred in the other, more rigid implementation approaches.

The paper is organized as follows. Section III gives an overview of RSBK algorithm, which is extended to the decentralized case with only local communications in Section IV. Finally, Section V shows simulation results.

II. PROBLEM STATEMENT

In this paper, the index or subscript p, q denotes the vehicle index, index k denotes the current time step, and index j denotes the prediction step. There are total of n vehicles whose dynamics are described by LTI model

$$\boldsymbol{x}_p(k+1) = A_p \boldsymbol{x}_p(k) + B_p \boldsymbol{u}_p(k) + \boldsymbol{w}_p(k) \qquad (1)$$

for $p = 1, \ldots, n$, where $\boldsymbol{x}_p(k)$ is the state vector, $\boldsymbol{u}_p(k)$ is the input vector, and $\boldsymbol{w}_p(k)$ is the disturbance vector for the p^{th} vehicle. The disturbances $\boldsymbol{w}_p(k)$ are unknown but lie in known bounded sets $\boldsymbol{w}_p(k) \in \mathcal{W}_p$. The environment has obstacles to be avoided, and the vehicles have flight envelope limitations. The general output sets $\mathcal{Y}_p$ capture these local constraints of each vehicle $p = 1, \ldots, n$

$$\boldsymbol{y}_p(k) = C_p \boldsymbol{x}_p(k) + D_p \boldsymbol{u}_p(k) \in \mathcal{Y}_p. \qquad (2)$$

The coupling between vehicles is captured by a further set of constraints $c = 1, \ldots, m$ applied to the sum of outputs

Y. Kuwata, Dept. of Aeronautics and Astronautics, MIT, Cambridge, MA 02139, USA, kuwata@mit.edu

A. Richards, Lecturer, University of Bristol, Dept. of Aerospace Engineering, Arthur.Richards@bristol.ac.uk

T. Schouwenaars, Dept. of Aeronautics and Astronautics, MIT, Cambridge, MA 02139, USA, toms@mit.edu

J. How, Associate Professor, Dept. of Aeronautics and Astronautics, MIT, Cambridge, MA 02139, USA, jhow@mit.edu

1-4244-0210-7/06/$20.00 ©2006 IEEE

from each vehicle

$$\forall c: \quad \boldsymbol{z}_{cp}(k) = E_{cp}\boldsymbol{x}_p(k), \quad \forall p = 1,\dots,n$$
$$\sum_{p=1}^{n} \boldsymbol{z}_{cp}(k) \in \mathcal{Z}_c. \tag{3}$$

For pair-wise collision avoidance constraints, each constraint c has only two nonzero matrices E_{cp} and E_{cq}, and enforces a minimum separation between that pair of vehicles $\|\boldsymbol{r}_p(k) - \boldsymbol{r}_q(k)\| \geq S$, where $\boldsymbol{r}_p$ is a position of the vehicle p. Note that each set $\mathcal{Z}_c$ is non-convex in this case. Finally, the objective of the trajectory optimization is to navigate vehicles to their assigned targets, and the objective function is assumed to be decoupled.

$$J = \sum_{p=1}^{n}\sum_{k} \tilde{l}_p\Big(\boldsymbol{x}_p(k), \boldsymbol{u}_p(k)\Big) \tag{4}$$

III. Robust Safe but Knowledgeable Algorithm

The problem of interest has the overall goal of **reaching the target** while robustly **maintaining feasibility**. The algorithm relaxes the constraints that the target set must be reached in the planning horizon, allowing the controller to use a short planning horizon. The computational burden then becomes less severe even when the target set is far. A cost-to-go function provides a good estimate of the remainder of the path to reach the target, even in a complicated environment. Additional constraints are added that require that *some* control invariant admissible set is reachable over the short planning horizon.

A. RSBK Algorithm

This section briefly describes a robust constrained RHC algorithm for a centralized case, *i.e.* solving for the plans of all vehicles $p = 1,\dots,n$ in a single optimization. Section IV will show how this computation can be divided into a sequence of smaller problems. The problem in this section corresponds to the *Robust Safe but Knowledgeable* (RSBK) algorithm presented in Ref. [8]. The on-line optimization develops the control inputs for a finite horizon of N steps. The prediction of a value at time $(k+j)$ made at time k is denoted by indices $(k+j|k)$. The MPC optimization problem at time k is defined as:

$$J^* = \min_{\substack{{}_p(\cdot),\mathcal{Q}_p(k)\\ p=1,\dots,n}} \sum_{p=1}^{n}\Big\{\sum_{j=0}^{N} l_p\big(\boldsymbol{u}_p(k+j|k), \boldsymbol{x}_p(k+j|k)\big) + f_p\big(\boldsymbol{x}_p(k+N+1|k)\big)\Big\} \tag{5}$$

subject to $\forall p = 1,\dots,n$, and $\forall j = 0,\dots,N$

$$\boldsymbol{x}_p(k|k) = \boldsymbol{x}_p(k) \tag{6}$$
$$\boldsymbol{x}_p(k+j+1|k) = A_p\boldsymbol{x}_p(k+j|k) + B_p\boldsymbol{u}_p(k+j|k), \tag{7}$$
$$\boldsymbol{y}_p(k+j|k) = C_p\boldsymbol{x}_p(k+j|k) + D_p\boldsymbol{u}_p(k+j|k) \in \mathcal{Y}_p(j), \tag{8}$$
$$\boldsymbol{z}_{cp}(k+j|k) = E_{cp}\boldsymbol{x}_p(k+j|k), \tag{9}$$
$$\sum_{p=1}^{n} \boldsymbol{z}_{cp}(k+j|k) \in \mathcal{Z}_c(j), \ \forall c = 1,\dots,m. \tag{10}$$
$$\boldsymbol{x}_p(k+N+1|k) \in \mathcal{Q}_p(k). \tag{11}$$

The sets $\mathcal{Y}_p(j)$ are constructed by tightening the original set $\mathcal{Y}_p$ using a linear controller $K_p(j)$ that rejects the disturbance.

$$\mathcal{Y}_p(j+1) = \mathcal{Y}_p(j) \sim (C_p + D_pK_p(j))L_p(j)\mathcal{W}_p$$
$$\mathcal{Y}_p(0) = \mathcal{Y}_p \tag{12}$$

where the operator $\sim$ denotes the Pontryagin difference, and $L_p(j)$ is a state transition matrix $(j = 0,\dots,N-1)$

$$L_p(0) = I, \quad L_p(j+1) = (A_p + B_pK_p(j))L_p(j). \tag{13}$$

Similar tightening is performed on the coupling constraint sets in Eq. 10, allowing uncertainty margin for all subsystems within each constraint $(\forall c = 1,\dots,m)$

$$\mathcal{Z}_c(j+1) = \mathcal{Z}_c(j) \sim \Big(E_{c1}L_1(j)\mathcal{W}_1 \oplus \dots \oplus E_{cn}L_n(j)\mathcal{W}_n\Big)$$
$$\mathcal{Z}_c(0) = \mathcal{Z}_c \tag{14}$$

where the operator $\oplus$ denotes the Minkowski summation. The set $\mathcal{Q}_p(k)$ in Eq. 11 is called a *safety* set, defined as

$$\mathcal{Q}_p(k) = \mathcal{R}_p(k) \sim L_p(N)\mathcal{W}$$

where every set $\mathcal{R}_p(k)$ is a robust control invariant admissible set [12], which has the following property

$$\boldsymbol{x}_p \in \mathcal{R}_p(k) \Rightarrow \exists \kappa_p(\boldsymbol{x}_p) \ \text{ s.t.}$$
$$A_p\boldsymbol{x}_p + B_p\kappa_p(\boldsymbol{x}_p) + L_p(N)\boldsymbol{w}_p \in \mathcal{R}_p(k), \ \forall \boldsymbol{w}_p \in \mathcal{W}_p$$
$$C_p\boldsymbol{x}_p + D_p\kappa_p(\boldsymbol{x}_p) \in \mathcal{Y}_p(N)$$
$$\forall c = 1,\dots,m: \ \sum_{p=1}^{n} E_{cp}\boldsymbol{x}_p \in \mathcal{Z}_c(N)$$
$$\forall(\boldsymbol{x}_1,\dots,\boldsymbol{x}_n) \in \{\mathcal{Q}_1(k) \times \dots \times \mathcal{Q}_n(k)\}.$$

Then, one can prove the robust feasibility, i.e. feasibility of the optimization at time k implies feasibility at time $k+1$ under the action of disturbance [13]. The RSBK algorithm parameterizes the invariant set, and by using nilpotent candidate controllers, which gives $L_p(N) = 0$, it can solve for a simple nominal control invariant admissible set. One simple invariant set for fixed-wing aircraft is a loiter circle, or for rotorcraft, any point with zero velocity is invariant.

A function $f_p(\boldsymbol{x})$ in Eq. 5 represents the cost-to-go associated with the trajectory beyond the planning horizon. RSBK uses a sophisticated cost-to-go function [2] for the environment with obstacles. First, a shortest path algorithm is applied to a graph-based representation of the environment to estimate the approximate cost $c(\boldsymbol{r}_{\text{corner}})$ to fly from each obstacle corner $\boldsymbol{r}_{\text{corner}}$ to the target. The pairs of corner location and the cost are stored as a cost map. Then, the optimization chooses the best corner $\boldsymbol{r}_{p,\text{vis}}$ that is visible from plan's terminal position $\boldsymbol{r}_p(k+N+1|k)$ of the vehicle p, so that the cost-to-go function is

$$f_p\big(\boldsymbol{x}_p(k+N+1|k)\big) = \|\boldsymbol{r}_p(k+N+1|k) - \boldsymbol{r}_{p,\text{vis}}\|_2 + c(\boldsymbol{r}_{p,\text{vis}}).$$

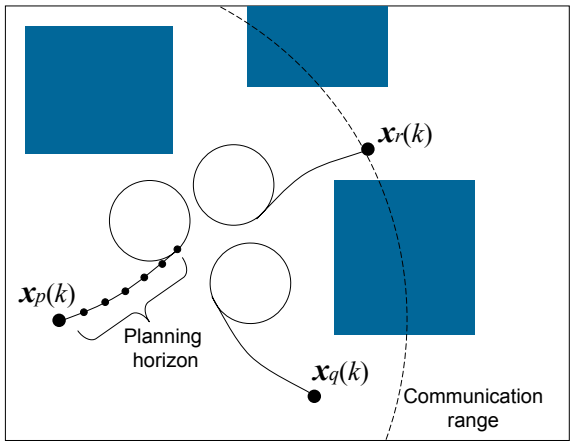


Fig. 1. Neighbor of vehicle p is shown in a dashed line. Each plan terminates in a safety circle.

From the optimal solution at time k, the first control input $\boldsymbol{u}_p^*(k|k)$ for each vehicle is applied to the system Eq. 1. At the next time $k+1$, the states of each vehicle $\boldsymbol{x}_p(k+1)$ is measured, and the optimization is repeated.

IV. DECENTRALIZED RSBK ALGORITHM

This section presents a decentralized version of the RSBK algorithm using an approach that is similar to Ref. [9]. In this new decentralized algorithm, the control inputs for each subsystem are optimized in sequence. However, in this setup, each vehicle exchanges information only with its neighbors, enabling the local optimization to be based on local information [11].

A. Algorithm Description

The basic idea is to include only the vehicles that could have direct conflicts with the vehicle that is planning. Figure 1 shows an example with three aircraft. Any plan of the vehicle r would not have conflict with p's plan because they are far apart. On the other hand, the vehicle q could have a conflict with p if both p and q generate their plans independently and move towards each other. Therefore, p's optimization must include the intention of q, but the vehicle r could be disregarded.

Before presenting the algorithm, several aspects of the notation are defined. First, define vehicle p's neighbor $\mathcal{N}_p(k)$ as an ordered set of vehicles whose plans made at time k could have direct conflicts with p's plan made at time k. For the multi-vehicle collision avoidance problem, the neighborhood of the vehicle p is defined as all the vehicles within distance $2D$ from vehicle p's position, where

$$D = (N+1)v_{\max}\Delta t + \alpha(N) + d \tag{15}$$

with d being the diameter of the loiter circle, and $\alpha(N)$ is the separation distance with an extra margin included for robustness [9]. The dashed line in Figure 1 shows the boundary of p's neighborhood. Note that the neighbor set is a function of time k, because the relative position of the vehicles will change over time.

The set $\mathcal{N}_p(k)$ determines the order that the vehicles calculate their new plans, although Subsection IV-E modifies the assumption on this strict ordering. $\texttt{pre}(q)$ denotes the vehicle ordered prior to the vehicle q, and $\texttt{next}(q)$ denotes the vehicle ordered after q. The first and the last element of this set is expressed as $\texttt{first}(\mathcal{N}_p(k))$ and $\texttt{last}(\mathcal{N}_p(k))$, respectively. With the definition of $\mathcal{N}_p(k)$, we know a priori that for any two vehicles p and q with $q \notin \mathcal{N}_p(k)$ (and hence $p \notin \mathcal{N}_q(k)$), the coupling avoidance constraints are satisfied for all steps with the plans of the two vehicles. Hence, even if the the subproblem for the vehicle p includes only the plans of its neighbors, all of the coupling avoidance constraints will be satisfied.

The result of this analysis is that only a subset of all m coupling constraints need to be considered in each subproblem. Define $\mathcal{C}_p(k) \subset \{1,\dots,m\}$ as the set of coupling constraints to be included in subproblem p at time k. Then $\mathcal{C}_p(k) = \{c \in 1,\dots,m \ : \exists q \in \mathcal{N}_p(k), [E_{cp}\ E_{cq}] \neq \mathbf{0}\}$. This excludes two kinds of constraint irrelevant to p: those coupling p to vehicles outside its neighborhood, and those that do not involve p at all, *i.e.,* with $E_{cp} = \mathbf{0}$.

B. Decentralized RSBK Algorithm

At time k, the p^{th} subsystem generates its own control inputs $\boldsymbol{u}_p(\cdot|k)$ by solving the following optimization subproblem $\mathsf{P}_p(k)$:

$$\min_{p(\cdot|k),\mathcal{Q}_p(k)} \sum_{j=0}^{N} l_p\Big(\boldsymbol{x}_p(k+j|k), \boldsymbol{u}_p(k+j|k)\Big) + f_p(\boldsymbol{x}_p(k+N+1|k)) \tag{16}$$

subject to $\forall j = 0,\dots,N$

$$\text{Eq. (6)–(9), (11),}$$
$$\boldsymbol{z}_{cp}(k+j|k) + \tilde{\boldsymbol{z}}_{cp}(k+j|k) \in \mathcal{Z}_{cp}(j), \quad \forall c \in \mathcal{C}_p(k). \tag{17}$$

The term $\tilde{\boldsymbol{z}}_{cp}(k+j|k)$ is a summation of the outputs from the other vehicles and is constant in this local optimization. The term has two components

$$\tilde{\boldsymbol{z}}_{cp}(k+j|k) = \sum_{\substack{q\in\mathcal{N}_p(k),\\ \texttt{ord}(q)<\texttt{ord}(p)}} \boldsymbol{z}_{cq}^*(k+j|k) + \sum_{\substack{q\in\mathcal{N}_p(k),\\ \texttt{ord}(q)>\texttt{ord}(p)}} \boldsymbol{z}_{cq}^*(k+j|k-1),$$
$$\forall j = 0,\dots,N, \quad \forall c \in \mathcal{C}_p(k) \tag{18}$$

The first term is the summation over the vehicles that have already planned at time k. The second term is for the vehicles that have not planned at time k, so that the prediction made at $(k-1)$ is used. The original coupling constraint sets $\mathcal{Z}_c$ are modified in the following manner, dividing the tightening process from Eq. 14 into intermediate stages for each subsystem

$$\mathcal{Z}_{cp_n}(0) = \mathcal{Z}_c \tag{19a}$$
$$\mathcal{Z}_{c\,\texttt{pre}(q)}(j) = \mathcal{Z}_{cq}(j) \sim E_{cq}L_q(j)\mathcal{W}_q, \quad \forall j = 0,\dots,N, \quad q \in \mathcal{N}_p(k),\ q \neq p_0 \tag{19b}$$
$$\mathcal{Z}_{cp_n}(j+1) = \mathcal{Z}_{cp_0}(j) \sim E_{cp_0}L(j)\mathcal{W}_{p_0}, \quad \forall j = 0,\dots,N-1 \tag{19c}$$

with $p_0 = \texttt{first}(\mathcal{N}_p(k))$ and $p_n = \texttt{last}(\mathcal{N}_p(k))$. Eq. 19b tightens the constraints from the vehicle q to $\texttt{pre}(q)$. This represents that the vehicle $\texttt{pre}(q)$ saves some margin for the vehicle q so that q can use it to reject the disturbances $\mathcal{W}_q$. Eq. 19c tightens the constraints from the prediction step j to $(j+1)$. This represents that the optimization at time k for vehicle p_n saves some margin so that the optimization at time $(k+1)$ for vehicle p_0 can use it to also reject the disturbances.

The terminal set $\mathcal{Q}_p(k)$ has the following property:

$$\begin{aligned} \boldsymbol{x}_p \in \mathcal{Q}_p(k) \Rightarrow \exists \kappa_p(\boldsymbol{x}_p) \ \text{ s.t.} \\ A_p\boldsymbol{x}_p + B_p\kappa_p(\boldsymbol{x}_p) \in \mathcal{Q}_p(k), \\ C_p\boldsymbol{x}_p + D_p\kappa_p(\boldsymbol{x}_p) \in \mathcal{Y}_p(N) \\ \forall c \in \mathcal{C}_p(k): \sum_{q\in\mathcal{N}_p(k)} E_{cq}\boldsymbol{x}_q \in \mathcal{Z}_c(N) \\ \forall (\boldsymbol{x}_{p_0},\ldots,\boldsymbol{x}_{p_n}) \in \{\mathcal{Q}_{p_0}(k)\times\ldots\times\mathcal{Q}_{p_n}(k)\}, \end{aligned} \tag{20}$$

where for the vehicles that have not planned

$$\forall q \in \{\texttt{next}(p),\ldots,p_n\}, \quad \mathcal{Q}_q(k) = \mathcal{Q}_q(k-1).$$

The full DRSBK algorithm is as follows:

1) Find a feasible solution of the DRSBK optimization starting from the current states (See Remark 1).
2) Set $k = 1$.
3) For each vehicle p, measure vehicle states $\boldsymbol{x}_p(k)$ and update the neighbor $\mathcal{N}_p(k)$.
4) For each vehicle p,
   a) Gather, by communication, the latest plans $\boldsymbol{z}^*_{cq}(\cdot|k)$ or $\boldsymbol{z}^*_{cq}(\cdot|k-1)$ from p's neighbors $q \in \mathcal{N}_p(k)$.
   b) Construct a cost map $\boldsymbol{r}_{\text{corner}},\ c(\boldsymbol{r}_{\text{corner}})$.
   c) Solve subproblem $\mathsf{P}_p(k)$.
5) Apply control $\boldsymbol{u}^*_p(k|k)$ to each subsystem p
6) Increment $k := k+1$ and go to 3

Note that this algorithm is also a generalization of the two previously published decentralized MPC algorithms [9], [11], in that it includes both robustness and a short plan that does not necessarily reach the target.

C. Robust Feasibility

Even though each subproblem only uses local information, the robust feasibility of the entire fleet can be proven using an approach that parallels Ref. [10].

Theorem 4.1: If feasible solutions to all subproblems $\mathsf{P}_1(k),\ldots,\mathsf{P}_n(k)$ can be found at time k, then the subproblems at the future times $t \geq k$ are feasible under the action of disturbances.

Proof. The proof is based on a recursion. The details are beyond the scope of this paper, but the following provides an outline of the proof. Without loss of generality, the planning order is assumed to be $1,\ 2,\ldots,n$.

1) Assume all the subproblems $\mathsf{P}_p(k)$ have a feasible solution at time k.
2) Then, it can be shown that a feasible solution exists to the first subproblem $\mathsf{P}_1(k+1)$ at time $k+1$ for all disturbances $\boldsymbol{w}_1(k)$ acting on the vehicle 1 despite the change in the neighbor set $\mathcal{N}_1(k+1)$, by showing that the following candidate solution is feasible.

$$\hat{\boldsymbol{u}}_1(k+j+1|k+1) = \boldsymbol{u}^*_1(k+j+1|k) + K_1(j)L_1(j)\boldsymbol{w}_1(k) \quad j = 0,\ldots,N-1 \tag{21}$$

$$\hat{\boldsymbol{x}}_1(k+j+1|k+1) = \boldsymbol{x}^*_1(k+j+1|k) + L_1(j)\boldsymbol{w}_1(k) \quad j = 0,\ldots,N \tag{22}$$

$$\hat{\boldsymbol{u}}_1(k+N+1|k+1) = \kappa_1(\hat{\boldsymbol{x}}_1(k+N+1|k+1)) \tag{23}$$

$$\hat{\boldsymbol{x}}_1(k+N+2|k+1) = A_1\hat{\boldsymbol{x}}_1(k+N+1|k+1) + B_1\hat{\boldsymbol{u}}_1(k+N+1|k+1) \tag{24}$$

3) Under the assumption in Step 1, it can be shown that given any solution to the problem $\mathsf{P}_p(k+1)$ for $p \in \{1,\ldots,n-1\}$, the next subproblem $\mathsf{P}_{(p+1)}(k+1)$ is feasible, by showing feasibility of a candidate sequence. Similar to Eqs. 21 to 24 in Step 2, the candidate solution is constructed by shifting the previous plan for subsystem $p+1$, assumed known in Step 1, by one time step and adding a perturbation sequence using the predetermined controller $K_{(p+1)}$.

Therefore, at $k+1$, all subproblems $\mathsf{P}_1(k+1),\ldots,\mathsf{P}_n(k+1)$ are feasible. ∎

D. Remarks

Remark 1: *Simple Initialization*: Initializing this algorithm requires the other vehicles' previous solution, as shown in Eqs. 18 and 20. But because the algorithm uses a short horizon and does not require the vehicles to reach the goal over the first plan, a simple loiter pattern could be used for the initialization, assuming the vehicles are far enough apart compared to the diameter of the loiter circle.

Remark 2: *Scalability*: If each subproblem includes the interactions with all the other vehicles, as in Ref. [9], the number of constraints grows rapidly with the size of the fleet, which would increase the problem complexity. The algorithm presented here only requires the information about its neighbors, resulting in a more scalable approach. Furthermore, each vehicle only needs the information from its neighbors, so that the algorithm requires much less communication bandwidth.

E. Simultaneous Computation

Theorem 4.2: Two vehicles p and q can generate trajectories simultaneously without causing infeasibility in the algorithm if $p \notin \mathcal{N}_q$ (and hence $q \notin \mathcal{N}_p$).

Proof. By the definition of neighbor $\mathcal{N}_p$ and $\mathcal{N}_q$, the plans for p and q have no conflict. Given an arbitrary vehicle r $(\neq p,\ q)$, both optimizations by p and q ensure that the same candidate plan similar to Eqs. 21 to 24 for each vehicle r is feasible. Thus, when p and q calculate simultaneously, the vehicle r has a feasible solution at the next optimization. ∎

By applying this theorem to pairs of vehicles in the fleet, it can be shown that more than two vehicles can perform optimization simultaneously. In order to maximize the number of vehicles that compute simultaneously, the grouping problem is casted as a vertex coloring problem, where each vertex represents a vehicle and vertices are connected if they

are neighbors. The goal is to color all the vertices with a minimum number of colors while using different colors for adjacent vertices. Brelaz's heuristic algorithm [14] is used here because it provides good solutions very rapidly. Vehicles of the same color are in one group and can compute their solutions simultaneously.

Note that in order to color the vehicles, the location of all the vehicles in the graph must be known. However, the vehicle can obtain this information by communicating only locally through neighbors.

V. Simulation Results

A. Vehicle Model

The simulation uses homogeneous fixed-wing UAVs. A point-mass dynamics model is used

$$\begin{bmatrix} \boldsymbol{r}_p(k+1) \\ \boldsymbol{v}_p(k+1) \end{bmatrix} = A_p \begin{bmatrix} \boldsymbol{r}_p(k) \\ \boldsymbol{v}_p(k) \end{bmatrix} + B_p \boldsymbol{a}_p(k) + B_p \boldsymbol{w}_p(k),$$

$$\text{with } A_p = \begin{bmatrix} I_2 & \Delta t\, I_2 \\ O_2 & I_2 \end{bmatrix}, B_p = \begin{bmatrix} \frac{(\Delta t)^2}{2} I_2 \\ \Delta t I_2 \end{bmatrix}$$

where $\boldsymbol{r}_p$, $\boldsymbol{v}_p$, and $\boldsymbol{a}_p$ are the position, the velocity, and the acceleration vector respectively. Matrices I_2 and O_2 express an identity matrix and a zero matrix of size 2 respectively. The disturbance $\boldsymbol{w}_p(k)$ enters through the input acceleration and $\boldsymbol{w}_p(k) \in \mathcal{W} = \left\{\boldsymbol{w} \in \mathcal{R}^2 \,|\, \|\boldsymbol{w}\|_\infty \le w_{\max}\right\}$. The disturbance magnitude $w_{\max}$ is 5% of the control authority $a_{\max}$. The planning horizon length N is 5.

The general output constraints include the obstacle avoidance, the maximum/minimum speed, and the maximum input constraints

$$\begin{aligned} \boldsymbol{r}_p(k+j|k) &\notin \mathcal{O} \\ v_{\min} \le \|\boldsymbol{v}_p(k+j|k)\|_2 &\le v_{\max}, \\ \|\boldsymbol{a}_p(k+j|k)\|_2 &\le a_{\max} \end{aligned}$$

where $\mathcal{O} \subset \mathcal{R}^2$ expresses the no-fly zones, and $v_{\min} = 18$, $v_{\max} = 24$, $a_{\max} = 3.84$ are the minimum speed, maximum speed, and maximum acceleration of the UAVs. A two-step nilpotent controller K_p for this system is

$$K_p = \left[-\frac{1}{\Delta t^2} I_2, \quad -\frac{3}{2\Delta t} I_2 \right].$$

Performing constraint tightening (Eq. 12) gives the following constraint set [9]

$$\begin{aligned} \boldsymbol{r}_p(k+j|k) &\notin \mathcal{O} \oplus \alpha(j)\mathcal{W} \\ v_{\min} + \beta(j) \le \|\boldsymbol{v}_p(k+j|k)\|_2 &\le v_{\max} - \beta(j) \\ \|\boldsymbol{a}_p(k+j|k)\|_2 &\le a_{\max} - \gamma(j) \end{aligned}$$

where

$$\begin{aligned} &\alpha(0) = 0, \quad \beta(0) = 0, \quad \gamma(0) = 0, \\ &\alpha(1) = 2.4, \quad \beta(1) = 1.4, \quad \gamma(1) = 0.54, \\ &\alpha(j) = 4.8, \quad \beta(j) = 2.7, \quad \gamma(j) = 0.81, \quad j \ge 2. \end{aligned} \tag{25}$$

The inter-vehicle avoidance constraints are written as

$$\begin{aligned} \left\|\boldsymbol{r}_p(k+j|k) - \boldsymbol{r}_q^*(k+j|k)\right\| &\ge 2\alpha(j), \text{ if } q < p \\ \left\|\boldsymbol{r}_p(k+j|k) - \boldsymbol{r}_q^*(k+j|k-1)\right\| &\ge \alpha(j) + \alpha(j+1), \text{ if } q > p. \end{aligned}$$

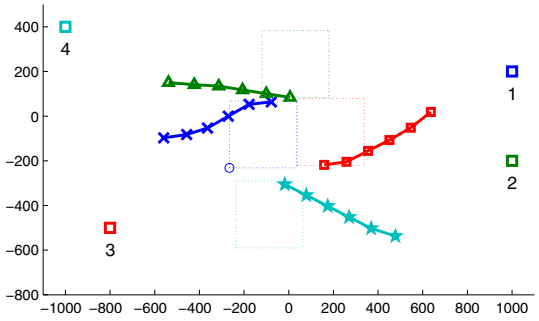


Fig. 2. Snapshot at time 5 of the four vehicle scenario.

These non-convex constraints are implemented using Mixed-integer Linear Programming (MILP).

As shown in Eq. 20, the terminal invariant sets $\mathcal{Q}_p$ must not overlap with each other, so that the sets $\mathcal{Q}_q$ $(\forall q \neq p)$ are treated as no-fly zones in the optimization $\mathsf{P}_p(k)$. Given the terminal position $\boldsymbol{r}_p(k+N|k)$ and the terminal velocity $\boldsymbol{v}_p(k+N|k)$, the safety circle is expressed with n_s sample points on it, where the i^{th} sample point $\boldsymbol{r}_{p,\text{circ},i}$ is

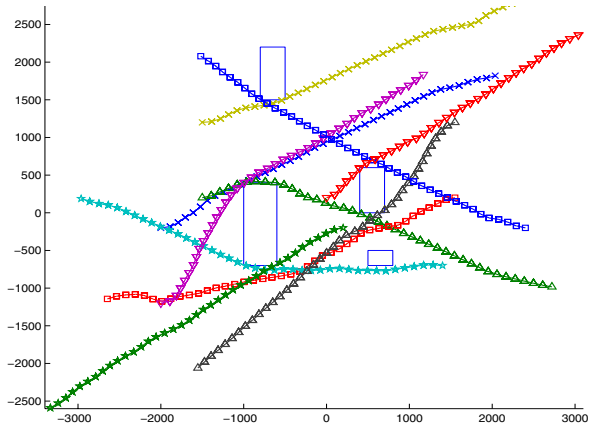
$$\begin{aligned} \boldsymbol{O}_p &= \boldsymbol{r}_p(k+N|k) + R\left(\pm\frac{\pi}{2}\right) \frac{\boldsymbol{v}_p(k+N|k)}{v_{\min} + \beta(N)}\, r_{\min}, \\ \boldsymbol{r}_{p,\text{circ},i} &= \boldsymbol{O}_p + R(\theta_i)\big(\boldsymbol{r}_p(k+N|k) - \boldsymbol{O}_p\big) \end{aligned}$$

where the point $\boldsymbol{O}_p$ is the center of the safety circle, $R(\theta)$ is a rotation matrix, the positive sign corresponds to the left safety circle, and the negative sign corresponds to the right safety circle. The avoidance constraints are enforced on these sampled points [11].

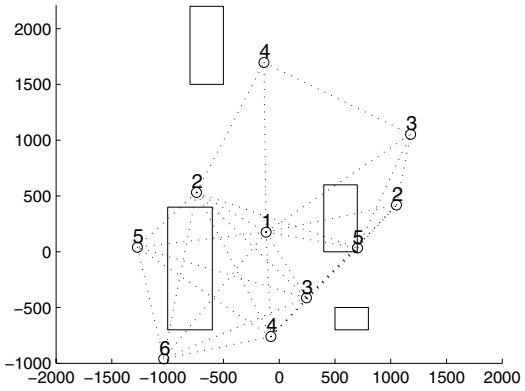
B. Results

The first scenario uses four vehicles. Figure 2 shows the plans made at time $t = 5$. Goals are marked with □ together with the corresponding vehicle indices. The vehicles must avoid collision with each other and the obstacles. The rectangle in dotted lines shows a region where the safety circle is contained and the other vehicles cannot enter after time $k+N$. Note that the plan of the vehicle 4 (marked with star) aims for the corner (marked with ○) of the rectangle of the vehicle 1 because this corner is in the cost map. The vehicle can enter the dotted rectangles as long as its loiter circle does not overlap the loiter circles of other vehicles, which ensures safety of the vehicles.

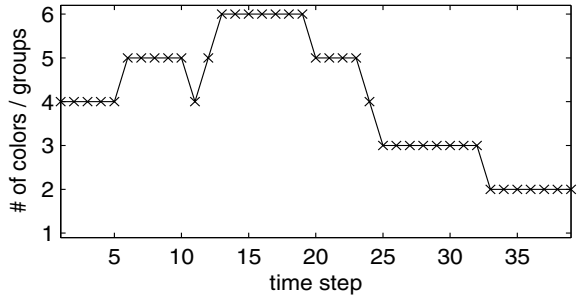
The second scenario is much more complicated and involves ten vehicles and four obstacles. Figure 3(a) shows the trajectories of all 10 vehicles. Figure 3(b) shows a snapshot of the vehicle locations (marked with ○) at time $t = 14$. The neighbors are connected by the dashed line, and each vehicle is labeled with a color/group number. Note that no two vehicles connected to each other have the same label. The vehicles in the same group can simultaneously solve their optimization without any conflict in their trajectories. Figure 3(c) shows the time history of the number of colors required for grouping vehicles. In this scenario, the number of groups is low when the vehicles are far apart, but as might



(a) Trajectories of all the vehicles



(b) Graph representation of neighborhood at time 14



(c) Time history of the vehicle grouping

Fig. 3. Ten vehicle scenario with four obstacles.

be expected, this increases to six in the middle of the mission when the vehicles are in close proximity.

The average computation time for these scenarios are summarized in Table I. The computation time of the cost map calculation grows with the number of vehicles because the loiter circles of other vehicles are treated as obstacles. Since the DRSBK algorithm solves for only a short plan, the computational complexity of MILP does not change when the targets are far from the vehicles. In order to demonstrate the effect of the local communication, the same 10 vehicle scenario is tested with all the vehicles taken into account with full communication among the fleet. As shown by the last two rows of Table I, DRSBK with local communication solves the problem much faster. The grouping algorithm also enables simultaneous computation, which is a function of the maximum number of colors needed to label the graph. In this scenario, the number is reduced from 10 (full communication) to 6 (local communication), which further reduces the computation time of the fleet by 40%.

TABLE I. AVERAGE COMPUTATION TIME (SEC) OF EACH SUBPROBLEM

Scenario	cost map	MILP
4 veh	0.04	0.21
10 veh (local comm.)	0.21	0.25
10 veh (full comm.)	0.21	0.37

VI. CONCLUSIONS

This paper presented a new decentralized robust Model Predictive Control algorithm for multi-vehicle trajectory optimization. The approach extends previous results to ensure robust feasibility without having to plan all of the way to the goal and with only needing communication within a local neighborhood rather than the entire fleet. This new approach greatly reduces the computationally effort and facilitates a significantly more general implementation architecture for the decentralized trajectory optimization.

ACKNOWLEDGMENTS

Research funded by AFOSR grant FA9550-04-1-0458.

REFERENCES

[1] A. Jadbabaie, "Receding horizon control of nonlinear systems: A control lyapunov function approach," Ph.D. dissertation, California Institute of Technology, Pasadena, CA, 2000.

[2] J. Bellingham, A. Richards, and J. How, "Receding Horizon Control of Autonomous Aerial Vehicles," in *Proceedings of the IEEE American Control Conference*, Anchorage, AK, May 2002, pp. 3741–3746.

[3] M. V. Kothare, V. Balakrishnan, and M. Morari, "Robust Constrained Model Predictive Control using Linear Matrix Inequalities," *Automatica*, vol. 32, no. 10, pp. 1361–1379, 1996.

[4] F. A. Cuzzolaa, J. C. Geromel, and M. Morari, "An improved approach for constrained robust model predictive control," *Automatica*, vol. 38, no. 7, p. 1183 1189, 2002.

[5] 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 IEEE Conference on Decision and Control*, Washington DC, 2000, pp. 845–851.

[6] P. O. M. Scokaert and D. Q. Mayne, "Min-Max Feedback Model Predictive Control for Constrained Linear Systems," *IEEE Transactions on Automatic Control*, vol. 8, no. 43, pp. 1136–1142, August 1998.

[7] A. Casavola, M. Giannelli, and E. Mosca, "Min-max predictive control strategies for input-saturated polytopic uncertain systems," *Automatica*, vol. 36, pp. 125–133, 2000.

[8] Y. Kuwata, T. Schouwenaars, A. Richards, and J. How, "Robust Constrained Receding Horizon Control for Trajectory Planning," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, August 2005.

[9] A. Richards and J. How, "Decentralized Model Predictive Control of Cooperating UAVs," in *Proceedings of the IEEE Conference on Decision and Control*, December 2004.

[10] ——, "Decentralized Algorithm for Robust Constrained Model Predictive Control," in *Proceedings of the IEEE American Control Conference*. Boston, MA: IEEE, 2004.

[11] T. Schouwenaars, J. How, and E. Feron, "Decentralized Cooperative Trajectory Planning of Multiple Aircraft with Hard Safety Guarantees," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, August 2004.

[12] E. C. Kerrigan, "Robust Constraint Satisfaction Invariant Sets and Predictive Control," Ph.D. dissertation, University of Cambridge, November 2000.

[13] A. Richards, "Robust Constrained Model Predictive Control," Ph.D. dissertation, MIT, December 2004.

[14] S. Skiena, *Implementing discrete mathematics: combinatorics and graph theory with Mathematica*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1990.