Resource-Aware Asynchronous Multi-Agent Coordination via Self-Triggered MPC

Yingzhao Lian[†], Stefan Wildhagen[†], Yuning Jiang[†,*], Boris Houska, Frank Allgöwer, Colin N. Jones

***Abstract*— In many applications, resource-aware devices are connected through a network, such as in the Internet of Things and energy hubs. These devices require proper coordination to achieve a high performance without violation of their resource limits. In this paper, we propose an asynchronous resource-aware multi-agent model predictive control to cooperatively coordinate agents to conduct a common task, whose resource concern is handled by a self-triggered control scheme. The consistency and recursive feasibility of the proposed MPC scheme are investigated. A reliable numerical implementation is introduced to deal with non-constant sampling times among agents, which is subsequently validated by a numerical example.**

I. Introduction

Resource-aware devices are ubiquitous in domains such as the Internet of Things (**IoT**) and wireless sensing systems. The actuation/triggering of these devices is limited by some resource factors, such as battery life or hardware longevity. Event-triggered [13] and self-triggered control [16] are two main paradigms to accommodate this issue. Event-triggered control operates *reactively*, i.e., it determines the triggering instants by monitoring a triggering condition. As a result, sensors are required to update at regular frequencies. Instead, self-triggered control operates *proactively*, i.e., it plans the next trigger time already at the previous trigger time. It enables both sensors and actuators to update at non-equidistant sampling instants. Self-triggered control is, therefore, preferable for resource-aware devices.

Trigger time selection is the key component in self-triggered controller design. The trigger time can be chosen as long as possible as in [2], [3]. In [9], [24], trigger time is optimized in a Model Predictive Control (**MPC**) scheme. A mixed-integer problem is solved in [9] in order to balance control performance and resource consumption. In [24], a non-convex reformulation [24] is proposed in which performance is optimized while resources are handled as constraints.

In practice, resource-aware devices are widely connected by a network, over which devices are coordinated cooperatively, such as energy hubs [7] or the IoT [14]. However, this coordination has never been investigated properly regarding the resource limit. This work focuses on optimizing overall performance by optimally deploying the resources of each device. The optimal coordination problem was investigated under the framework of distributed MPC [5] in the literature, where agents (i.e., resource-aware devices in this work) have been manipulated either non-cooperatively [19], [22] or cooperatively [8], [15], [12]. In this work, we consider the case where full information of each agent is accessible to all other agents, such that each agent is self-triggered asynchronously and optimizes their control inputs and next trigger time regarding the states and resources of all other agents. The proposed framework can be applied to a wide category of tasks such as network traffic coordination constraints by limited network resources [21] and energy-hub coordination constraints by energy source physics [17].

The main contribution of this paper is threefold:

- We propose an asynchronous MPC scheme for multi-resource-aware agent coordination.
- We investigate sufficient conditions for recursive feasibility of the proposed MPC scheme.
- We elaborate on reliable numerical implementation details of the proposed MPC to address convergence and scalability issues.

The remainder of this paper is organized as follows: In Section II, we introduce the dynamics of the system and the coordination problem of interest. In Section III, the receding horizon scheme of the proposed MPC is detailed and recursive feasibility is investigated. In Section IV, implementation details and its distributed form are elaborated. The proposed MPC is then validated in Section V.

Notation: In this paper, we use notation $\{a_i\}_{i=1}^{N_a}$ to denote a finite set of N_a elements indexed by i, $f(x(t))|_{t=T}$ denotes a function $f(x(t))$ evaluated at time $t=T$, and $\mathbb{Z}_a^b$ denotes $[a,b]\cap\mathbb{Z}$.

II. System Model and Problem Statement

This section presents a tracking coordination problem of multiple linearly coupled agents.

A. Self-Triggered Agent Model

This paper considers a multi-agent system with M local resource-aware self-triggered agents, whose control laws are fixed within two sequential triggering times $t_{i,k}$ and $t_{i,k+1}$. In particular, the dynamics of the i-th agent is governed by

$$\forall\, t\in[0,\infty)\ ,\quad \dot{x}_i(t)=f_i(x_i(t),u_i(t)), \tag{1}$$

where $x_i:\mathbb{R}\to\mathbb{R}^{n_x}$ and $u_i:\mathbb{R}\to\mathbb{R}^{n_u}$ denote the state and control inputs in continuous time. Moreover, f_i is assumed

[†]The first three authors contributed equally.
[*]Corresponding author.
Y. Lian and C. N. Jones are with the Automatic Control Laboratory, Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland {yingzhao.lian, colin.jones}@epfl.ch
S. Wildhagen and F. Allgöwer are with the Institute for Systems Theory and Automatic Control, University of Stuttgart, Germany {stefan.wildhagen}@ist.uni-stuttgart.de
Y. Jiang and B. Houska are with the School of Information Science and Technology, ShanghaiTech University, Shanghai, China {jiangyn, borish}@shanghaitech.edu.cn

to be differentiable. In direct optimal control [4], a self-triggered input over a time horizon $\overline{T}_i$ is parametrized by

$$u_i(t) = \sum_{k=0}^{N-1} v_{i,k}\xi_k(t, \Delta_{i,k}) \ , \tag{2}$$

where the orthogonal functions $\xi_k \in \mathbb{L}^2[0, \overline{T}_i]$, $k \in \mathbb{Z}_0^{N-1}$, encapsulate the triggering property and are given by parametric piece-wise constant functions

$$\xi_k(t, \Delta_{i,k}) = \begin{cases} 1 & t \in [t_{i,k}, t_{i,k+1}) \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

with $t_{i,0} = 0$, $t_{i,N} = \overline{T}_i$ and

$$\Delta_{i,k} = t_{i,k+1} - t_{i,k} \ , \ \ k \in \mathbb{Z}_0^{N-1} \ .$$

For the sake of simplicity, we use the notation

$$u_i(t) := u_i(t, v_i, \Delta_i)$$

to denote parametrization (2) with parameters

$$v_i := [v_{i,0}^\top, v_{i,1}^\top, ..., v_{i,N-1}^\top]^\top$$
$$\text{and} \quad \Delta_i := [\Delta_{i,0}, \Delta_{i,1}, \ldots, \Delta_{i,N-1}]^\top \ .$$

With initial state $\hat{x}_i$, the state trajectory evolving under dynamics (1) is denoted as

$$x(t) = \zeta_i(t, v_i, \Delta_i, \hat{x}_i) \in \mathbb{R}^{n_x}, \ \forall \ i \in \mathbb{Z}_1^M, \tag{4}$$

such that the trajectory $\zeta_i(\cdot, v_i, \Delta_i, \hat{x}_i)$ is parametrized by its intervals Δ_i, its control inputs parameters v_i and the initial state $\hat{x}_i$. In practice, numerical integration can be used to compute ζ_i approximately. Moreover, the control constraint and state constriants are given by

$$v_{i,k} \in \mathbb{U}_i \ , \ k \in \mathbb{Z}_0^{N-1} \quad \text{and} \quad \zeta_i(t, v_i, \Delta_i, \hat{x}_i) \in \mathbb{X}_i \ .$$

A self-triggered agent updates its control law by triggering, where the trigger time points are limited by a local resource. When the control input is fixed, the i-th agent recharges its local resource r_i at a constant rate ρ_i until saturation at $\overline{r}_i$. More specifically,

$$\begin{cases} \forall \ t \in (t_{i,k}, t_{i,k+1}) \ , \ \ \forall \ k \in \mathbb{Z}_0^{N-1} \ , \\ \dot{r}_i(t) = \sigma(\overline{r}_i - r_i(t))\rho_i \ , \end{cases}$$

where σ denotes a step function with $\sigma(s) = 1, \ s \geq 0$ and 0 elsewhere. When the control input is updated (i.e., the agent is triggered), the local resource discharges by an amount of μ_i to pay the update cost. Hence, the resource at the triggering times $t_{i,k}$ for all $k \in \mathbb{Z}_1^{N-1}$ is

$$r_i(t_{i,k}) = \begin{cases} \hat{r}_i, \ k = 0 \\ \lim\limits_{t \to t_{i,k}^-} r_i(t) - \mu_i(\Delta_{i,k-1}) \quad k \in \mathbb{Z}_1^N \ , \end{cases}$$

with initial value $\hat{r}_i$. Here, $t \to t_{i,k}^-$ denotes the limit from left[1] and μ_i denotes the update cost depending on the chosen interval. Moreover, the resource r_i is required to be bounded within the interval

$$r_i(t) \in [\underline{r}_i, \overline{r}_i] \ .$$

In general, a resource-aware self-triggered agent can update its control input when its resource can be charged without violating the the lower bound $\underline{r}_i$. Otherwise, it waits until enough resources are available.

B. Problem Formulation of Multi-Agent Coordination

M self-triggered agents are coordinated to track a common reference signal z^{ref}, which, in practice, might be the demand of a product or an aggregation signal in a power grid. However, the triggering times of the M self-triggered agents are asynchronous. At a given triggering time, only the triggered agents can update their control law while the non-triggered agents should maintain their control input until they are triggered themselves. The corresponding multi-agent coordination problem is defined as

$$\min_{v,\Delta} \ \sum_{i=1}^{M} \int_0^{\overline{T}_i} L_i(\zeta_i(t, v_i, \Delta_i, \hat{x}_i), u_i(t, v_i, \Delta_i))dt \tag{5a}$$

$$+ \int_0^{\overline{T}_0} \left\| z(t) - z^{\text{ref}}(t) \right\|_2^2 dt \tag{5b}$$

$$\text{s.t.} \ \ \forall \ t \in [0, \overline{T}_0] \ , \ \ z(t) = \sum_{i=1}^{M} A_i \zeta_i(t, v_i, \Delta_i, \hat{x}_i) \ , \tag{5c}$$

$$\forall i \in \mathbb{Z}_i^M \begin{cases} \zeta_i(t, v_i, \Delta_i, \hat{x}_i) \in \mathbb{X}_i, \ \forall t \in [0, \overline{T}_i] \\ v_{i,k} \in \mathbb{U}_i, \ \forall \ k \in \mathbb{Z}_0^{N-1} \ , \end{cases} \tag{5d}$$

$$\forall \ i \in \mathbb{Z}_i^M \ , \begin{cases} \forall \ k \in \mathbb{Z}_0^{N-1} \ , \ \forall \ t \in (t_{i,k}, t_{i,k+1}) \\ \dot{r}_i(t) = \sigma(\overline{r}_i - r_i(t))\rho_i, r_i(0) = \hat{r}_i \end{cases} \tag{5e}$$

$$\forall \ i \in \mathbb{A}^c \ ,$$
$$\begin{cases} r_{i,1}(t_{i,1}) = \lim\limits_{t \to t_{i,1}^-} r_i(t) - \mu_i(\Delta_{i,0} + \hat{\Delta}_i^-) \ , \\ \Delta_{i,0} + \hat{\Delta}_i^- \in [\underline{\Delta}_i, \overline{\Delta}_i] \ , \\ r_i(t_{i,k}) \in [\underline{r}_i, \overline{r}_i] \ , \ \ k \in \mathbb{Z}_1^N \\ \Delta_{i,k-1} \in [\underline{\Delta}_i, \overline{\Delta}_i] \ , \ \ k \in \mathbb{Z}_2^N \\ \Delta_{i,0} = \hat{\Delta}_i^+ \ , \ v_{i,0} = \hat{v}_i \end{cases} \tag{5f}$$

$$\forall \ i \in \mathbb{A} \ , \ \forall \ k \in \mathbb{Z}_1^N \ , \begin{cases} r_i(t_{i,k}) \in [\underline{r}_i, \overline{r}_i] \\ \Delta_{i,k-1} \in [\underline{\Delta}_i, \overline{\Delta}_i] \ , \end{cases} \tag{5g}$$

where the cooperative state $z \in \mathbb{R}^{n_z}$ couples M agents linearly in (5c) and matrices $A_i \in \mathbb{R}^{n_x \times n_z}$ model the effectiveness of each agent. The global cost (5b) penalizes the tracking error against the common reference signal $z^{\text{ref}}(t)$ over $[0, \overline{T}_0]$. Moreover, L_i denotes the local stage cost in (5a) over the local time horizon $[0, \overline{T}_i]$. $\overline{T}_0$ is fixed by the user and we require $\overline{T}_0 \leq \overline{T}_i, \ \forall i \in \mathbb{Z}_1^M$. The choice of $\overline{T}_0$ will be discussed in Section III-B.

The triggering interval is bounded by $[\underline{\Delta}_i, \overline{\Delta}_i]$. Moreover, the asynchronous triggering scheme discussed at the begining

[1]Specifically, $t \to t_{i,k}^- := t \to t_{i,k}^-, \ t < t_{i,k}$

of this section is handled by (5f) and (5g), where agents are divided into two groups. One group includes agents that are triggered at the current time instant, dubbed $\mathbb{A}$, while another group includes the non-triggered ones, denoted by $\mathbb{A}^c = \mathbb{Z}_1^M \backslash \mathbb{A}$. Before their next trigger in $t_{i,1} = \hat{\Delta}_i^+$, the non-triggered agents in $\mathbb{A}^c$ will maintain their current control law with parameter $v_{i,0}$, which is determined at its last trigger $\hat{\Delta}^-$ ago, where $\hat{\Delta}^-$ denotes the time difference between the last triggering time and the current time instant. In closed loop, the first two constraints in (5f) ensure that the resource of the non-triggered agents is consistent from its last trigger up to its next trigger. This is required for recursive feasibility in Section III-B, together with initial conditions $\forall i \in \mathbb{A}^c$, $\Delta_{i,0} = \hat{\Delta}_i^+$, $v_{i,0} = \hat{v}_i$. The online update of $\hat{\Delta}^+$ and $\hat{v}_i$ will be elaborated on in the next section.

Remark 1 *As opposed to the global coordination error* (5b) *cost, the local cost L_i in general does not penalize the tracking error. If this was the case, the resulting problem would become a challenging bi-level optimization, which could be investigated as a multi-follower problem [25], [17].*

III. Self-Triggered Model Predictive Control

The receding horizon scheme of the proposed multi-agent coordination is detailed in this section. Additionally, a sufficient condition for recursive feasibility is investigated.

A. Algorithm

Algorithm 1 outlines an asynchronous self-triggered MPC algorithm for multi-agent coordination.

Algorithm 1 MPC Scheme

Initialization:

- Set the current time instant to $t_c = 0$ and set $\mathbb{A} = \mathbb{Z}_1^M$.
- Each agent sets $\hat{\Delta}_i^- = 0,\ \hat{\Delta}_i^+ = 0$.

Online:

1) At time t_c, each agent measures their current state $\hat{x}_i$ and resource level $\hat{r}_i$.
2) Solve Problem (5) and obtain solution (v_i^*, Δ_i^*) for all $i \in \mathbb{Z}_1^M$.
3) Set the time shift $\Delta^s = \min_i\ \Delta_{i,0}^*$.
4) Each agent applies $u_i(t, v_i^*, \Delta_i^*)$ within the interval $[t_c, t_c + \Delta^s]$ and shifts the horizon with $t_c \leftarrow t_c + \Delta^s$.
5) Set $\mathbb{A} = \emptyset$ and update $\mathbb{A}$ as follows:
   For $i = 1 : M$ **do**
   - If $\Delta_{i,0}^* = \Delta^s$, set $\mathbb{A} \leftarrow \mathbb{A} \cup \{i\}$ and $\hat{\Delta}_i^- = 0$.
   - Else, set $\hat{v}_i = v_{i,0}^*$ and
   $$\hat{\Delta}_i^- = \hat{\Delta}_i^- + \Delta^s\ ,\ \ \hat{\Delta}_i^+ = \Delta_{i,0}^* - \Delta^s\ .$$

   End
6) Set $\mathbb{A}^c = \mathbb{Z}_1^M \backslash \mathbb{A}$ and go to Step 1).

Step 1) and 2) of Algorithm 1 are the standard MPC scheme [18] solving (5) based on the current measurement of states and resource levels. Step 3) sets the time shifting as the minimal triggering time $\Delta_{i,0}^*$ over i, which yields an asychronous triggering pattern online. Then, each agent applies the optimized control input to the real process within $[t_c, t_c + \Delta^s]$ and shifts its horizon with Δ^s in Step 4). In Step 5) and 6), the sets of triggered/non-triggered agents $\mathbb{A}/\mathbb{A}^c$ and their trigger time registers $\hat{\Delta}_i^-$, $\hat{\Delta}_i^+$ are updated.

In Algorithm 1, the triggering time sequence in closed loop among M self-triggered agents are not synchronized, since only $|\mathbb{A}|$ agents triggered at each time t_c. The triggered agents at t_c only optimize the manipulable parts, including:

- Trajectories of the triggered agents from current time t_c up to their end time $\overline{T}_i$.
- Trajectories of the non-triggered agents from their next scheduled trigger $t_c + \hat{\Delta}_i^+$ to their end time $\overline{T}_i$.

Remark 2 *In this paper, we shift the horizon forward to the first of the next scheduled trigger times of all agents. However, other ways to shift the horizon are also conceivable. For example, one could shift the horizon to the last of the next scheduled trigger times, such that all the agents are triggered at least once. We conjecture that with the same assumptions as presented in this paper, recursive feasibility could be established for this case as well.*

B. Properties of the MPC scheme

This section discusses recursive feasibility of Problem (5) in Algorithm 1, First, it is assumed that there exists a feasible $\tilde{\Delta}_i$, for all $i \in \mathbb{Z}_1^M$, which does not drain the resource. In order to introduce this assumption, we denote sets

$$\mathbb{D}_i := \{\Delta | \rho_i \Delta - \mu_i(\Delta) \geq 0\}\ ,$$
$$\text{and} \quad \mathbb{C}_i := \mathbb{D}_i \cap [\underline{\Delta}_i, \overline{\Delta}_i]\ .$$

Assumptions *We assume that*

A1 *The set $\mathbb{C}_i$ is nonempty for all i.*

A2 *There exists a $\Delta_i^d \in \mathbb{C}_i$ such that $\overline{r}_i - \mu(\Delta_i^d) \geq \underline{r}_i$ for all $i \in \mathbb{Z}_1^M$ and $(N-1) \cdot \underline{\Delta}_i \geq \overline{T}_0$.*

A3 *The sets $\mathbb{X}_i$ for all $i \in \mathbb{Z}_1^M$ are control forward invariant, i.e., for any given $\hat{x}_i \in \mathbb{X}_i$ and $\Delta_i^d \in \mathbb{C}_i$ there, exists a $v_i^d \in \mathbb{U}_i$ such that*

$$\forall\ t \in [0, \overline{T}_i]\ ,\ \ \zeta_i(t, v_i^d, \Delta_i, \hat{x}_i) \in \mathbb{X}_i\ .$$

Theorem 1 *Let A1–A3 be satisfied. If the MPC optimization problem* (5) *is feasible at current time t_c, then under Algorithm 1, it is feasible at time $t_c + \Delta^s$ with Δ^s as well.*

Proof. Let us denote the optimal solution of (5) at time t_c as (v_i^*, Δ_i^*) for all $i \in \mathbb{Z}_1^M$. Then, we show the recursive feasibility of (5) by the following two steps.

1) For the non-triggered agents $i \in \mathbb{A}^c$ at time $t_c + \Delta^s$, we can construct a feasible solution for (5) at time $t_c + \Delta^s$ as $\tilde{v}_i = v_i^*$ and
$$\tilde{\Delta}_i = [\Delta_{i,0}^* - \Delta^s, \Delta_{i,1}^*, ..., \Delta_{i,N-1}^*]^\top\ . \qquad (6)$$
According to the construction of constraint (5f), we have that $\tilde{\Delta}_i$ is feasible for both (5e) and (5f) for all

$i \in \mathbb{A}^c$. The time horizon $\overline{T}_i$ at $t_c + \Delta^s$ of $\tilde{\Delta}_i$ is

$$\overline{T}_i = \sum_{k=0}^{N-1} \Delta^*_{i,k} - \Delta^s \geq \overline{T}_0 \ ,$$

where the last inequality holds because $\Delta^*_{i,0} > \Delta^s$ and A2, and thus fulfills the requirement on $\overline{T}_i$. Moreover, $\zeta_i(t, \tilde{v}_i, \tilde{\Delta}_i, \hat{x}_i)|_{t=t_c+\Delta^s}$ is consistent with the trajectory from t_c, i.e., with $\zeta_i(t, v^*_i, \Delta^*_i, \hat{x}_i)|_{t=t_c}$, thus (v^+_i, Δ^+_i) is feasible for constraint (5d).

2) For the triggered agents $i \in \mathbb{A}$ at time $t_c + \Delta^s$, we can construct $\tilde{v}_i = [v^{*T}_{i,1}, \ldots, v_{i,N-1}, v^d_i]$, and $\tilde{\Delta}_i = [\Delta^*_{i,1}, \Delta^*_{i,2}, ..., \Delta^*_{i,N-1}, \Delta^d_i]$ is a feasible solution for constraints (5e) and (5g), because $r(t_{i,N}) \geq r(t_{i,N-1})$ due to A1 and A2. The resulting time horizon at $\overline{T}_i$ at $t_c + \Delta^s$ is

$$\overline{T}_i = \sum_{k=1}^{N-1} \Delta^*_{i,k} + \Delta^d_i \overset{\text{Ass 2}}{\geq} \overline{T}_0 \ .$$

Moreover, based on A3, v^d_i ensures that $\zeta_i(t, \tilde{v}_i, \tilde{\Delta}_i, \hat{x}_i)|_{t=t_c+\Delta^s}$ is still feasible for constraint (5d) from $t_{i,N-1}$ to $t_{i,N}$. ■

Theorem 1 establishes recursive feasibility of Problem (5) in Algorithm 1, where the non-triggered agents can reuse their solution from the last trigger, while the triggered agents rely on forward invariance of the constraint set.

IV. Numerical Implementation Details

This section elaborates on how to implement Algorithm 1 numerically. Note that Problem (5) is non-trivial to solve because of the asynchronous update among agents.

A. Integration among Different Time Grids

In practice, the state trajectory $x_i(t)$ (i.e., $\zeta_i(t, v_i, \Delta_i, \hat{x}_i)$) is only represented by some finite evaluations at different time stamps, called the local time grid. Triggering times $\{t_{i,k}\}^N_{k=0}$ are used as local time grid in our implementation and hence $\{x(t_{i,k})\}^N_{k=0}$ are critical points used to represent the whole trajectory from $\hat{x}_i$ to $x(\overline{T}_i)$. Meanwhile, the numerical representation of the cooperative state $z(t)$ also requires a global time grid. As the local time grid is a decision variable in the optimization problem (5), the choice of the global time grid is critical. It is intuitive to combine all local time grids $\{t_{i,k}\}^N_{k=0}$ to a global time grid. However, since the local time grids are decision variables, the resulting optimization problem has poor numerical robustness [23], i.e., high sensitivity and low dual feasibility. Instead, we propose to resolve this issue by using a fixed global time grid and approximating the trajectories of each agent by a piece-wise polynomial. We denote the global time grid as $\{t_{0,k}\}^{N_c}_{k=0}$ with N_c denoting number of grid points.

The value of the dynamics of the i-th agent at its local time grid $x_i(t_{i,k})$ is evaluated by (4) through numerical integration. The trajectory of the ith agent is interpolated as

$$\tilde{\zeta}_i(t) = \sum_{n=1}^{N_p} \phi_{(l,n)}(t) b_n(t), \tag{7}$$

where $\phi_{(l,n)}$ is an lth order Lagrange polynomial and b_n is a bumper function. In particular, $\phi_{(l,n)}$ is the Lagrange polynomial interpolating $l+1$ evaluations of the system states x on local grid points from $t_{i,l(n-1)}$ to $t_{i,ln}$, i.e., of $\{x_i(t_{i,k})\}^{ln}_{k=l(n-1)}$. The function b_n is defined as

$$b_n(t) = \begin{cases} 1, & t \in (t_{i,l(n-1)}, t_{i,ln}) \\ 0.5, & t \in \{t_{i,l(n-1)}, t^{ln}_i\} \\ 0, & \text{else} \end{cases} \ . \tag{8}$$

Notice that b_n and ξ_i in (3) are different and they cannot be interchanged with each other. This interpolated trajectory is of at most l^{th} order globally, and hence has a lower sensitivity in the optimization than one single N^{th} order Lagrange polynomial. In general, function b_n selects a region within which a low order polynomial is applied, meanwhile, its value at the end points $\{t_{i,l(n-1)}, t_{i,ln}\}$ ensures continuity at the sampling time.

With the help of the trajectory interpolation, the coordination error (5b) can be integrated by evaluation of $z(t)$ and $z^{\text{ref}}(t)$ at the global time grid $\{t_{0,k}\}^{N_c}_{k=0}$. For example,

$$z(t_{0,k}) = \sum_{i=1}^{M} A_i \tilde{\zeta}(t_{0,k}) \ .$$

B. Problem Discretization

The discontinuity in the resource dynamics is a main concern in solving the MPC Problem (5). It can be discretized as discussed in [24] in the form

$$r_i(t_{i,k+1}) = r_i(t_{i,k}) + \rho_i \Delta_{i,k} - \mu_i(\Delta_{i,k})$$

such that the evolution can be summarized as

$$r_i(t_{i,k}) = \hat{r}_i + \sum_{m=1}^{k-1} (\rho_i \Delta_{i,m} - \mu_i(\Delta_{i,m}))$$

for all $k \in \mathbb{Z}^{N-1}_0$. Then, we write the local constraint sets

$$\Lambda_i = \left\{ \begin{array}{l|l} \Delta_i \in \mathbb{R}^N & \forall\, k \in \mathbb{Z}^N_0 \begin{Bmatrix} x_{i,k} = \tilde{\zeta}_i(t_{i,k}) \\ x_{i,k} \in \mathbb{X}_i \end{Bmatrix} \\ v_i \in \mathbb{R}^{Nn_u} & \begin{Bmatrix} r_{i,1}(t_{i,1}) = \hat{r}_i + \rho \Delta_{i,0} \\ \qquad -\mu_i(\Delta_{i,0} + \hat{\Delta}^-_i) \ , \\ \Delta_{i,0} + \hat{\Delta}^-_i \in [\underline{\Delta}_i, \overline{\Delta}_i] \ , \\ \Delta_{i,0} = \hat{\Delta}^+_i \ , \ v_{i,0} = \hat{v}_i \ , \end{Bmatrix} \\ x_i \in \mathbb{R}^{(N+1)n_x} & \\ & \forall\, k \in \mathbb{Z}^N_1 \begin{Bmatrix} r_i(t_{i,k}) \in [\underline{r}_i, \overline{r}_i] \\ \Delta_{i,k} \in [\underline{\Delta}_i, \overline{\Delta}_i] \\ v_{i,k} \in \mathbb{U}_i \end{Bmatrix} \end{array} \right\}$$

for all $i \in \mathbb{A}^c$ and

$$\Lambda_i = \left\{ \begin{array}{l|l} \Delta_i \in \mathbb{R}^N & \forall\, k \in \mathbb{Z}^N_0 \begin{Bmatrix} x_{i,k} = \tilde{\zeta}_i(t_{i,k}) \\ x_{i,k} \in \mathbb{X}_i \end{Bmatrix} \\ v_i \in \mathbb{R}^{Nn_u} & \\ x_i \in \mathbb{R}^{(N+1)n_x} & \forall\, k \in \mathbb{Z}^N_0 \begin{Bmatrix} r_i(t_{i,k}) \in [\underline{r}_i, \overline{r}_i] \\ \Delta_{i,k} \in [\underline{\Delta}_i, \overline{\Delta}_i] \\ v_{i,k} \in \mathbb{U}_i \end{Bmatrix} \end{array} \right\}$$

for all $i \in \mathbb{A}$. Based on the discussion above, Problem (5) can be then formulated in discrete-time as

$$\min_{\Delta,s,v,z} \quad \sum_{k=0}^{N_c} \|z_k - z_k^{\text{ref}}\|_2^2 + \sum_{i=1}^{M} \ell_i(v_i, \Delta_i) \tag{9a}$$

$$\text{s.t.} \quad \forall\, k \in \mathbb{Z}_0^{N_c}\ ,\ \ z_k = \sum_{i=1}^{M} A_i \tilde{\zeta}(t_{0,k})\ , \tag{9b}$$

$$\forall\ i \in \mathbb{Z}_1^{M}\ ,\ \ (s_i, v_i, \Delta_i) \in \Lambda_i\ . \tag{9c}$$

Problem (9) has a particular distributed structure, where the local decision variables (x_i, v_i, Δ_i) are only coupled via the affine equality constraints (9b) over x_i while having fully decoupled inequality constraints (9c). Here, the decoupled cost of each agent is given by

$$\ell_i(v_i, \Delta_i) = \sum_{k=0}^{N-1} L_i(\zeta(t_{i,k}, v_i, \Delta_i, \hat{x}_i), u_i(t_{i,k}, v_i, \Delta_i))\ .$$

Remark 3 (Online Solver) *Solving* (9) *in a limited sampling time requires an efficient and fast online solver. In practice, as each agent prefers to preserve their local privacy while Problem* (9) *is a structured optimization problem as discussed above, distributed optimization has its potential to exploit this structure and solve* (9) *via neighbor-to-neighbor communication, which is always required in a large scale multi-agent systems. For instance, a recently proposed algorithm, called augmented Lagrangian based alternating direction inexact Newton (ALADIN) method [10] was developed for solving optimization problems in the form of* (9)*. ALADIN has already been applied to solve coordination problems arising in different multi-agent systems such as smart building coordination [20], traffic coordination [11] and so on. Future work will develop this novel method for solving* (9) *as an online solver.*

C. Tricks for Initialization

We observed that Problem (5) is highly sensitive to initialization. A proper initialization trick is therefore introduced. One can imagine that the coordination problem actually assigns sub-tracking references $z_i^{\text{ref}}(t)$ to each agent, even though finding $z_i^{\text{ref}}(t)$ is at least as difficult as solving the original MPC problem (5). This viewpoint still allows a feasible initialization of the agents' trajectories $\{x_i(t_{i,k}), r_i(t_{i,k})\}_{k=0}^{N-1}$ and local decision variables v_i, Δ_i. One can give an initial guess of $z_i^{\text{ref}}(t)$, such that

$$z^{\text{ref}}(t) = \sum_{i=1}^{M} z_i^{\text{ref}}(t), \forall t \in [0, \overline{T}].$$

Without loss of generality, we assume that if the i-th agent is triggered at t_c, its initialization is the optimal solution to the following optimization problem:

$$\begin{aligned} \min_{v_i,\Delta_i,s_i} \quad & \ell_i(v_i, \Delta_i) + \sum_{k=0}^{N_c} \|A_i x_{i,k} - z_i^{\text{ref}}(t_{0,k})\|_2^2 \\ \text{s.t.} \quad & (s_i, v_i, \Delta_i) \in \Lambda_i\ . \end{aligned} \tag{10}$$

After initialization of the local agents, the coordination error can be calculated accordingly.

D. Discussion

One might suggest enforcing end time synchronization, which in turn means that $\overline{T}_0 = \overline{T}_i,\ \forall i, j \in \{1, \ldots, M\}$. In theory, this formulation enables the end time of coordination error $\overline{T}_0$ to be $\overline{T}_i$, which should result in better closed-loop performance. However, this formulation leads to significantly lower numerical stability because the extra constraints propagate sensitivity through the whole trajectory. This is the same reason why we interpolate the agents' trajectories with piecewise polynomials with local control instead of one single polynomial with global control [6].

V. Numerical Example

The following simulation is implemented by CASADI [1] with IPOPT [23]. We consider coordination of two double integrators, whose dynamics are

$$\dot{x}(t) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} v(t),$$

with input constraints $v(t) \in [-2, 2]$. The resource dynamics for both agents have parameters $\rho = 1,\ \underline{r} = 0$ and $\overline{r} = 1$. In this numerical example, we consider that the first agent has less effective contribution to the output but a lower update cost, while the second agent has more effective contribution to the output but pays more update cost. In particular, for the first agent, $A_1 = [1,\ 0]$ and $\mu_1(\Delta) = 0.3$ with output constraints $A_1 x_1 \in [-0.5, 0.5]$, while for the second agent, $A_2 = [2,\ 0],\ \mu_2(\Delta) = 0.5$ with $A_2 x_2 \in [-1, 1]$. The end time for the coordination error $\overline{T}$ is set to $3.2\,\text{s}$ while the local cost functions are set to $L_i(t) = u_i(t)^T u_i(t)$.

The tracking of a step-like signal is shown in Figure 1. The corresponding resource levels for each agent are shown in Figure 2 (top), and the inter-trigger times are depicted in Figure 2 (bottom). It is observed that the resources are mainly consumed when the reference signal changes and the system tends to use the second agent to track a fast tracking signal while uses the first agent to stabilize the output, More specifically, in Figure 2 (bottom), when the reference signal changes at $0\,\text{s}$, $5\,\text{s}$ and $12\,\text{s}$, the inter-trigger time of the second agent drops significantly and ramps out last. However, when the system output almost converges, Agent 1 has more frequent triggers (for example from $6\,\text{s}$-$10\,\text{s}$ and $1\,\text{s}$-$5\,\text{s}$ in Figure 2 (bottom)). This observation is aligned with our intuition that the fast agent is triggered for fast change while slower agent is triggered for slow convergence.

Even though recursive feasibility was analyzed in Section III, stability of the closed loop is not clear yet, especially for the local agents. The output trajectories of the local agents $A_i x_i(t)$ in the aforementioned example are shown in Figure 1, in which both systems converge when the common output z converges at first and the third step signal, while in the second step signal, the local agents' trajectories have not yet converged. To ensure stability of the local agents, a tracking problem has to be enforced in the local cost $L_i(t)$,

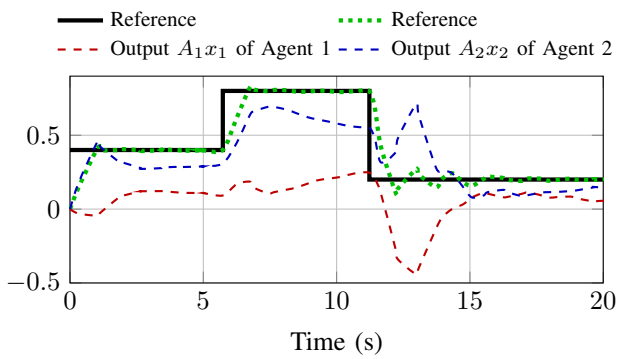

Fig. 1. Tracking of step-like reference signal.

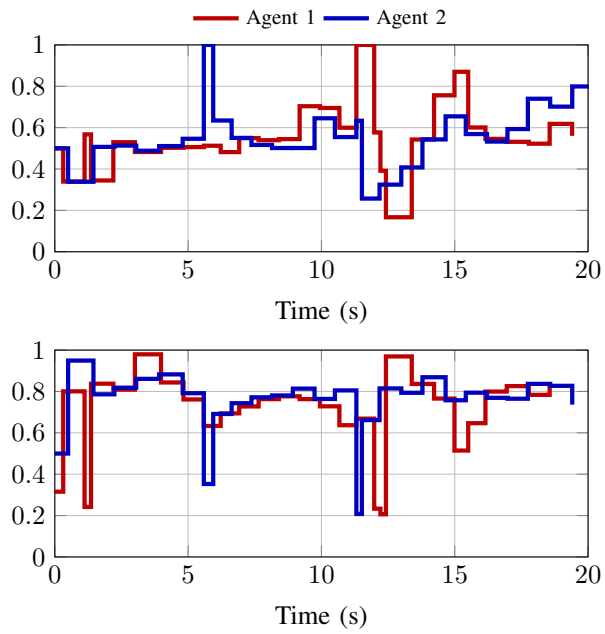

Fig. 2. Resource (top) and inter-trigger times (bottom) of local agents.

which is another layer of bi-level optimization as discussed in Remark 1.

VI. Conclusions

In this paper, an asynchronous resource-aware multi-agent self-triggered MPC is proposed to optimally coordinate all agents to conduct a common task cooperatively regarding their system dynamics limited by their resource limits. Theoretical questions with respect to the consistency and recursive feasibility have been answered. A reliable numerical implementation has been detailed and validated by a toy example. Guarantees for stability and a larger scale real-world application will be covered in the future works.

Acknowledgements

The authors would like to thank the German Research Foundation (DFG) for financial support within the German Excellence Strategy under grant EXC-2075, the Swiss National Science Foundation under the RISK project (grant number 200021 175627), and ShanghaiTech University, Grant-Nr. F0203- 14-012.

References

[1] J. A. E. Andersson, J. Gillis, G. H., J. B. Rawlings, and M. Diehl. CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36, 2019.

[2] J.D.J. Barradas Berglind, T.M.P. Gommans, and W.P.M.H. Heemels. Self-triggered MPC for constrained linear systems and quadratic costs. *IFAC Proceedings Volumes*, 45(17):342–348, 2012.

[3] Daniele Bernardini and Alberto Bemporad. Energy-aware robust model predictive control based on noisy wireless sensors. *Automatica*, 48(1):36–44, 2012.

[4] H.G. Bock and K.J. Plitt. A multiple shooting algorithm for direct solution of optimal control problems. In *Proceedings 9th IFAC World Congress Budapest*, volume 6, pages 4555–4559, 1984.

[5] E. Camponogara, D. Jia, B. H. Krogh, and S. Talukdar. Distributed model predictive control. *IEEE Control Systems Magazine*, 22(1):44–52, 2002.

[6] J. Gallier. *Curves and surfaces in geometric modeling: theory and algorithms*. Morgan Kaufmann, 2000.

[7] M. Geidl and G. Andersson. Optimal power flow of multiple energy carriers. *IEEE Trans. on Power Systems*, 22(1):145–155, 2007.

[8] P. Giselsson and A. Rantzer. On feasibility, stability and performance in distributed model predictive control. *IEEE Transactions on Automatic Control*, 59(4):1031–1036, 2013.

[9] E. Henriksson, D. E. Quevedo, H. Sandberg, and K. H. Johansson. Self-triggered model predictive control for network scheduling and control. *IFAC Proc. Volumes*, 45(15):432–438, 2012.

[10] B. Houska, J. Frasch, and M. Diehl. An augmented lagrangian based algorithm for distributed non-convex optimization. *SIAM Journal on Optimization*, 26(2):1101–1127, 2016.

[11] Y. Jiang, M. Zanon, R. Hult, and B. Houska. Distributed algorithm for optimal vehicle coordination at traffic intersections. In *Proc. 20th IFAC World Congress*, page 12082–12087, 2017.

[12] T. Keviczky, F. Borrelli, and G. J. Balas. A study on decentralized receding horizon control for decoupled systems. In *Proc. American Control Conf.*, pages 4921–4926, 2004.

[13] H Li and Y Shi. Event-triggered robust model predictive control of continuous-time nonlinear systems. *Automatica*, 50(5):1507–1513, 2014.

[14] S. Madakam and V. Lake. Internet of things (IoT): A literature review. *Journal of Computer and Communications*, 3(05):164, 2015.

[15] J.M. Maestre, D. M. De La Peña, E.F. Camacho, and T. Alamo. Distributed model predictive control based on agent negotiation. *Journal of Process Control*, 21(5):685–697, 2011.

[16] M. Mazo and P. Tabuada. On event-triggered and self-triggered control over sensor/actuator networks. In *2008 47th IEEE Conference on Decision and Control*, pages 435–440. IEEE, 2008.

[17] A. Mirzapour-Kamanaj, M. Majidi, K. Zare, and R. Kazemzadeh. Optimal strategic coordination of distribution networks and interconnected energy hubs: A linear multi-follower bi-level optimization model. *International Journal of Electrical Power & Energy Systems*, 119:105925, 2020.

[18] J. B. Rawlings, D. Q. Mayne, and M. Diehl. *Model Predictive Control: Theory, Computation, and Design*. Nob Hill Publishing, 2017.

[19] A. Richards and J. P. How. Robust distributed model predictive control. *Int. Journal of Control*, 80(9):1517–1531, 2007.

[20] J. Su, Y. Jiang, A. Bitlislioğlu, C. N. Jones, and B. Houska. Distributed multi-building coordination for demand response. In *Proc. 21st IFAC World Congress*, 2020.

[21] A. S. Tanenbaum and D. J. Wetherall. Computer networks fifth edition. In *Pearson Education, Inc.* Prentice Hall, 2011.

[22] P. A. Trodden and J. M. Maestre. Distributed predictive control with minimization of mutual disturbances. *Automatica*, 77:31–43, 2017.

[23] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006.

[24] S. Wildhagen, C. N. Jones, and F. Allgöwer. A resource-aware approach to self-triggered model predictive control. In *Proc. 21st IFAC World Congress*, 2020.

[25] J. J. Ye. Optimal strategies for bilevel dynamic problems. *SIAM Journal on Control and Optimization*, 35(2):512–531, 1997.