Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16)

Hierarchical Model Predictive Control for Multi-Robot Navigation

Chao Huang[1], Xin Chen[1]*, Yifan Zhang[1], Shengchao Qin[2 3], Yifeng Zeng[2], Xuandong Li[1]
[1]State Key Laboratory for Novel Software Technology, Nanjing University, China
[2]School of Computing, Teesside University, UK [3]Shenzhen University, China

Abstract

Ensuring the stability is the most important requirement for the navigation control of multi-robot systems with no reference trajectory. The popular heuristic-search methods cannot provide theoretical guarantees on stability. In this paper, we propose a Hierarchical Model Predictive Control scheme that employs reachable sets to decouple the navigation problem of linear dynamical multi-robot systems. The proposed control scheme guarantees the stability and feasibility, and is more efficient and viable than other Model Predictive Control schemes, as evidenced by our simulation results.

1 Introduction

The navigation of multi-robot systems is the problem of how to control a group of robots making them move from starting states to their goal states. It has recently attracted increasing attention in the robotics and multi-agent fields. Most of existing work has been devoted to control problems in which a reference trajectory is given as a known condition [Rosales *et al.*, 2011; Antonelli *et al.*, 2013; Tian and Sarkar, 2012]. Given the reference trajectory, the techniques aim to drive a group of robots to cooperatively move toward a destination with a desired formation. Since the predefined reference trajectory can act as explicit landmarks to guide the robots in the navigation, convergence of the methods, referred as *stability*, and the collision avoidance, referred as *feasibility*, are naturally guaranteed.

We consider the case where *no* reference trajectory is available in advance. The situation occurs in many practical applications, particularly when the knowledge of a navigation environment cannot be sufficiently obtained. Undoubtedly, ensuring the stability is the most important requirement of the method for no reference trajectory navigation. Heuristic-search based methods are quite popular in this area [Calliess *et al.*, 2014; Godoy *et al.*, 2015; Janovsky *et al.*, 2014]. However, most of them are unable to provide theoretical results aiding in guaranteeing the stability and feasibility. They may produce trajectories that cause robots to stick at one point or never converge to the target position. That is why a failure ratio becomes a primary index when different methods are compared.

Model Predictive Control (MPC) is one of the most popular optimal control techniques that derives the current control action of a system from its explicit model with constraints and its current state. For centralized MPC (CMPC), the theory of feasibility and stability has been adequately studied [Mayne *et al.*, 2000; Olfati-Saber *et al.*, 2003; Dunbar and Murray, 2002]. However, as CMPC composes together the cost function, the constraints and the dynamics of every robot into a big optimal problem and relies on a central controller to solve the input actions for all robots, it is prone to be computationally prohibitive.

An intuitive method for solving the computational issue is to decompose the big problem into many independent subproblems, distribute them to many robots and solve it in parallel. In basic distributed MPC (BDMPC) [Keviczky *et al.*, 2004b], each robot's local controller only cares for its neighbours and computes its own actions based on the local knowledge of its neighbours. Such a decomposition simply drops many constraints, thus in theory, it gives no guarantee on feasibility and stability. Sequential distributed MPC (SDMPC) [Kuwata and How, 2011] forces local controllers to make computation according to a pre-defined order. Iterative distributed MPC (IDMPC) [Mercangöz and Doyle III, 2007] introduces an iterative computation process to determine the input where each local controller uses MPC to predict its future behavior based on the predicted behaviors of its neighbours received in the last iteration and exchanges it with its neighbours. The process repeats until the solution of the original problem is found. Under certain conditions[1], SDMPC and IDMPC can ensure stability and feasibility, but at the cost of much more computational time.

The parallel decomposition of the navigation problem continues to be a challenging issue since a successful decomposition must carefully decouple all couplings in the objective functions, constraints and dynamics. It should save the computation time while ensuring stability and feasibility. In this paper we propose a *hierarchical* MPC (HMPC) scheme to

*Corresponding author: chenxin@nju.edu.cn

[1]One such condition is that there are only convex constraints involved and the neighbourhood topology for each robot remains unchanged during the whole moving. However, the navigation problem studied in the paper does not meet such a condition.

deal with the navigation problem of linear dynamical multi-robot systems with no reference trajectory. Different from existing works, our HMPC employs a notion of *reachable sets* to help with the decomposition. It ensures stability and feasibility while greatly reducing the computation time. It can also deal with problems with *nonlinear* objective functions, compared to the most recent work based on the Dantzig-Wolfe decomposition [Bourdais *et al.*, 2014] where their objective functions (and constraints) have to be linear.

In the proposed HMPC, each local controller computes its reachable set based on its dynamics and its current state, and then reports it to the central controller. The central controller combines the collected reachable sets with the objective function and the constraints to form an optimization problem whose solution consists of the target states for every robot. Then, each robot's local controller computes its input according to the target assigned to it. Our reachable set is parameterized with the robot's current state and can be derived off-line. Compared with CMPC, the central controller in HMPC needs to solve a much smaller problem as all variables, that are irrelevant to the objective function, global and local constraints and to all the dynamics, are all shielded by the reachable set. The computation of inputs for each robot also gets an acceleration as they are performed in parallel. In addition, the stability and feasibility of the HMPC can be proved in the same way as the conventional CMPC. In this context, this paper makes the following contributions:

- We propose a new control scheme HMPC to address the navigation problem without a reference trajectory based on a new efficient decoupling method via reachable sets.
- We prove the feasibility and stability of the HMPC scheme.
- We illustrate the performance for HMPC in several navigation scenarios and demonstrate its advantages in comparison to multiple MPC schemes.

2 Problem Formulation

We consider a multi-robot system *S* containing *N* robots on a plane. The behavior of each robot is represented as a discrete-time control sub-system and these sub-systems may differ in the dynamic and sampling cycle. Given that the sampling interval of the *i*-th robot is T_i, the interval of synchronization of all the subsystems is the least common multiple of their sampling intervals. We refer to the interval of synchronization as one collaboration cycle. For the *i*-th robot, the collaboration cycle is K_i times its local control cycle.

A state of the *i*-th robot is denoted as $q_i = [x_i, y_i, \dot{x}_i, \dot{y}_i]^T$, where $q_{i,pos} = [x_i, y_i]^T$ and $q_{i,vel} = [\dot{x}_i, \dot{y}_i]^T$ denote the position and velocity along the x-axis and the y-axis, respectively. The dynamic model of the *i*-th robot is described as the linear discrete-time time-invariant state equality below.

$$q_i(k, k_i{+}1){=}A_i\, q_i(k, k_i){+}B_i\, u_i(k, k_i),\ 0{\le}k_i{\le}K_i{-}1, k{\ge}0 \quad (1a)$$

$$\text{with} \qquad q_i(k+1, 0) = q_i(k, K_i) \quad (1b)$$

where k denotes the *k*-th collaboration instant, $k_i{\in}\{0, 1, ..., K^i{-}1\}$ represents the k_i-th local control instant of the *i*-th robot, and u_i denotes the input. Equation (1b) defines the beginning state of one collaboration cycle as the ending state of the last collaboration cycle. For convenience, we may use $q_i(k)$ and $u_i(k)$ to represent $q_i(k, 0)$ and $u_i(k, 0)$ respectively. Let a convex polytope $\mathcal{U}_i$ denote the feasible set of input for the *i*-th robot:

$$u_i(k, k_i) \in \mathcal{U}_i. \quad (2)$$

The input sequence of the *i*-th robot in the *k*-th collaboration cycle is denoted by $\tilde{u}_i(k) \triangleq u_i(k), u_i(k, 1), ..., u_i(k, K_i{-}1)$. The constraint set of the sequence $\tilde{u}_i(k)$ is denoted as $\tilde{\mathcal{U}}_i$:

$$\tilde{u}_i(k) \in \tilde{\mathcal{U}}_i, \qquad \tilde{\mathcal{U}}_i \triangleq \underbrace{\mathcal{U}_i \times ... \times \mathcal{U}_i}_{K_i} \quad (3)$$

For a multi-robot system *S*, let $q(k) \triangleq [q_1(k), ..., q_N(k)]$ denote the global state of *S* at the *k*-th collaboration instant, $\tilde{u}(k) \triangleq \tilde{u}_1(k), ..., \tilde{u}_N(k)$ denote the (list of) input sequence of S in the *k*-th collaboration cycle.

In order to avoid collision during the navigation, we define the collision avoidance specification requiring that the (infinity norm) distance between any two robots must not be less than a given safety distance d_{safe}:

$$h(q) \le 0, \qquad k \ge 0, \quad (4)$$

$$\text{where} \qquad h(q) = [h_{i,j}(q_i, q_j)]_{i,j=1}^{N},$$

$$h_{i,j}(q_i, q_j) = \begin{cases} d_{safe} - \|q_{i,pos} - q_{j,pos}\|_\infty, & i \neq j, \\ 0, & i = j. \end{cases}$$

The infinite norm of a vector $x = [x_1, ..., x_n]^T$ is defined as $\|x\|_\infty \triangleq \max(|x_1|, ..., |x_n|)$.

Following [Gondhalekar *et al.*, 2009], we define two core concepts for MPC, namely *feasibility* and *stability*.

Definition 1. *(*Feasibility*) A MPC controller is* recursively feasible *iff for any feasible initial state, the actual state* $\{q(t), t{\ge}0\}$ *computed by the controller at every collaboration instant satisfies the collision avoidance specification* $h(q(t)){\le}0$.

The definition requires the collision avoidance specification to be met at every control instant, as mentioned in [Gondhalekar *et al.*, 2009]. This definition of feasibility is sufficient to assure that the trajectory satisfies the collision avoidance specification everywhere, if the density of the collaboration points is high enough. Let us assume there exist two robots whose sampling points are disjoint but whose trajectories intersect at one point (*). Provided that the interval of sampling points is small enough, the assumption indicates that we find a case that a discrete sequence of samples is unable to capture all the information from a continuous-time signal (the trajectories), which clearly contradicts with the Nyquist-Shannon sampling theorem. Thus, the assumption (*) cannot be true.

Definition 2. *(*Stability*) A MPC scheme is* stable *iff starting from any feasible initial state, the state sequence* $\{q(t)\}_{t=0}^{\infty}$ *computed by the MPC controller converges to the goal state and the input sequence of each subsystem* $\{\tilde{u}_i(t)\}_{t=0}^{\infty}$*, for* $i{=}1, ..., N$*, converges to zero.*

We shall now formulate the navigation control problem.

Problem 1. *Given a multi-robot system* S*, where*

- *the* i*-th robot has the kinematic model described as in (1a) (1b) and its input constraint described as in (2);*
- S *has an initial state* $q(0) = [q_1(0), ..., q_N(0)]^T$*;*

- S *has a goal state* $q' = [q'_1, ..., q'_N]^T$*;*
- *the collision avoidance specification is given as in (4),*

determine a control strategy to generate the input sequence $\tilde{u}_i(k)$ *for the* i*-th robot,* i=1, ..., N*, at each collaboration instant* k*, so that the state of* S *is transformed from* $q(0)$ *to* q'*, with the satisfaction of feasibility and stability.*

3 Hierarchical Model Predictive Control

To implement a MPC control strategy for a multi-robot system *S*, we give the function $l(q(k|t), \tilde{u}(k|t))$ to denote the cost at the collaboration instant *t*+*k* that is predicted at *t*:

$$\begin{aligned} l(q(k|t), \tilde{u}(k|t)) =& l_q(q(k|t)) + l_u(\tilde{u}(k|t)); \\ l_q(q(k|t)) =& \max_{1\le i\le N} \left\| (C_{abs}(q_i(k|t) - q'_i) \right\|_p + \\ & \max_{1\le i,j\le N} \left\| C_{rel}((q_i(k|t) - q_j(k|t)) - (q'_i - q'_j)) \right\|_p; \\ l_u(\tilde{u}_{k|t}) =& \sum_{i=1}^{N} \left\| C_u \tilde{u}_i(k|t) \right\|_p. \end{aligned}$$

The total cost l consists of the state cost l_q and input cost l_u, where the state cost l_q is the metric of the absolute error and relative error of states, the input cost l_u is the input metric referring to fuel consumption. Note that each function annotated with $...|t$ in their argument denotes a value *predicted* at t. For instance, $q_i(k, k_i|t)$ is similar to $q_i(k{+}t, k_i)$ that we have seen in Equation (1a), except that this is not the actual value at $k{+}t{+}k_i$ for the *i*-th robot, but a value predicted at t. Note that $(\cdot)_i(k|t)$ is short for $(\cdot)_i(k, 0|t)$. Similarly for the whole multi-robot system, the notation $(\cdot)(k|t)$ denotes the *predictive* value at the (*t*+*k*)-th collaboration instant computed at time t. C_{abs} (resp. C_{rel}) is the weighting factor of the absolute (resp. relative) error, and C_u is the weighting factor of the input. For different navigation scenarios, their values can be different. p is a constant and usually assigned 1, 2 or ∞. Note that when $p = 1$ *or* ∞, l is linear. When $p = 2$, l is quadratic.

The HMPC scheme works as follows. It computes the reachable set function of each robot *off-line* in advance, parameterized with state variables. At each collaboration instant, the central controller obtains the reachable set of each robot by instantiating the reachable set function with the current state of the robot. It then calculates the control goal that each robot should reach at the next collaboration instant. Upon receiving its desired control goal, each robot then uses its own local controller to find inputs that drive itself from the current state to the desired goal. Compared with CMPC, the central controller in HMPC solves a much smaller problem by using the reachable set. The computation of inputs for each robot also gets an acceleration as they are performed in parallel. The stability and feasibility of HMPC are ensured by the calculation of goals of all robots.

3.1 Computation of the Reachable Set Function

The reachable set of a robot describes the region that it can reach at the next collaboration instant from the current state. For a linear discrete-time time-invariant system, the reachable set is a set-valued function of the initial state, which we refer to as *the reachable set function*, and can be computed.

We first define what we mean by a reachable set.

Definition 3. *The* m*-step reachable set* $R_i(q_i(t), m)$ *of the* i*-th robot is defined as the set comprising all states that the* i*-th robot can reach in* m *local steps from the state* $q_i(t)$*.*

We need to compute the reachable set function in a collaboration cycle of each robot, i.e. $R_i(q_i(t), K_i), i = 1, \cdots, N$.

For the dynamic model of the *i*-th robot specified by Equation (1a), (1b), (2). Recall that

$$\begin{aligned} q_i(k+1) &= A_i\, q_i(k, K_i{-}1) + B_i\, u_i(k, K_i{-}1) \\ &= A_i(A_i\, q_i(k, K_i{-}2) + B_i\, u_i(k, K_i{-}2)) \\ &\quad + B_i\, u_i(k, K_i{-}1) \\ &= \cdots \\ &= (A_i)^{K_i} q_i(k) + \sum_{p=0}^{K_i-1} (A_i)^{K_i-p-1} B_i\, u_i(k, p), \\ & u_i(k, p) \in \mathcal{U}_i, p = 0, ..., K_i{-}1 \end{aligned}$$

Notice that $(A_i)^{K_i-p-1}B_i u_i(k, p), u_i(k, p) \in \mathcal{U}_i$ is a convex polytope. So $\sum_{p=0}^{K_i-1}(A_i)^{K_i-p-1}B_i u_i(k, p), u_i(k, p) \in \mathcal{U}_i$ is actually the Minkowski sum of $K_i{-}1$ convex polytopes. It is also a polytope [Gritzmann and Sturmfels, 1993], and can be computed by the multi-parametric toolbox [Herceg *et al.*, 2013], which is the K_i-step reachable set from the origin, i.e. $R_i(0, K_i)$. Then we have the reachable set function

$$R_i(q_i(k), K_i) = \{q : (q - (A_i)^{K_i} q_i(k)) \in R_i(0, K_i)\}.$$

3.2 The HMPC Scheme

The HMPC consists of two successive steps.

Compute the target states

At a collaboration instant *t*, the central controller collects the current state $q(t)$ and solves the following optimization problem so as to find the states leading to a minimal total *state* cost (including a terminal cost l_H) while satisfying the corresponding constraint system:

$$\left.\begin{aligned} J_c^*(q(t)) \;\triangleq\; & \min_{Q(H|t)} \sum_{k=0}^{H-1} l_q(q(k|t)) + l_H(q(H|t)) \\ \texttt{s.t.} \quad & q_i(k{+}1|t) \in R_i(q_i(k|t), K_i), \\ & k \ge 0, \quad i = 1, ..., N, \\ & h(q(k|t)) \le 0, \quad k = 1, ..., H{-}1, \\ & q(H|t) \in \tilde{\mathcal{Q}}_f, q(0|t) = q(t) \end{aligned}\right\} \quad (5)$$

where $Q(H|t) \triangleq q(1|t), ..., q(H|t)$ is a feasible solution of the optimization problem (5). Let $Q^*(H|t) \triangleq q^*(1|t), ..., q^*(H|t)$ denote the optimal solution. The first sample $q^*(1|t)$ will be used as the desired states $q(t{+}1)$ at the instant $t{+}1$.

$$q(t{+}1) = q^*(1|t)$$

For the vector $q(t{+}1) = [q_1(t{+}1), ..., q_N(t{+}1)]^T$, its *i*-th element $q_i(t{+}1)$ specifies the objective state of the *i*-th robot at the collaboration instant $t{+}1$. Thus, we refer to $q(t{+}1)$ as the *control goal* at the instant $t{+}1$.

Compute the inputs

At the collaboration instant t, the local controller of the *i*-th robot receives the *control goal* $q_i(t{+}1)$ and solves the following optimization problem, which minimizes the total input

cost, ensuring the robot can drive to the state $q_i(t+1)$ from $q_i(t)$ in the *t*-th collaboration cycle.

$$\left.\begin{array}{l} J_i^*(q_i(t),q_i(t+1)) \triangleq \min\limits_{\tilde{u}_i(t)} l_u(\tilde{u}_i(t)) \\ \texttt{s.t.} \quad q_i(t,k_i+1) = A_i q_i(t,k_i) + B_i u_i(t,k_i), \\ \qquad 0 \le k_i \le K_i - 1, \\ \qquad \text{with} \quad q_i(t+1) = q_i(t,K_i), \\ \qquad u_i(t,k_i) \in \mathcal{U}_i, \quad 0 \le k_i \le K_i - 1 \end{array}\right\} \quad (6)$$

The optimal solution $\tilde{u}_i^*(t)$ is the actual input sequence $\tilde{u}_i(t)$ for the *i*-th robot in the next collaboration cycle: $\tilde{u}_i(t)=\tilde{u}_i^*(t)$.

The overall HMPC scheme for the navigation control of a multi-robot system *S* is summarized as follows:

1. Let *t* denote the current collaboration instant: $t := 0$.
2. The central navigation controller measures the current global state $q(t)$ at the *t*-th collaboration instant. If $q(t)$ is the goal state q', end; else, move on to the step 3;
3. The central controller solves the optimization problem (5) and obtains the optimal solution $Q^*(H|t)$;
4. Each element of the first sample $q^*(1|t)$ of $Q^*(H|t)$ is transferred to the corresponding robot as the *control goal* for the next instant $t+1$;
5. Upon receiving the *control goal* $q_i(t+1)$, the dynamic controller of the *i*-th robot solves the optimization problem (6) to obtain the optimal solution $\tilde{u}_i^*(t)$, $i = 1, ..., N$;
6. The *i*-th robot applies input sequence $\tilde{u}_i^*(t)$, $i=1, ..., N$.
7. Increase *t*: $t := t+1$. Go to the step 2.

4 Feasibility and Stability

Consider the HMPC optimization problems (5) and (6). Note that as long as the control goal of the *i*-th robot is determined by (5), the solution of (6) ensures that the control goal is achieved. It implies that the feasibility of HMPC is determined by its central controller. Since the central controller knows the complete state of the whole system and solves the optimization problem that contains the complete collision avoidance specification, feasibility is automatically assured.

Theorem 1. *The* HMPC *scheme is feasible.*

Stability is a key issue in MPC. The terminal constraint set and cost function methods are widely used to ensure the stability of traditional MPC approaches. Different terminal constraint sets and cost function methods may differ from each other in the setting of parameters, but they share the same basic idea – that is to make the optimal cost sequence computed at all sampling instants (the collaboration instants in our setting) monotonically decreasing and convergent to origin. Based on the idea, a sufficient condition is given for MPC stability [Mayne *et al.*, 2000]: a MPC is stable, if its chosen parameters (namely, a terminal cost l_H, a terminal constraint set $\tilde{\mathcal{Q}}_f$ and a controller function $\kappa(\cdot)$) satisfy:

A1 $q' \in \tilde{\mathcal{Q}}_f : \tilde{\mathcal{Q}}_f \subseteq \{q | h(q) \le 0\}$, $\tilde{\mathcal{Q}}_f$ closed
A2 $\forall q(t) \in \tilde{\mathcal{Q}}_f$: $\kappa(q(t)) \in \tilde{\mathcal{U}}$
A3 $\forall q(t) \in \tilde{\mathcal{Q}}_f$: $q(t+1) \in \tilde{\mathcal{Q}}_f$
A4 $\forall q(t) \in \tilde{\mathcal{Q}}_f$: $l_H(q(t+1) - l_H(q(t)) \le -l(q(t), u(t))$

where $q(t+1)$ denotes the state at $t+1$ computed from $q(t)$ and the input $\kappa(q(t))$.

Intuitively they signify that : Q_f satisfies the collision avoidance constraints (**A1**), the input generated by the controller κ satisfies the input constraints (**A2**) and makes the state $q(t+1)$ to stay in Q_f (**A3**), and that the terminal cost monotonically decreases from $q(t)$ to $q(t+1)$ in Q_f (**A4**).

Since centralized MPCs (CMPCs) deal with the multi-robot system as a whole, the above-mentioned sufficient condition ensures the stability of CMPCs. In what follows, we show that any setting of parameters that meet the properties (**A1**-**A4** in a specified CMPC) can also ensure the stability of HMPC.

Theorem 2. *If a given set of parameters $l_H, \tilde{\mathcal{Q}}_f, \kappa_f(\cdot)$ satisfy assumptions (A1-A4) for a specified* CMPC*, the same parameter setting also ensures the stability of* HMPC.

Proof. Treating the multi-robot system as a whole like CMPC, we construct the following CMPC problem:

$$\left.\begin{array}{l} J(q(t)) \triangleq \min\limits_{U(H|t)} (\sum\limits_{k=0}^{H-1} l(q(k|t), \tilde{u}(k|t)) + l_H(q(H|t))) \\ \texttt{s.t.} \quad q(k+1|t) = A\, q(k|t) + B\, \tilde{u}(k|t),\ k = 1, ..., H-1, \\ \qquad \tilde{u}(k|t) \in \tilde{\mathcal{U}}_1 \times ... \times \tilde{\mathcal{U}}_N,\ k = 1, ..., H-1, \\ \qquad h(q_{k|t}) \le 0,\ k = 1, ..., H-1, \\ \qquad q(0|t) = q(t), \quad q(H|t) \in \tilde{\mathcal{Q}}_f, \end{array}\right\} \quad (7)$$

where

$$A = \begin{bmatrix} A_1^{K_1} & & \\ & \ddots & \\ & & A_N^{K_N} \end{bmatrix} \quad B = \begin{bmatrix} A_1^{K_1-1} B_1 ... B_1 & & \\ & \ddots & \\ & & A_N^{K_N-1} B_N ... B_N \end{bmatrix}$$

Provided that $l_H, \tilde{\mathcal{Q}}_f, \kappa_f(\cdot)$ satisfy assumptions (A1-A4) for (7). We need to prove $l_H, \tilde{\mathcal{Q}}_f, \kappa_f(\cdot)$ satisfy (A1-A4) for (5).

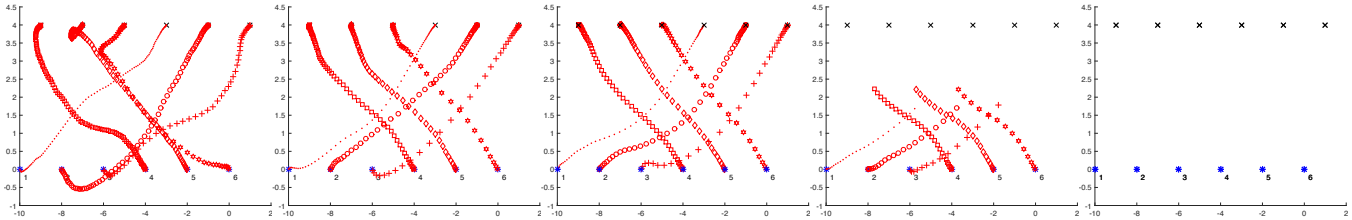
Note that (1) is equivalent to the reachable set expression in the optimization problem (5). So the difference between two problems (7) and (5) is the absence of the input cost l_u in (5). So A1, A2 and A3 are obviously satisfied for (5). A4 is satisfied for (5) due to that we have $\forall q(t) \in \tilde{\mathcal{Q}}_f$,

$$\begin{array}{ll} l_H(q(t+1)(q(t), \kappa(q(t))) - l_H(q(t)) & \\ \le -l(q(t), \tilde{u}(t)) & \text{(A4 holds for (7))} \\ = -l_q(q(t)) - l_u(\tilde{u}(t)) & \text{(defn of cost func.)} \\ \le -l_q(q(t)) & (l_u(\tilde{u}) \ge 0, \forall \tilde{u}) \end{array}$$

Notice that each robot is ensured by (6) to reach its control goal computed by the central controller in each collaboration cycle. Stability provided by the central controller in turn guarantees that the HMPC scheme is stable. □

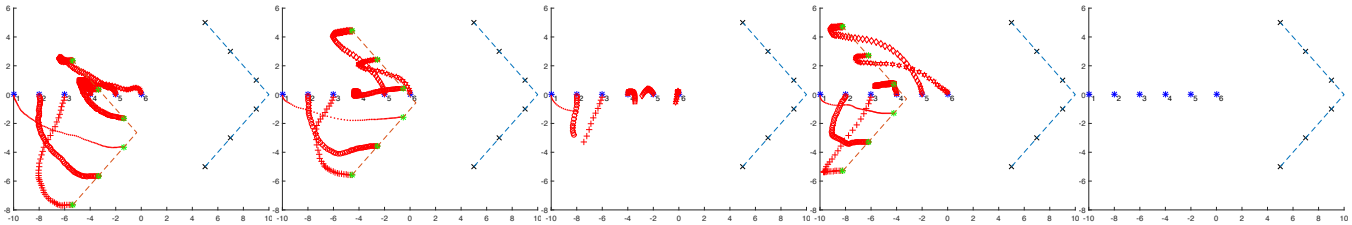
5 Simulations

We have implemented the proposed HMPC and other MPC schemes, including centralized MPC (CMPC), basic distributed MPC BDMPC) ([Keviczky *et al.*, 2004b], sequential distributed MPC (SDMPC) [Kuwata and How, 2011], and iterative distributed MPC (IDMPC) [Mercangöz and Doyle III, 2007]. We conduct a number of simulations in the multi-robot navigation problem, and compare them in terms of effectiveness and efficiency. Notably, the problem studied here involves non-convex constraints and the neighbourhood topology of each robot can change at any collaboration instant, and none of the above-mentioned distributed approaches can



(a) HMPC (b) CMPC (c) BDMPC (d) SDMPC (e) IDMPC

Figure 1: Trajectories of robots in the stationary formation scenario



(a) HMPC (b) CMPC (c) BDMPC (d) SDMPC (e) IDMPC

Figure 2: Trajectories of robots in the moving aggregation scenario

guarantee the stability and feasibility for such a problem. While we currently do not consider stationary obstacles and constraints on robots' states, they can be incorporated into HMPC using coordination constraints in the same way as collision avoidance constraints.

5.1 Stationary Formation

Let us consider two scenarios, namely stationary formation and moving aggregation, with 6 robots to compare the effectiveness of these MPC schemes. We simulate the five MPC schemes and report some interesting results.

The parameter setting in the simulation is as follows. First, the sampling interval of *i*-th robot is defined:

$$\textit{sampling interval of i-th robot} = \begin{cases} 0.075, & i \bmod 3 \equiv 1, \\ 0.1, & i \bmod 3 \equiv 2, \\ 0.15, & i \bmod 3 \equiv 0. \end{cases}$$

The dynamics (specified by Eq. (1a), (1b)) of the *i*-th robot is obtained by discretizing a double integrator at its sampling frequency. The input constraint of the *i*-th robot is:

$$\mathcal{U}_i = \{u : |u| \leq [2, 2]^T\}$$

The safety distance is $d_{safe} = 0.6$. In practice, most works [Rawlings and Muske, 1993; Sznaier and Damborg, 1987] advocate choosing the horizon H online or big enough so that the optimal state $q(H|t)$ obtained by solving problem (5) *without the terminal constraint* actually satisfies $q(H|t) \in \tilde{\mathcal{Q}}_f$. Also, a large enough weight of l_H can ensure Assumption **A4** to be met. Accordingly we give the configuration of parameters to make CMPC stable (hence, so is our HMPC):

$$l_H(q) = 10 l_q(q); \quad \tilde{\mathcal{Q}}_f = \mathbb{R}^4; \quad H = 9$$

$p = 2$ in the cost function, i.e. l is quadratic. Other parameters will be chosen differently for each scenario.

To simulate the two scenarios, (5) can be rewritten as mix integer programming and we use the CVX - a Matlab-based package - [CVX Research, 2012; Grant and Boyd, 2008] for solving optimization problems in multiple MPC schemes.

Stationary formation is a basic navigation scenario, where each robot moves from a stationary initial position to a stationary goal position. From the initial states of the robots, we intuitively assume robot *i* has neighbour *i-1* and *i+1*. Weighting factors in the cost function are chosen to be

$$C_{abs} = C_{rel} = I_4, \quad C_u = 0.01 I_2.$$

where $I_n \in \mathbb{R}^n$ denotes the *n*-dimension identity matrix.

We show the trajectories computed by five MPC schemes in Fig 1. Blue *'s denote the initial positions, black x's denote the goal positions, and other shapes denote the trajectories of robots respectively. CMPC and HMPC successfully complete the formation, as shown in Fig 1b and 1a. BDMPC also finishes the simulation while SDMPC and IDMPC fail.

In the simulation of SDMPC, as shown in Fig 1d, the distance between the *1*st robot and the *5*th becomes 0.5371 at the *7*-th collaboration instant, which is less than the given safety distance 0.6. The reason is due to the assumption that the neighbourhood topology for each robot remains unchanged in DMPC schemes [Keviczky *et al.*, 2004a]. The topology structure seems appropriate in the beginning of the simulation. However, we can see that at the *7*-th collaboration instant, the *1*st robot and the *5*th robot become neighbours. The unexpected change of the topology structure causes such a feasibility problem. A potential improvement is to update the topology structure at each collaboration instant. Notably, it cannot settle the problem as the topology may even change unexpectedly when robots are predicting their future actions.

In the simulation of IDMPC, as shown in Fig 1e, the robots cannot even make the first step. Actually, at the first collab-

oration instant, the prediction for the *3*rd robot in the *2*nd iteration is infeasible, which means no feasible solution exists when the *3*rd robot receives the corresponding prediction values from its neighbours. Since each robot predicts its own dynamics only based on its neighbours, the choice made by some robots may be too aggressive for others.

5.2 Moving Aggregation

Moving aggregation is relatively complex since the initial and goal velocities are both non-zero, where the goal states indicate that the desired formation is an arrow shape. In detail, the initial and the goal velocity vectors of each robot are $[0, -1]$ and $[1, 0]$, respectively. The topological matrix remains the same as the stationary aggregation scenario. Different from the stationary formation, the moving aggregation scenario excludes the goal position from the cost function by setting the weighting factor of absolute position to 0:

$$C_{abs} = \begin{bmatrix} 0 & 0 \\ 0 & I_2 \end{bmatrix}, \ C_{rel} = I_4, \ C_u = 0.01 I_2.$$

The trajectories computed by five MPC schemes are shown in Fig 2. CMPC and HMPC still successfully complete the formation as shown in Fig 2b and 2a. However, only SDMPC finishes the simulation among three DMPCs.

In the simulation of BDMPC, as shown in Fig 2c, the distance between the *4*th and *5*th robots becomes 0.5823 at the *5*th collaboration instant, which is less than the given safety distance 0.6. The reason is that in BDMPC, the *i*-th robot only computes its action and predicts the behavior of its neighbours once at a collaboration instant. However, these neighbours often make different choices from what the *i*-th robot predicts since they may have different neighbourhoods. When two adjacent robots both make wrong predictions on each other's behavior, a collision may occur.

In the simulation of IDMPC, as shown in Fig 2e, the robots cannot make the first step. This time, at the first collaboration instant, the iteration process cannot stop since it is non-convergent. In fact, the iteration process runs for 30 minutes and the number of iterations is more than 90 before we stop it, while CMPC and HMPC both take less than 10 seconds at each collaboration instant. Although IDMPC can work well in the scenarios where constraints are all convex, it still needs further study to ensure the convergence of its iteration process in the scenarios with non-convex constraints.

Overall, the simulation in two scenarios shows that HMPC is as effective as CMPC, while most popular DMPC schemes have some troubles for the problem.

5.3 Efficiency

We simulate the stationary formation scenario with different numbers of robots N=$3, 6, 9, 12, 15, 18, 21, 24$ for the efficiency comparison purpose. We set $p = 1$ in the cost function, i.e. l is linear. The performance of multiple MPC schemes is intuitively presented in Fig 3. Only if a MPC scheme finishes a certain simulation scenario, the average computation time at each collaboration instant in terms of the scenario is plotted. Although BDMPC is faster in the first two small-scale problems, few DMPC schemes can finish the simulation due to various problems encountered, including what we have discussed in the previous sub-section. HMPC and CMPC are the only two survivors under all the scenarios. It is empirically verified that HMPC is more efficient than CMPC from Fig 3.

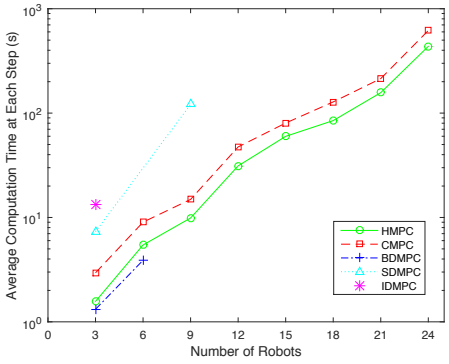


Figure 3: Performance of five control strategies. The incomplete lines are due to the method failure in complex scenarios.

The principle behind is more worthy of discussion. Since there has been no explicit conclusion on complexity of Mixed Integer Programming so far, we conduct the complexity analysis based on the number of variables and constraints used by CMPC and HMPC. By replacing the state equations and input constraints, input variables and intermediate state variables treated by a central controller in CMPC with the reachable set, the problem computer by the centralized controller of HMPC is much simpler. Furthermore, the optimization problem in each robot restricts to local state equations and input constraints in HMPC, whose scale is small. And all the problems are solved concurrently. So it is not surprised that HMPC requires much less time than CMPC.

When applied to real world problems, if the time given to the controller synthesis is too short, similar to existing heuristic-search methods, HMPC will let robots stop and wait at each collaboration instant, until the controller is derived. Also, the computation time can be further reduced by removing certain constraints in the central optimization problem. For example, collision avoidance constraints between robots that are not close to each other may be safely removed.

6 Conclusion

We propose a hierarchical Model Predictive Control (HMPC) for the navigation problem of linear dynamical multi-robot systems with no reference trajectory. Using reachable set function computed off-line, our proposed HMPC scheme can successfully decouple constraints in objective functions and constraints, thus allowing computation to be distributed to the local controller of each robot and to run in parallel. Therefore, HMPC is computationally more efficient than CMPC as evidenced by our simulation. On the other hand, while the DMPC schemes cannot always ensure feasibility and stability, HMPC can guarantee its feasibility and stability in the same way as CMPC, therefore can be more viable, which is also evidenced by our simulation. In essence, the proposed HMPC combines the merits from both CMPC and DMPC schemes. An immediate future work is to expand HMPC for multi-robot systems with non-linear dyamics.

Acknowledgement

This research is supported by National Key Basic Research Program of China (2014CB340703), National Natural Science Foundation of China (No.61561146394, No.61321491, No.61373033), Specialized Research Fund for the Doctoral Program of Higher Education (20110091120058), Project on the Integration of Industry, Education and Research of Jiangsu Province (BY2014126-03), SZSTI Project JCYJ201418193546117 and also by Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

[Antonelli *et al.*, 2013] G. Antonelli, F. Arrichiello, F. Caccavale, and A. Marino. Decentralized centroid and formation control for multi-robot systems. In *Proceeedings of 2013 IEEE International Conference on Robotics and Automation (ICRA2013)*, pages 3511–3516, May 2013.

[Bourdais *et al.*, 2014] R. Bourdais, J. Buisson, D. Dumur, H. Guéguen, and P-D. Moroşan. *Distributed Model Predictive Control Made Easy*, chapter Distributed MPC Under Coupled Constraints Based on Dantzig-Wolfe Decomposition, pages 101–114. Springer Netherlands, Dordrecht, 2014.

[Calliess *et al.*, 2014] J.-P. Calliess, M. A. Osborne, and S. J. Roberts. Conservative collision prediction and avoidance for stochastic trajectories in continuous time and space. In *Proceedings of the 2014 International conference on Autonomous Agents and Multi-Agent Systems, AAMAS '14, Paris, France, May 5-9, 2014*, pages 1109–1116, 2014.

[CVX Research, 2012] Inc. CVX Research. CVX: Matlab software for disciplined convex programming, version 2.0. http://cvxr.com/cvx, aug 2012.

[Dunbar and Murray, 2002] W. B Dunbar and R. M Murray. Model predictive control of coordinated multi-vehicle formations. In *Proceedings of the 2002 IEEE Conference on Decision and Control*, volume 4, pages 4631–4636, 2002.

[Godoy *et al.*, 2015] J. E. Godoy, I. Karamouzas, S. J. Guy, and M. L. Gini. Adaptive learning for multi-agent navigation. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems (AAMAS), Istanbul, Turkey, May 4-8, 2015*, pages 1577–1585, 2015.

[Gondhalekar *et al.*, 2009] R. Gondhalekar, J. Imura, and K. Kashima. Controlled invariant feasibilityła general approach to enforcing strong feasibility in mpc applied to move-blocking. *Automatica*, 45(12):2869–2875, 2009.

[Grant and Boyd, 2008] M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. http://stanford.edu/~boyd/graph_dcp.html.

[Gritzmann and Sturmfels, 1993] P Gritzmann and B Sturmfels. Minkowski addition of polytopes: computational complexity and applications to gröbner bases. *SIAM Journal on Discrete Mathematics*, 6(2):246–269, 1993.

[Herceg *et al.*, 2013] M. Herceg, M. Kvasnica, C.N. Jones, and M. Morari. Multi-Parametric Toolbox 3.0. In *Proceedings of the European Control Conference*, pages 502–510, Zürich, Switzerland, July 17–19 2013. http://control.ee.ethz.ch/~mpt.

[Janovsky *et al.*, 2014] P. Janovsky, M. Cap, and J. Vokrinek. Finding coordinated paths for multiple holonomic agents in 2-d polygonal environment. In *Proceedings of the 2014 International conference on Autonomous Agents and Multi-Agent Systems, AAMAS '14, Paris, France, May 5-9, 2014*, pages 1117–1124, 2014.

[Keviczky *et al.*, 2004a] T. Keviczky, F. Borrelli, and G. J Balas. Hierarchical design of decentralized receding horizon controllers for decoupled systems. In *Proceedings of the 43rd IEEE Conference on Decision and Control*, volume 2, pages 1592–1597. IEEE, 2004.

[Keviczky *et al.*, 2004b] T. Keviczky, F. Borrelli, and G. J Balas. A study on decentralized receding horizon control for decoupled systems. In *Proceedings of the 2004 American Control Conference*, volume 6, pages 4921–4926. IEEE, 2004.

[Kuwata and How, 2011] Y. Kuwata and J. P How. Cooperative distributed robust trajectory optimization using receding horizon milp. *IEEE Transactions on Control Systems Technology*, 19(2):423–431, 2011.

[Mayne *et al.*, 2000] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. OM Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[Mercangöz and Doyle III, 2007] M. Mercangöz and F. J Doyle III. Distributed model predictive control of an experimental four-tank system. *Journal of Process Control*, 17(3):297–308, 2007.

[Olfati-Saber *et al.*, 2003] R. Olfati-Saber, W. B Dunbar, and R. M Murray. Cooperative control of multi-vehicle systems using cost graphs and optimization. In *Proceedings of the 2003 American Control Conference*. Citeseer, 2003.

[Rawlings and Muske, 1993] J B Rawlings and K R Muske. The stability of constrained receding horizon control. *IEEE Transactions on Automatic Control*, 38(10):1512–1516, 1993.

[Rosales *et al.*, 2011] A. Rosales, G. Scaglia, V. Mut, and F. di Sciascio. Formation control and trajectory tracking of mobile robotic systems–a linear algebra approach. *Robotica*, 29(03):335–349, 2011.

[Sznaier and Damborg, 1987] M. Sznaier and M.J. Damborg. Suboptimal control of linear systems with state and control inequality constraints. In *Proceedings of 26th IEEE Conference on Decision and Control*, volume 26, pages 761–762, Dec 1987.

[Tian and Sarkar, 2012] Y Tian and N Sarkar. Formation control of mobile robots subject to wheel slip. In *Proceeedings of 2012 IEEE International Conference on Robotics and Automation (ICRA2012)*, pages 4553–4558. IEEE, 2012.