Event-Triggered Polynomial Model Predictive Control for Multi-Agent Navigation

Harini V[1], Anusree Rajan[2], and Pavankumar Tallapragada[1]

***Abstract*— This paper proposes an event-triggered polynomial model predictive control method for collision-free point-to-point multi-agent navigation. In this control method, each control input to each agent is a polynomial whose coefficients are updated in an event-triggered manner. For each agent, we design an event-triggering rule that guarantees non-Zeno behavior of inter-event times. At each event, the controller updates the coefficients of the polynomial control law corresponding to a subset of agents by solving one or more finite horizon optimization problems. We also ensure feasibility of the optimization problems solved at each event. Through numerical simulations, we illustrate the results and compare the proposed method with other existing methods.**

I. Introduction

A. Motivation

Collision-free multi-agent navigation problems are very popular and widely studied in applications such as swarm control, fleet management, etc. Many of the approaches used to solve this problem typically involve solving optimization problems that are computationally expensive. Some methods require sending large amounts of information to the agents at every communication time instant, which is not desirable in networked control systems. In this work, our primary motivation is to propose a control method that generates collision-free trajectories for all agents with improved usage of communication and computation resources. With that motivation, we propose an event-triggered polynomial model predictive control method for collision-free multi-agent navigation that helps to improve the usage of communication and computation resources.

B. Literature Review

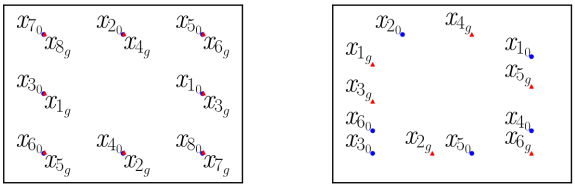
In [1] Buffered Voronoi Cells are used, which generate collision-free trajectories by ensuring that the trajectories generated stay in sets that don't intersect for a finite time horizon. In [2], an algorithm based on modified collision cone is given to avoid static/dynamic obstacles. But, this approach doesn't factor in practical limitations of any drone i.e. constraints on acceleration or velocities that translates to a constraint on heading. In [3], a state-lattice based approach is used for multi-agent motion planning of non-holonomic agents. Here the main contribution is to bring in constraints for traditional graph-based planners such as A*, RRT etc. However, it has limitations of the graph-based approach which is that as the searchable area increases, the computation time also increases.

There are also several approaches based on model predictive control [MPC] among which fewer works implement event-triggering for real-time applications reviewed in [4]–[6]. In [7], sequential convex programming is used to address the computational complexity of MPC based collision-free trajectory planning for multiple robots. However, the convex approximation of non-convex collision constraints in the optimization problem is very conservative. In [8], [9] signal temporal logic is used to generate timed waypoints where control is held constant between two waypoints. In [10]–[12], MPC based signal temporal logic is used to generate collision-free paths where states are variables of the optimization problem. However, in these types of approaches, it is difficult to work with complex dynamics. In [13] MPC is used to predict trajectory of obstacles while Deep Deterministic Policy Gradient approach is used for learning autonomous decision making for robots. In [14], deep reinforcement learning and force based motion planning are used together to generate time optimal collision-free paths. Individually, reinforcement learning is often not suitable for collision-free path generation while force based motion planning lacks optimality. In [15], a decentralized tube based MPC approach was given for collision-free multiagent trajectory generation. However, this is not implementable in real time. In [16], an event-triggered distributed MPC approach is used for on-demand collision avoidance using BVC. This approach is real-time implementable but is not suitable for high density of agents. Some other papers use convex hull properties of bezier curves for collision avoidance such as [17], [18] that reduce the computational complexity but is very conservative as collision avoidance is ensured by separation of convex hulls. In [19], event-triggered distributed MPC is used for point to point transitions of multiple drones where, replanning is done only for a select number of agents by different computational units. However, the approach here is still conservative as convex hull separation is used for collision avoidance. In [20], hyperplane separation used in convex hulls is learnt from generated expert trajectories for collision avoidance.

In [21], a swarm formation modelling is used to group multiple agents as a singleton to improve scalability while MPC is used on swarm kinematics for optimality. In [22], Non-linear MPC is used where agents other than the agent of interest are considered to be dynamic obstacles. The MPC problem is solved for each agent and then trajectory is broadcasted to all other agents. However, the predicted trajectory used for collision avoidance constraint may not be valid for any particular pair of agents due to the optimization problem being solved. Thus, it is possible that there can be conditions where agents collide such as when disturbances

[1] Robert Bosch Centre for Cyber Physical Systems, Indian Institute of Science, Bengaluru. {hariniv, pavant}@iisc.ac.in

[2] Department of Electrical and Electronics Engineering, Christ University, Bengaluru. anusree.rajan@christuniversity.in



(a) (b)

Fig. 1: x_{i_0} and x_{i_g} indicate initial and goal positions of the agents. Setups in which agents need to (a) swap, (b) cross with other agents.

are high in the practical system.

C. Contributions

The main contributions of this paper are as follows:

- This paper proposes an event-triggered polynomial model predictive control method for multi-agent navigation that generates collision-free trajectories for all agents within the operating region.
- Compared to other existing methods, the proposed method allows for parallel computation and thus reduces the computational complexity by breaking down the overall trajectory planning problem into smaller subproblems.
- Compared to the basic event-triggered model predictive control method, the proposed control method requires sending only a finite number of coefficients, rather than sending a finite horizon control trajectory, to each agent intermittently. Thus, it helps to improve the usage of communication resources in networked control systems.
- The proposed method ensures the feasibility of the optimization problems solved at each event. It also ensures that the inter-event times generated by the proposed event-triggering rule do not exhibit Zeno behavior.

D. Notation

Let $\mathbb{R}, \mathbb{N}$, and $\mathbb{N}_0$ denote the set of all real numbers, positive integers, and non-negative integers, respectively. Let the Euclidean norm of x, for $x \in \mathbb{R}^n$, be denoted as $\|x\|$.

II. Problem Setup

In this section, we present the problem setup that includes the multi-agent system dynamics, the event-triggered polynomial control law, and the objective of this paper.

A. Multi-Agent System Dynamics and Control Objective

We consider a multi-agent system that consists of $N \in \mathbb{N}$ agents with single integrator dynamics as follows,

$$\dot{x}_i(t) = u_i(t),\ \forall t \geq 0,\ \forall i \in \{1,2,\ldots,N\}, \tag{1}$$

where $x_i \in \mathbb{R}^2$ denotes the position of agent i and $u_i \in \mathbb{R}^2$ denotes the control input to agent i. Each agent i has an initial position $x_i(0) \in \mathbb{R}^2$ and a desired goal position $x_{ig} \in \mathbb{R}^2$ as illustrated in Figure 1b. For simplicity, we consider the single integrator dynamics here which can later be extended to complex non-linear dynamics.

In this paper, our primary objective is to generate collision-free trajectories for all agents while staying in a given region and under bounded velocities and accelerations so that each agent reaches its goal position as quickly as possible. The problem that we wish to *ideally* solve is the following.

$$\begin{aligned}
&\min_{\{u_i\},\ \{t_{g_i}\}} \quad \sum_{i=0}^{N} t_{g_i}, \\
&\text{s.t.}\ \ \dot{x}_i(t) = u_i(t),\ \forall t \in [0,t_{g_i}],\ \forall i \in \{0,\ldots,N\}, \\
&\|\dot{x}_i(t)\| \leq \bar{u},\ \forall t \in [0,t_{g_i}],\ \forall i \in \{0,\ldots,N\}, \\
&\|\ddot{x}_i(t)\| \leq \bar{a},\ \forall t \in [0,t_{g_i}],\ \forall i \in \{0,\ldots,N\}, \\
&\|x_i(t)\|_\infty \leq \bar{b},\ \forall t \in [0,t_{g_i}],\ \forall i \in \{0,\ldots,N\}, \\
&\left\|x_i(t) - x_j(t)\right\| \geq \varepsilon,\ \forall t \in [0,t_{g_i}],\ \forall i \neq j \in \{0,\ldots,N\}, \\
&\left\|x_i(t_{g_i}) - x_{ig}\right\| \leq \delta, \forall i \in \{0,\ldots,N\}.
\end{aligned} \tag{2}$$

In this optimization problem, t_{g_i} is the time taken by agent i to reach its goal under the control trajectory u_i and $\bar{u}, \bar{a}, \bar{b}, \varepsilon \in \mathbb{R}_{>0}$ are parameters. The constraints encode the dynamics (1), bounds on velocity and acceleration, agents staying within the bounded region $\{x \in \mathbb{R}^2 \mid \|x\|_\infty \leq \bar{b}\}$, collision avoidance between agents and the reaching of the goal defined as reaching a point that is within δ distance of x_{ig}. However, the optimization problem (2) is computationally very complicated: it has a variable terminal time, with each agent having a different terminal time, and also has non-convex collision avoidance constraints. Moreover, it is not scalable and not amenable to recomputation of trajectories online if there are disturbances or if there are additions or deletions of agents.

An approach to solve this problem is to use a model predictive control method where a finite fixed horizon optimal control trajectory is generated for each agent by solving a joint optimization problem online so as to minimize the cumulative distance to goal for all agents. This approach is typically not suitable for continuous-time systems, as it requires solving an optimization problem instantaneously. The event-triggered model predictive control method overcomes this drawback by solving the optimization problem intermittently. But, this method requires transmitting a finite-horizon control trajectory to each agent at each event, which is not desirable if the controller is communicating with the agents over a wireless communication network.

B. Event-Triggered Polynomial Control Law

In order to overcome all the challenges discussed in the previous subsection, in this paper, we consider an event-triggered polynomial control method. In this method, each control input to each agent is a polynomial of degree p whose coefficients are updated in an event-triggered manner. Specifically, for each agent i, we consider the following polynomial control law,

$$u_i(t_k^{\{i\}} + \tau) = \Phi(\tau) a_i(k),\ \forall \tau \in [0, t_{k+1}^{\{i\}} - t_k^{\{i\}}), \tag{3}$$

where

$$\Phi(\tau) := \begin{bmatrix} \phi^\top(\tau) & 0 \\ 0 & \phi^\top(\tau) \end{bmatrix} \in \mathbb{R}^{2\times 2(p+1)},$$

$\phi^\top(\tau) := \begin{bmatrix} 1 & \tau & \dots & \tau^p \end{bmatrix}$ and $a_i(.) \in \mathbb{R}^{2(p+1)}$. Here, $(t_k^{\{i\}})_{k\in\mathbb{N}_0}$ denotes the sequence of time instants at which the coefficients of the polynomial control law, $a_i(.)$, are updated.

Note that, in this control method, we allow for updating the coefficients of the polynomial control law asynchronously for all agents. Note also that, compared to the basic event-triggered model predictive control method, the proposed control method requires sending only a finite number of coefficients, rather than sending a finite horizon control trajectory, to each agent intermittently.

C. Design Objective

The main objective of this paper is to propose an algorithm that gives, for each agent i, the coefficients of the polynomial control law and an event-triggering rule that implicitly determines the time instants at which the coefficients are updated, so that the position of each agent converges to the respective goal position in the shortest possible time while maintaining a minimum distance between all agents at all times.

III. Design of Polynomial Control Law and Event-Triggering Rule

In this section, we design an algorithm that gives, for each agent i, the coefficients of the polynomial control law (3) and an event-triggering rule that implicitly determines the time instants at which the coefficients are updated.

A. Design of Polynomial Control Law

As discussed in the previous section, we can generate collision-free trajectories by jointly optimizing the coefficients of the polynomial control law for all agents. In this work, we determine the coefficients of the polynomial control law (3) for an agent i at an event-triggering instant $t_k^{\{i\}}$ by jointly optimizing the coefficients for only a subset of agents that includes i. This idea helps to break down the overall problem into smaller subproblems. This allows for parallel computation and reduces the computational complexity of finding collision-free trajectories for all agents.

However, the subset of agents for which the trajectories are jointly optimized has to be chosen carefully. Notice that it is not enough to optimize just over the set of agents that are close to agent i at time $t_k^{\{i\}}$. This is because, at time $t_k^{\{i\}}$, even though an agent k may not be close to agent i, it may be close to an agent j, which in turn is close to agent i. So, trajectories of agents i, j, k have to be jointly optimized at time $t_k^{\{i\}}$. More generally, we need to consider all such direct and multi-level indirect influences.

In order to formally present this idea, we define a time-varying undirected graph $G(t)$ with a fixed set of nodes $V := \{1, 2, \dots, N\}$ and a time-varying set of edges

$$E(t) := \{(i,j) \in V \times V \ : \ \Delta_{ij}(t) := \left\|x_i(t) - x_j(t)\right\| \leq \gamma\},$$

for some $\gamma \in \mathbb{R}_{>0}$. Now, for any agent i, let $\mathscr{P}_i(t)$ denote the connected component of $G(t)$ to which i belongs, i.e.,

$$\mathscr{P}_i(t) := \{j \in V \mid \exists \text{ a path between } i \text{ and } j \text{ in } G(t)\}.$$

Then, at an event-triggering instant $t_k^{\{i\}}$, the controller jointly determines the coefficients of the polynomial control law (3) for all agents $l \in \mathscr{P}_i(t_k^{\{i\}})$ by solving the following optimization problem,

$$\begin{aligned}
&\min_{\{a_l\}_{l\in\mathscr{P}_i(t_k^{\{i\}})}} \ \sum_{l\in\mathscr{P}_i(t_k^{\{i\}})} \int_0^{T_h} \left\|\hat{x}_l(t_k^{\{i\}}+\tau) - x_{lg}\right\|^2 d\tau, \\
&\text{s.t. } \hat{x}_l(t_k^{\{i\}}) = x_l(t_k^{\{i\}}), \ \forall l \in \mathscr{P}_i(t_k^{\{i\}}), \\
&\dot{\hat{x}}_l(t_k^{\{i\}}+\tau) = \Phi(\tau)a_l, \ \forall \tau \in [0,T_h], \ \forall l \in \mathscr{P}_i(t_k^{\{i\}}), \\
&\|\Phi(\tau)a_l\|_\infty \leq \bar{u}, \ \forall \tau \in [0,T_h], \ \forall l \in \mathscr{P}_i(t_k^{\{i\}}), \\
&\left\|\dot{\Phi}(\tau)a_l\right\|_\infty \leq \bar{a}, \ \forall \tau \in [0,T_h], \ \forall l \in \mathscr{P}_i(t_k^{\{i\}}), \\
&\left\|\hat{x}_l(t_k^{\{i\}}+\tau)\right\|_\infty \leq \bar{b}, \ \forall \tau \in [0,T_h], \ \forall l \in \mathscr{P}_i(t_k^{\{i\}}), \\
&\left\|\hat{x}_l(t_k^{\{i\}}+\tau) - \hat{x}_j(t_k^{\{i\}}+\tau)\right\|^2 \geq \varepsilon^2, \\
&\forall \tau \in [0,T_h], \ \forall l \in \mathscr{P}_i(t_k^{\{i\}}),
\end{aligned} \tag{4}$$

Here, $T_h \in \mathbb{N}$ is a design parameter.

In the optimization problem (4), the coefficients of the polynomial control law, for all agents $l \in \mathscr{P}_i(t_k^{\{i\}})$, are updated so as to minimize the cumulative distance to goal over the finite time horizon $t_k^{\{i\}} + T_h$ under the given constraints. The constraints of the optimization problem (4) impose a bound on the velocity and acceleration of each agent $l \in \mathscr{P}_i(t_k^{\{i\}})$. The last two constraints help to restrict the movement of agents within the operating region and to avoid collision between agents by imposing a minimum inter-agent distance of ε, respectively.

B. Design of Event-Triggering Rule

Next, we design an event-triggering rule that implicitly determines the time instants at which the coefficients of the polynomial control law are updated for each agent i.

We first define an indicator variable that helps in our design. For any $i \neq j \in V$,

$$\theta_{ij}(t) := \begin{cases} 1, & \text{if } l_i(t) = l_j(t) = \bar{t} \text{ and } j \in \mathscr{P}_i(\bar{t}), \\ 0, & \text{otherwise,} \end{cases}$$

where $l_i(t)$ denotes the latest time instant upto time t at which the coefficients of the polynomial control input to agent i were updated. That is,

$$l_i(t) := t_k^{\{i\}}, \ \forall t \in [t_k^{\{i\}}, t_{k+1}^{\{i\}}).$$

Note that $\theta_{ij}(t) = 1$ indicates that at time t, agents i and j are executing a plan that was jointly optimized at time $l_i(t)$.

Now, for each $i \in V$, we design the following event-triggering rule that involves two conditions,

$$t_{k+1}^{\{i\}} = \min\{t \geq t_k^{\{i\}} \mid \alpha_i(t) = 1 \text{ or } \beta_i(t) = 1\}, \tag{5}$$

where,

$$\alpha_i(t) := \begin{cases} 1, & \text{if } t - l_i(t) \geq T_h \text{ or} \\ & \text{if } \exists \ j \neq i \in V : \theta_{ij}(t) = 0 \ \& \ \Delta_{ij}(t) \leq \beta, \\ 0, & \text{otherwise,} \end{cases}$$

for some $\beta < \gamma$ and,

$$\beta_i(t) := \begin{cases} 1, & \text{if } \exists j \neq i \in \mathscr{P}_i(t) \ : \ \alpha_j(t) = 1, \\ 0, & \text{otherwise.} \end{cases}$$

Here, the primary triggering condition $\alpha_i(t) = 1$ implies that either the time elapsed since the last event for agent i is greater than or equal to T_h, or there exists an agent $j \neq i$ such that i and j are not executing a joint plan at time t and the distance between these two agents is less than or equal to β. The secondary triggering condition $\beta_i(t) = 1$ implies that there exists an agent $j \neq i \in \mathscr{P}_i(t)$ for which the primary triggering condition is satisfied. At any time instant, if either one of these two conditions are satisfied then the controller updates the coefficients of the polynomial control law for all agents $l \in \mathscr{P}_i(t)$ by solving the joint optimization problem (4).

Remark 1. *According to the event-triggering rule (5), for any agent $i \in V$, $t_{k+1}^{\{i\}} - t_k^{\{i\}} \leq T_h$ for all $k \in \mathbb{N}_0$.* •

C. Overall Algorithm

Algorithm 1 presents the overall method that gives, for each agent i, the coefficients of the polynomial control law (3) and an event-triggering rule that implicitly determines the time instants at which the coefficients are updated.

Algorithm 1: Overall algorithm for the proposed event-triggered polynomial control method

Input: $x_i(0),\ x_{ig},\ \forall i \in V$
Output: $\{t_k^{\{i\}}\}_{k\in\mathbb{N}_0},\ \{a_i(k)\}_{k\in\mathbb{N}_0},\ \forall i \in V$
Initialize: $t = 0,\ t_0^{\{i\}} = 0, k_i = 0,\ \forall i \in V,$
determine $a_i(0)$ by solving (4), $\forall i \in V$

for $t > 0$ **do**
  Initialize $\text{Flag}_i(t) = 0$, for all $i \in V$
  for $i \in V : \text{Flag}_i(t) = 0$ **do**
    if $\alpha_i(t) = 1$ **then**
      Compute $\mathscr{P}_i(t)$
      $k_l = k_l + 1,\ t_{k_l}^{\{l\}} = t,\ \forall l \in \mathscr{P}_i(t)$
      $\text{Flag}_l(t) = 1,\ \forall l \in \mathscr{P}_i(t)$
      Solve (4) to get $a_l(k_l),\ \forall l \in \mathscr{P}_i(t)$
    end
  end
end

According to Algorithm 1, at every time instant t, the controller either solves no optimization problem or it solves one or more optimization problems for smaller subsets of agents rather than solving a single optimization problem for all agents. Notice that for an optimization problem to be solved at t, there needs to be at least one agent i for which the primary triggering condition is satisfied. Then, the graph $G(t)$ and the connected component that i belongs to are determined and the optimization problem (4) is solved only once for the connected component $\mathscr{P}_i(t)$.

IV. Analysis of the Proposed Control Method

In this section, we analyze the performance of the proposed control method. Specifically, we analyze the feasibility and recursive feasibility of the optimization problem (4) and the non-Zeno behavior of inter-event times generated by (5).

We first make the following assumption on the initial and desired goal positions of all agents.

(A1) For all $i \in V$, $\|x_i(0)\|_\infty \leq \bar{b}$ and $\|x_{ig}\|_\infty \leq \bar{b}$. $\forall i \neq j \in V$, $\|x_i(0) - x_j(0)\| \geq \varepsilon$ and $\|x_{ig} - x_{jg}\| \geq \varepsilon$.

This assumption just means that the initial and goal positions of all agents are within the operating region and that all agents' initial and goal positions are non-colliding.

Now, we present a couple of propositions which help to analyze the feasibility of the optimization problem (4).

Proposition 2. *At any time instant $t \geq 0$, for any connected component $C(t)$ of $G(t)$, the optimization problem (4) is feasible if $\|x_i(t)\|_\infty \leq \bar{b},\ \forall i \in C(t)$ and $\|x_i(t) - x_j(t)\| \geq \varepsilon$, $\forall i \neq j \in C(t)$.*

Proof. Note that, under the given conditions, the optimization problem (4) is feasible as $a_i = 0,\ \forall i \in C(t)$ is a solution for the same. □

The proof just relies on the fact that just staying at their starting locations is a feasible solution for the agents.

Proposition 3. *Consider the multi-agent system (1) under the polynomial control law (3), whose coefficients are updated by solving the optimization problem (4) at the event-triggering instants determined by the triggering rule (5). Let $\varepsilon < \beta < \gamma$. Then, for any $i \in V$ and for any $k \in \mathbb{N}_0$,*

- *If $\left\|x_i(t_k^{\{i\}})\right\|_\infty \leq \bar{b}$ then $\|x_i(t)\|_\infty \leq \bar{b},\ \forall t \in [t_k^{\{i\}}, t_{k+1}^{\{i\}}]$.*
- *If $\left\|x_i(t_k^{\{i\}}) - x_j(t_k^{\{i\}})\right\| \geq \varepsilon,\ \forall i, j \in V,\ i \neq j$ then $\|x_i(t) - x_j(t)\| \geq \varepsilon,\ \forall i, j \in V,\ i \neq j,\ \forall t \in [t_k^{\{i\}}, t_{k+1}^{\{i\}}]$.*

Proof. The proof of this result follows directly from the polynomial control law (3), Proposition 2, Remark 1, and the fact that the position of each agent evolves continuously in time. □

Remark 4. *Under Assumption (A1), Propositions 2 and 3 imply that the optimization problem (4) is recursively feasible.* •

Next, we show that the inter-event times generated by the event-triggering rule (5), $\forall i \in V$, do not exhibit Zeno behavior.

Lemma 5. *Consider the multi-agent system (1) under the polynomial control law (3), whose coefficients are updated by solving the optimization problem (4) at the event-triggering instants determined by the triggering rule (5). Let $\varepsilon < \beta < \gamma$. Then, the inter-event times generated by the event-triggering rule (5), $\forall i \in V$, do not exhibit Zeno behavior.*

Proof. According to the event-triggering rule (5), for any $i \in V$, an event is triggered at time $t \geq t_k^{\{i\}}$ either if $\alpha_i(t) = 1$ or if $\beta_i(t) = 1$. Let us first suppose that an event occurs at time $t \geq t_k^{\{i\}}$ due to the primary triggering condition, that is $\alpha_i(t) = 1$. Note that, $\alpha_i(t) = 1$ indicates that either $t - t_k^{\{i\}} \geq T_h$ or there exists a $j \neq i \in V$ such that $\Delta_{ij}(t_k^{\{i\}}) > \gamma$ and $\Delta_{ij}(t) \leq \beta$. This implies that $t - t_k^{\{i\}}$ is either lower bounded by T_h or by the time taken by $\Delta_{ij}^2(.)$ to decrease from γ^2 to β^2. Using the constraint on the velocity of each agent, we can say that $\left\|\frac{d}{dt}\Delta_{ij}^2(t)\right\| \leq m := 4\sqrt{2}\gamma\bar{u}$, for all t such that $\Delta_{ij}(t) \leq \gamma$. Thus, we can say that $t - t_k^{\{i\}} \geq \min\{T_h, \frac{\gamma^2-\beta^2}{m}\}$. This implies that,

if an event occurs due to the primary triggering condition, the inter-event time is lower bounded by a positive value.

Now, let us analyze the case where an event occurs at time $t \geq t_k^{\{i\}}$ due to the secondary triggering condition, that is $\beta_i(t) = 1$ but $\alpha_i(t) = 0$. Note that, $\beta_i(t) = 1$ implies that $\exists j \neq i \in \mathscr{P}_i(t)$ such that $\alpha_j(t) = 1$. Here, we only have to analyze a special subcase whereas all the other subcases can be addressed using similar arguments as in the previous case. Specifically, we consider the subcase where $\exists j \neq i \in \mathscr{P}_i(t)$ such that $j \notin \mathscr{P}_i(t_k^{\{i\}})$ and $\alpha_j(t) = 1$. Then, we can say that $\Delta_{lj}(t_k^{\{i\}}) > \gamma$ for all $l \in \mathscr{P}_i(t_k^{\{i\}})$ and $\exists \bar{l} \in \mathscr{P}_i(t_k^{\{i\}})$ such that $\Delta_{\bar{l}j}(t) \in (\beta, \gamma)$. In that case, $t - t_k^{\{i\}}$ can be arbitrarily small. However, any agent $j \neq i$ involved in a joint planning with i does not cause the occurrence of such an event, at least for a finite time interval after the joint planning is done. As we have only finite number of agents, using the above fact, we can guarantee that the number of such events in a finite time interval is always finite. Thus, we can ensure the absence of Zeno behavior. This completes the proof of this result. □

Now, we present the main theorem of this paper.

Theorem 6. *Consider the multi-agent system* (1) *under the polynomial control law* (3)*, whose coefficients are updated by solving the optimization problem* (4) *at the event-triggering instants determined by the triggering rule* (5)*. Let* $\varepsilon < \beta < \gamma$*. Then, under Assumption* ***(A1)****,*

- $\|x_i(t)\|_\infty \leq \bar{b}$, $\forall i \in V$, $\forall t \geq 0$.
- $\left\|x_i(t) - x_j(t)\right\| \geq \varepsilon$, $\forall i \neq j \in V$, $\forall t \geq 0$.

Proof. Proof of this theorem follows directly from Proposition 2, Proposition 3, and Lemma 5. □

Theorem 6 shows that the proposed control method generates collision-free trajectories for all agents and it restricts the movement of agents within the operating region. Note that, using the proposed control method, the agents would reach their goal positions if the operating region is large enough compared to ε and if the number of robots N is small enough. We illustrate this idea using numerical simulations in the next section.

V. Simulations

We consider two different scenarios for evaluating the performance of the proposed method and also for comparing it with other existing methods. The following values of design parameters are used throughout the simulation. $\varepsilon = 0.5$ m, $\beta = 0.8$ m, $\gamma = 1$ m, $p = 2$, and $T_h = 0.7$ s. The sampling time is 1 ms, goal bound, $\delta = 0.02$ m, maximum allowable values of acceleration and velocity (absolute) was 0.5 m^2/s and 1 m/s respectively. The agents are assumed to be circular and have a radius of 0.25 m.

For generating the test cases for evaluations, we consider both of the scenarios given in Figures 1a and 1b. We generate a total of 20 test cases for 2 agent, 20 cases for 4 agents and 20 cases for 6 agents by varying the arena size, initial and goal positions of the agents. We compare our approach with popular graph-based path planning methods namely: Conflict Based Search [CBS], sequential planning based on A*, D*, Rapidly Exploring Random Trees [RRT] and Jump Point Search [JPS], where the paths of the agents are planned sequentially with the paths of the previously planned agents are taken as obstacles for the next agent. We also compare our approach with the Fixed Path [FP] method where the paths between an agent and its goal is a straight line. Additionally, we also consider the MPC approach used in the paper [19] to be called as Distributed MPC [DMPC] henceforth and a suite of methods based on MPC approach given in Table I for further comparisons

TABLE I: Summary of MPC Algorithms

Name	Polynomial Approximation	Event Triggering	Parallel Computation
V1	No	No	No
V2	No	Yes	Yes
V3	Yes	No	Yes
V4	Yes	Yes	Yes

To evaluate the performance of different methods, we consider the following performance metrics. The convergence time to goal (if all agents converge to the goal bound) is defined as

$$t_g = \min\{t \geq 0 | \left\|x_i(t) - x_{ig}(t)\right\| \leq \delta, \forall i \in V\}.$$

Let $T_{computation}$ be the total time taken for solving all optimization problems and any other algorithm specific features like computing the graph in our algorithm or computing the priority in DMPC. Note that computation time can be found only in the cases where convergence has occurred.

Figure 2 shows the absolute number of cases where all agents have converged to goal. It is evident that sequential graph-based algorithms like A*, D*, RRT, JPS and FP methods have very low success rates. It is also clear that the number of cases where all agents converge to goal are highest for CBS followed by MPC based methods. We also observe that CBS is a path planning algorithm. Additionally, there are no kinematic constraints considered while finding the paths. Due to this, the algorithm could generate infeasible trajectories. Thus, for further comparisons, we only consider a subset of algorithms based on MPC.

Figure 3b shows the convergence time for MPC based algorithms. It is observed that as the number of agents increases, the convergence time for DMPC increases due to conservative collision avoidance constraints and increased density of agents. Similar behaviour is also observed for V2, V3, V4 methods due to increased density and suboptimality introduced by our algorithm. However, for V1, there is no significant increase in the convergence time with increase in the number of agents, but, the computation time requirement scales exponentially as seen in Figure 3a.

Figure 3a shows the total computation time for MPC based algorithms. It is seen that the computation time for all MPC methods except DMPC increases with increasing number of agents due to the non-convex nature of collision avoidance constraint. Also, observe that using our triggering rule reduces the overall computation time as seen with reduction in computation time between V1 and V2. We also observe that using polynomial control reduces computation time as the number of optimization variables are decreased as seen in V1 and V3 methods. The usage of connected component idea reduces the number of agents for whom joint planning is done, which in turn decreases the computation

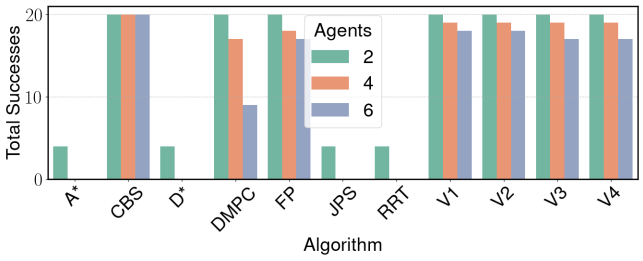
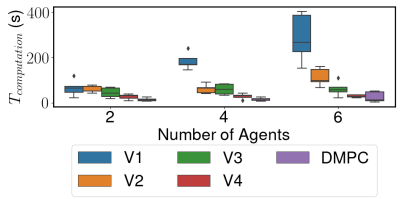
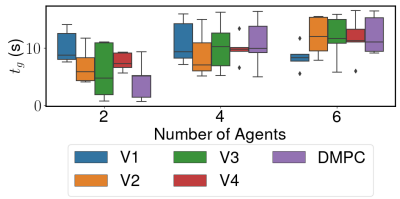


Fig. 2: Comparison of total success for all algorithms



(a)



(b)

Fig. 3: Comparison of (a) Total computation time, (b) Time to reach goal for all agents for selected algorithms.

time as seen between V3 and V4 particularly when many agents are involved.

VI. Conclusion

In this paper we presented an event triggered polynomial control method to handle point-to-point navigation for multiple agents. The proposed approach guarantees recursive feasibility of the optimization problem and non-Zeno behavior of the triggering rule. We have also compared our method with other candidate approaches by means of simulations. Future work includes finding a better approximation for the collision avoidance constraint to address scalability in highly congested scenarios and incorporation of disturbance in the MPC framework.

Acknowledgement

The authors would like to thank Prof. Bharadwaj Amrutur, Indian Institute of Science, Bengaluru, for his insightful comments on the problem.

References

[1] D. Zhou, Z. Wang, S. Bandyopadhyay, and M. Schwager, "Fast, online collision avoidance for dynamic vehicles using buffered voronoi cells," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1047–1054, 2017.

[2] M. Gnanasekera and J. Katupitiya, "A time optimal reactive collision avoidance method for uavs based on a modified collision cone approach," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 5685–5692.

[3] M. Cirillo, T. Uras, and S. Koenig, "A lattice-based approach to multi-robot motion planning for non-holonomic vehicles," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 232–239.

[4] A. da C Vangasse, E. JR Freitas, G. V Raffo, and L. CA Pimenta, "Safe navigation on path-following tasks: A study of mpc-based collision avoidance schemes in distributed robot systems," *Journal of Intelligent & Robotic Systems*, vol. 110, no. 4, pp. 1–17, 2024.

[5] J. Alonso-Mora, T. Naegeli, R. Siegwart, and P. Beardsley, "Collision avoidance for aerial vehicles in multi-agent scenarios," *Autonomous Robots*, vol. 39, pp. 101–121, 2015.

[6] M. Harun, S. Abdullah, M. Aras, M. Bahar, and F. Ali@Ibrahim, "Recent developments and future prospects in collision avoidance control for unmanned aerial vehicles (uavs): A review," *International Journal of Robotics and Control Systems*, vol. 4, no. 3, pp. 1207–1242, 2024. [Online]. Available: https://pubs2.ascee.org/index.php/IJRCS/article/view/1482

[7] Y. Zhou, H. Hu, Y. Liu, S.-W. Lin, and Z. Ding, "A real-time and fully distributed approach to motion planning for multirobot systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 12, pp. 2636–2650, 2019.

[8] D. Sun, J. Chen, S. Mitra, and C. Fan, "Multi-agent motion planning from signal temporal logic specifications," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3451–3458, 2022.

[9] D. Gundana and H. Kress-Gazit, "Event-based signal temporal logic synthesis for single and multi-robot tasks," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3687–3694, 2021.

[10] S. S. Farahani, V. Raman, and R. M. Murray, "Robust model predictive control for signal temporal logic synthesis," *IFAC-PapersOnLine*, vol. 48, no. 27, pp. 323–328, 2015.

[11] S. Sadraddini and C. Belta, "Robust temporal logic model predictive control," in *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2015, pp. 772–779.

[12] Y. E. Sahin, P. Nilsson, and N. Ozay, "Multirobot coordination with counting temporal logics," *IEEE Transactions on Robotics*, vol. 36, no. 4, pp. 1189–1206, 2019.

[13] J. Xue, X. Kong, B. Dong, and M. Xu, "Multi-agent path planning based on mpc and ddpg," *arXiv preprint arXiv:2102.13283*, 2021.

[14] S. H. Semnani, H. Liu, M. Everett, A. De Ruiter, and J. P. How, "Multi-agent motion planning for dense and dynamic environments via deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3221–3226, 2020.

[15] A. Nikou and D. V. Dimarogonas, "Decentralized tube-based model predictive control of uncertain nonlinear multiagent systems," *International Journal of Robust and Nonlinear Control*, vol. 29, no. 10, pp. 2799–2818, 2019.

[16] C. E. Luis, M. Vukosavljev, and A. P. Schoellig, "Online trajectory generation with distributed model predictive control for multi-robot motion planning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 604–611, 2020.

[17] J. Park and H. J. Kim, "Online trajectory planning for multiple quadrotors in dynamic environments using relative safe flight corridor," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 659–666, 2020.

[18] J. Park, D. Kim, G. C. Kim, D. Oh, and H. J. Kim, "Online distributed trajectory planning for quadrotor swarm with feasibility guarantee using linear safe corridor," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4869–4876, 2022.

[19] A. Gräfe, J. Eickhoff, and S. Trimpe, "Event-triggered and distributed model predictive control for guaranteed collision avoidance in uav swarms," *IFAC-PapersOnLine*, vol. 55, no. 13, pp. 79–84, 2022.

[20] F. Palafox, Y. Yu, and D. Fridovich-Keil, "Learning hyperplanes for multi-agent collision avoidance in space," 2023. [Online]. Available: https://arxiv.org/abs/2311.09439

[21] G. F. L. D'Alfonso and G. Fedele, "Distributed model predictive control for constrained multi-agent systems: a swarm aggregation approach," in *2018 Annual American control conference (ACC)*. IEEE, 2018, pp. 5082–5087.

[22] B. Lindqvist, P. Sopasakis, and G. Nikolakopoulos, "A scalable distributed collision avoidance scheme for multi-agent uav systems," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 9212–9218.