IEEE Control Systems Letters paper presented at
2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Multi-Agent Dynamic Scheduling with A Posteriori Path Tracking and Collision Avoidance Using Model Predictive Control *

Jeremy Bertoncini, Viktoriya Nikitina and Matthias Gerdts [1]

***Abstract*— This research work investigates a coordinated multi-agent path planning and tracking method. The solution of a pre-processed dynamic scheduling problem performs target assignment and provides optimal starting times and paths for each agent. Afterwards, a linear model predictive controller ensures robust and fast path tracking while preventing agents from collisions. This task is formulated as a discretized quadratic programming (QP) problem and is solved using an in-house developed semi-smooth Newton method. Numerical experiments have demonstrated the efficiency of the approach.**

I. INTRODUCTION

In the past decades, technological advances have considerably contributed to a further improvement of autonomous aerial vehicles (AAVs). Having a great potential to become a game changer, they have been extensively integrated into different areas of life such as urban mobility, logistics, agriculture etc. The ongoing upward trend will continue in the long term despite pandemics, political instabilities or other crises [1]. Thus, *efficient* operation and *safe* interaction of autonomous agents are major research challenges to address.

To the best of authors' knowledge, not many *deterministic* approaches have been proposed to couple dynamic scheduling, optimal path planning and MPC-based path tracking with collision avoidance. [2], [3] investigate scheduling problems using mixed-integer programming and present a multi-agent path planning method under collision avoidance. [4] provides interesting insights and results for dynamic scheduling and optimal control of autonomous vertical take-off and landing vehicles.

Collision avoidance is a strong requirement of robust path planning and tracking methods. Optimal control problems (OCP) dealt with are very often nonlinear and contain a large number of constraints. Generation of collision-free optimal trajectories may be performed by different approaches. Nonlinear nonconvex path planning methods demonstrated their real-time efficiency in complex collision avoidance scenarios in simulations, see [5], [6], [7], [8], as well as in experiments using a quadrocopter equipped with a monocular camera [9]. Nonetheless, the use of linearized models and linear model predictive control (LMPC) is advantageous in terms of robustness and real-time capability, see [10], [11]. Moreover, as centralized optimal control approaches suffer from their size [12], separate consideration of multiple agents and their coordination is a suitable approach, especially when dealing with an increasing number of agents.

The main contribution of this work is the development of a fast and efficient numerical method for optimal path planning and tracking of AAVs. Combining dynamic scheduling with coordinated LMPC, our approach benefits from the advantages of these concepts and mitigates their downsides.

II. OVERALL PROBLEM DESCRIPTION AND COUPLING STRUCTURE

Given a fixed number of agents, static targets and moving obstacles with known dynamics, the goal is to compute an optimal schedule, to assign each agent to a target, to provide paths with the shortest length and to track these trajectories by avoiding collisions with obstacles and other agents.

Our approach mainly consists of two blocks, see Figure 1. The first part includes dynamic multi-agent scheduling and optimal path planning. The solution of this bilevel mixed-integer problem (MIP) comprises an optimal schedule and optimal paths and is used as the input data for the second part of the method. It is based on an efficient LMPC algorithm and prevents agents from possible collisions with each other.

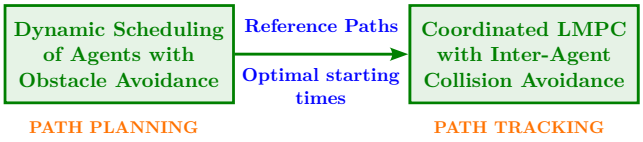

Fig. 1. Connection between dynamic scheduling (path planning) and coordinated LMPC (path tracking)

Throughout this article, the number of agents, static targets and dynamic obstacles are denoted by $N \in \mathbb{N}$, $T \in \mathbb{N}$ and $Q \in \mathbb{N}_{\geq 0}$, respectively, with the corresponding index sets $\mathcal{N} := \{1, \dots, N\}$, $\mathcal{T} := \{1, \dots, T\}$ and $\mathcal{Q} := \{0, \dots, Q\}$. The starting positions of all agents are denoted by $\overline{x}_i$, $i \in \mathcal{N}$, and are supposed to be fixed and given. The whole space domain is denoted by $\Omega \subset \mathbb{R}^d$, $d \in \{2, 3\}$. Moreover, static regions and dynamic obstacles are denoted by $\Omega_{T,k}$, $k \in \mathcal{T}$, and $\Omega_{Q,l}(t)$, $l \in \mathcal{Q}$, respectively. After having assigned an agent to a target, all other targets are treated by the algorithm as obstacles. The time-dependent set $\Omega_S(t)$, across which any agent is allowed to move on its way to its target region, can be represented as

$$\Omega_S(t) = \Omega \setminus \left(\bigcup_{k \in \mathcal{T}} \Omega_{T,k} \cup \bigcup_{l \in \mathcal{Q}} \Omega_{Q,l}(t) \right).$$

* This research is funded by dtec.bw – Digitalization and Technology Research Center of the Bundeswehr. dtec.bw is funded by the European Union– NextGenerationEU.

[1] Jeremy Bertoncini, Viktoriya Nikitina and Matthias Gerdts are with the Institute of Applied Mathematics and Scientific Computing, Department of Aerospace Engineering, University of the Bundeswehr Munich, 85579 Neubiberg, Germany {jeremy.bertoncini, viktoriya.nikitina, matthias.gerdts}@unibw.de

Assumption II.1. *Agents are supposed to be autonomous aerial vehicles flying at their cruise velocity. Their flight velocities v_i are assumed to be constant (on average) and equal implying $v_i = v,\ \forall i \in \mathcal{N}$, with $v \in \mathbb{R}_{>0}$.*

This assumption is introduced due to numerical reasons. It allows us to reduce the computational time considerably by calculating only one value function for all agents flying to the same target.

Assumption II.2. *Within the framework of this article, the obstacles are supposed to move periodically.*

This simplification does not represent a limitation since the method can be easily extended towards other dynamical systems. More details can be found in [4].

III. PATH PLANNING PROBLEM STATEMENT

This section explains the path planning method within our coupled approach. This part involves dynamic scheduling, which assigns each agent to a target and provides optimal starting times as well as optimal paths with the shortest length by avoiding collisions with obstacles.

A. Bilevel Structure and Problem Description

The dynamic scheduling setting with a bilevel structure can be described by two different optimization problems at the upper and lower level.

Problem 1 (Upper Level: Scheduling of Agents).

$$\min_{t_i,\omega_{ij},\eta_{ik}} \sum_{i=1}^{N} \left(t_i + \sigma \sum_{k=1}^{T} D_i^k(t_i)\eta_{ik} \right) \tag{1a}$$

subject to

$$t_i + \frac{D_i^k(t_i)}{\beta} - t_j \leq M(1-\omega_{ij}) + M(1-\eta_{ik}) + M(1-\eta_{jk})$$
$$\text{for } i<j,\ i,j \in \mathcal{N},\ k \in \mathcal{T}, \tag{1b}$$

$$t_j + \frac{D_j^k(t_j)}{\beta} - t_i \leq M\omega_{ij} + M(1-\eta_{ik}) + M(1-\eta_{jk})$$
$$\text{for } i<j,\ i,j \in \mathcal{N},\ k \in \mathcal{T}, \tag{1c}$$

$$\sum_{k=1}^{T} \eta_{ik} = 1 \qquad \text{for } i \in \mathcal{N}, \tag{1d}$$

$$t_i \in [t_{i,min}, t_{i,max}] \qquad \text{for } i \in \mathcal{N}, \tag{1e}$$

$$\omega_{ij} \in \{0,1\} \qquad \text{for } i<j,\ i,j \in \mathcal{N}, \tag{1f}$$

$$\eta_{ik} \in \{0,1\} \qquad \text{for } i \in \mathcal{N},\ k \in \mathcal{T}. \tag{1g}$$

Herein, t_i corresponds to the starting time of agent i, whose flight duration to target k is denoted by $D_i^k(t_i),\ i \in \mathcal{N},\ k \in \mathcal{T}$. Furthermore, $\sigma > 0$ is a flight duration penalization term, $\beta > 0$ is a scheduling constant, and M is a sufficiently large constant to ensure that only one of constraints (1b) or (1c) is active at the same time. Binary variables ω_{ij} and $\eta_{ik},\ i,j \in \mathcal{N},\ k \in \mathcal{T}$, are defined in the following way:

$$\omega_{ij} = \begin{cases} 1 & \text{if agent } i \text{ starts before agent } j, \\ 0 & \text{otherwise} \end{cases}$$

and

$$\eta_{ik} = \begin{cases} 1 & \text{if agent } i \text{ flies to target } k, \\ 0 & \text{otherwise.} \end{cases}$$

Constraints (1b) and (1c) are introduced to guarantee that an agent flying to the same target as another one can start only after some predefined period of time. For $\beta = 1$, this period coincides with the whole mission time of the previous agent. Next, equations (1d) ensure that each agent flies to exactly one target. In addition, (1e) are required to define a lower and upper bound for $t_i,\ i \in \mathcal{N}$. Finally, constraints (1f) and (1g) correspond to the integrality conditions.

Note that in case that $N \geq T$, an additional constraint

$$\sum_{i=1}^{N} \eta_{ik} \geq 1 \quad \text{for } k \in \mathcal{T}$$

is introduced to guarantee that each target is reached by at least one agent.

Denoting the length of the shortest path of agent i to target k by $d_k(t_i)$ and taking Assumption II.1 into account, $D_i^k(t_i)$ can be expressed as

$$D_i^k(t_i) = d_k(t_i)/v, \quad i \in \mathcal{N},\ k \in \mathcal{T}.$$

Herein, $d_k(t_i)$ is a parametric function. It can be determined solving the following parametric OCP, whose formulation involves the arc length parameterization:

Problem 2 (Lower Level: Paths with Shortest Length).

$$d_k(t_i) := \min_{x_i,u_i} \int_{t_i}^{\infty} \chi_{\Omega_S(t)}(x_i(t))dt \tag{2a}$$

subject to

$$\dot{x_i}(t) = u_i(t) \quad \text{for almost every } t \in [t_i, \infty), \tag{2b}$$

$$x_i(t) \in \Omega_S(t) \cup \Omega_{T,k} \quad \text{for every } t \in [t_i, \infty), \tag{2c}$$

$$u_i(t) \in U_i(x_i) = \begin{cases} \{u \in \mathbb{R}^{n_u} \mid ||u|| = 1\},\ x_i \in \Omega_S(t) \\ \{0 \in \mathbb{R}^{n_u}\},\ x_i \in \Omega_{T,k} \end{cases}$$
$$\text{for almost every } t \in [t_i, \infty), \tag{2d}$$

$$x_i(t_i) = \overline{x}_i. \tag{2e}$$

Herein, $n_u = d \in \{2,3\}$, $\overline{x}_i$ corresponds to the initial position of agent i, and χ_A is the indicator function.

B. Solution Procedure

The solution procedure in the first part of our method can be summarized in the following way:

1) Solve lower level problems for every target $k \in \mathcal{T}$ determining the value function $V^k(t,x)$ with the help of the Hamilton-Jacobi-Bellman equation:
$$\sup_{u \in U(x),\ x \in \Omega_S(t)} \{-\nabla_x V^k(t,x)^\top u(t) - 1\} = 0. \tag{3}$$
2) Insert parameters t_i and given initial positions $\overline{x}_i$ into $V^k(t,x)$ and set $d_k(t_i) := V^k(t_i, \overline{x}_i)$.
3) Compute $D_i^k(t_i)$ as indicated above, insert these functions into (1a) and solve the upper level problem.

4) Compute optimal control variables according to the following equation derived from (3):

$$u_i^*(t_i, x_i) = \underset{u_i \in U_i(x_i),\ x_i \in \Omega_S(t)}{\arg\min} \{\nabla_x V^k(t_i, x_i)^\top u_i + 1)\}.$$

5) Obtain optimal trajectories by inserting $u_i^*(t_i, x_i)$ into the differential equation (2b) and solving it.

The numerical implementation of this concept involves dynamic programming [13], Kružkov transformation [14], discretization of both the state and control space and interpolation. The execution of these steps yields a single-level mixed-integer nonlinear problem (MINLP). Since most modern nonlinear solvers have proven to be not efficient enough, an elegant piecewise linearization technique extensively described in [4], [15] is applied. Its main advantage is the possibility to control the linearization error at any time point. The final discretized mixed-integer linear problem (MILP) is solved using a well-performing linear solver [16].

IV. PATH TRACKING PROBLEM STATEMENT

This section explains the LMPC-based path tracking concept of our coupled approach. Note that a detailed analysis is omitted here since it goes beyond the scope of this work.

A. Motion Along A Reference Curve

The goal is to develop a path tracking controller based on a purely kinematic 2D agent model following a planar reference curve $\gamma_r : [0, L] \to \mathbb{R}^2$ of length $L > 0$. Hereby, γ_r is parameterized by its arc length $s \in [0, L]$ such that $\gamma_r(s) := (x_r(s), y_r(s))^\top$. The nonlinear equations of motion of an agent moving along a reference curve γ_r with a constant flight velocity v can be expressed in terms of the *Frenet* coordinates [17] such that:

$$\begin{cases} \dot{s}(t) = \frac{v \cdot \cos(\psi(t) - \psi_r(t))}{1 - r(t) \cdot \kappa_r(s(t))}, \\ \dot{r}(t) = v \cdot \sin(\psi(t) - \psi_r(t)), \\ \dot{\psi}(t) = v \cdot \kappa(t), \\ \dot{\kappa}(t) = u(t), \\ \dot{\psi}_r(t) = \dot{s}(t) \cdot \kappa_r(s(t)). \end{cases} \tag{4}$$

Herein, t denotes the time, $r(t)$ corresponds to the lateral deviation of the agent position $(x(t), y(t))^\top$ from the reference curve γ_r. Moreover, $\psi(t)$ and $\kappa(t)$ are the agent's yaw angle and its curvature, respectively. Similarly, $\psi_r(t)$ and $\kappa_r(t)$ denote the reference curve's yaw angle and its curvature, respectively. Note that the curvature of the reference curve is given by $\kappa_r(s(t)) = \dot{x}_r(t) \cdot \ddot{y}_r(t) - \dot{y}_r(t) \cdot \ddot{x}_r(t)$ and its yaw angle by $\psi_r(s(t)) = \arctan\left(\frac{\dot{y}_r(s(t))}{\dot{x}_r(s(t))}\right)$.

An agent is steered by controlling the time derivative of its curvature implying $\dot{\kappa}(t) = u(t)$. An observable state vector $y(t) = [r(t), \psi(t) - \psi_r(t)]^\top$ is introduced in order to linearize the equations of motion (4). Indeed, for a given constant flight velocity v, the approximations

$$\dot{s} \approx v, \quad \dot{r}(t) \approx v(\psi(t) - \psi_r(t)) \tag{5}$$

are acceptable under small deviations, i.e. $y(t) \approx (0, 0)$. Finally, we obtain the following linear time-variant (LTV) system:

$$\begin{cases} \dot{X}(t) = A \cdot X(t) + B \cdot u(t) + d(t), \\ y(t) = C \cdot X(t) \end{cases} \tag{6}$$

with $X(t) = [r(t), \psi(t), \kappa(t), \psi_r(t)]^\top$,

$$A = \begin{bmatrix} 0 & v & 0 & -v \\ 0 & 0 & v & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \ B = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix}, \ d(t)^\top = \begin{bmatrix} 0 & 0 & 0 & v \cdot \kappa_r(t) \end{bmatrix}.$$

B. Inter-Agent Collision Avoidance

In the sequel, each curve uses a spline approximation. We denote by $(V)_j^i$ the projection of any vector $\vec{V}$ related to agent i projected on spline j. While working with multiple agents, inter-agent collisions may occur. For each agent, any other one is viewed as a possible obstacle. Obstacles are modeled by imposing linear state constraints on the lateral deviation $r(t_n)$ at a discretized time step t_n such that $\bar{r}(t_n) \geq r(t_n) \geq \underline{r}(t_n)$, where $\bar{r}(t_n)$ and $\underline{r}(t_n)$ are the maximal and minimal lateral deviation bounds, respectively. The inter-agent collision avoidance is linear in order to keep the multi-agent problem in the QP form. The bounds of $r(t_n)$ may be deformed using an elastic Gaussian mode as indicated in [11]:

$$\underline{r}_j^i(t_n) \leq r_j^i(t_n) + \sum_{\forall j \neq i} \underline{\delta}_j^i - \sum_{\forall j \neq i} \bar{\delta}_j^i \leq \bar{r}_j^i(t_n), \tag{7}$$

where $\underline{\delta}_j^i$ and $\bar{\delta}_j^i$ have the form:

$$\delta_j^i(t_n) = \begin{cases} R_j^i(t_n) \cdot \exp\left(\frac{-(s_i^i - \epsilon(t_n) - s_j^i)^{c_2}}{c_{11}}\right) & \text{if } s_i^i \leq \epsilon(t_n) - s_j^i, \\ R_j^i(t_n) \cdot \exp\left(\frac{-(s_i^i + \epsilon(t_n) - s_j^i)^{c_2}}{c_{12}}\right) & \text{if } s_i^i \geq s_j^i + \epsilon(t_n), \\ R_j^i(t_n) & \text{else.} \end{cases} \tag{8}$$

Herein, s_j^i indicates the arc length of agent i projected on spline j with $1 < i, j < N$. The exponent c_2 is a positive even integer whereas the coefficients c_{11} and c_{12} are some positive constants. The term R_j^i defines a characteristic corridor width along spline i for a possible avoidance of agent j. The term $\epsilon(t_n)$ denotes the length of the avoidance trajectory $\delta_j^i(t_n)$ and is computed via $\epsilon(t_n) = c_\epsilon \cdot \frac{||v_j^i(t_n)||_2}{v}$ using the relative flight velocity rate of agents $i, j \in \mathcal{N}$ and a multiplicator c_ϵ. $R_j^i(t_n)$ is a shape function defined as $R_j^i(t_n) = c_\rho \cdot \left(1 + \frac{\psi_{r_j}^i}{\psi_{r_i}^j}\right)$ with all angles lying between $(0, 2\pi]$. As the kinematic model is equivalent to a point mass model, c_ρ is tuned depending on the characteristic size of the simulated agents.

A *Frenet* projection method and its operator $Proj_{S_i}$ is used to link the position of each agent to all splines. The

operator $Proj_{S_i}$ employs a *Newton* method to determine a minimal orthogonal distance (hereinafter referred to as d_j^i). Figure 2 visualizes the idea behind this projection method for agent 1 (blue) and agent 2 (green) moving along the respective splines S_1 and S_2.

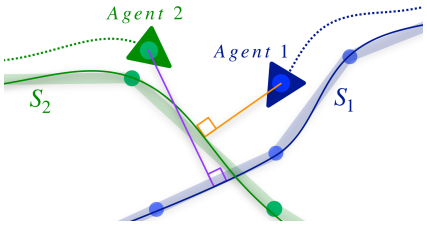

Fig. 2. Two-agent collision avoidance set-up and projections

For $i, j \in \mathcal{N}$, let the set $\mathcal{M}_i = \{(x_i^j, y_i^j), d_j^i \leq r_m\}$ contain all positions of agent j in a neighborhood of radius r_m around (x_i^i, y_i^i). The elastic mode becomes then active when $(x_i^j, y_i^j) \in \mathcal{M}_i$, but $s_i^j \geq s_i^i$.

Remark 1. In a scenario where agent i would come earlier than agent j, s_j^i would lie behind s_i^i. In this case, a *first-come first-served rule* would apply.

Note that a solution of the bilevel dynamic scheduling problem (1) - (2) yields reference paths free of collisions with obstacles. However, since agents may deviate from these optimal paths to avoid each other, they may collide with obstacles outside of the reference trajectories. To improve the safety concept of our approach, the obstacles are oversized on purpose by approximately twenty percents to provide each agent with an acceptable flight corridor risk margin.

C. Coordinated Linearized Model Predictive Controller

Our approach considers all agents separately and assumes that each agent knows the position of all other ones. In each step of the MPC algorithm, an OCP has to be solved for every agent. Using (6) as the underlying model, the QP-OCP can be formulated as [11]:

Problem 3 (QP-OCP).

$$\min_{X,u} \quad \frac{1}{2}\int_{t_0}^{t_f} \begin{bmatrix} X(t)^\top & u(t)^\top \end{bmatrix} \begin{bmatrix} Y(t) & 0 \\ 0 & S(t) \end{bmatrix} \begin{bmatrix} X(t) \\ u(t) \end{bmatrix} dt \tag{9a}$$

subject to the process dynamics (6), collision avoidance constraints (7), some state constraints

$$X_{min}(t) \leq X(t) \leq X_{max}(t) \tag{9b}$$

as well as control constraints

$$u_{min}(t) \leq u(t) \leq u_{max}(t). \tag{9c}$$

For the future implementation of the coordinated LMPC, the kinematic system (6) and Problem 3 are time-discretized with the help of the trapezoidal rule. Therefore, Problem 3 may be treated as one belonging to the QP problem class of the form:

Problem 4 (Predictive Discrete QP-OCP).

$$\min_{z:=(z_n,\ldots,z_{n+N_H})} \sum_{k=n}^{n+N_H} \frac{1}{2} z_k^\top H_k z_k + c_k^\top z_k \tag{10}$$

$$\text{s.t.} \quad \mathcal{A}_k z_k = \alpha_k, \quad \mathcal{B}_k z_k \leq \xi_k \tag{11}$$

with $z_k^\top = [X_k^\top \; u_k^\top]$, $H_k \in \mathbb{R}^{n_z \times n_z}$ positive semi-definite, $c_k \in \mathbb{R}^{n_z}$, $\mathcal{A}_k \in \mathbb{R}^{n_\mathcal{A} \times n_z}$, $\alpha_k \in \mathbb{R}^{n_\mathcal{A}}$, $\mathcal{B}_k \in \mathbb{R}^{n_\mathcal{B} \times n_z}$ and $\xi_k \in \mathbb{R}^{n_\mathcal{B}}$.

To obtain a solution, we use a local semi-smooth Newton method developed at our institute. Its clear advantage is given by the possibility to exploit a special structure of the system, which can be rearranged as a (sparse) band matrix. For the sake of brevity, the idea behind the method is not further explained here. The interested reader is instead referred to [11], [18].

The computational structure may be found in Algorithm 1. For each agent, before solving Problem 4, all projections are computed and stored for all prediction points of the LMPC method. The agents' positions are then regularized in order to find the actual arc length s_i^i of agent i located at (x_i^i, y_i^i). Indeed, depending on the lateral deviation $r_i^i(t_n)$, the arc length may be underevaluated using $s_i^i(t_n) = v \cdot t_n$ and needs to be therefore regularized. Subsequently, all relative velocities are computed in order to calculate all coefficients required for the collision avoidance method, see Subsection IV-B. Problem 4 is finally solved for all agents $i,\ i \in \mathcal{N}$.

Algorithm 1 Coordinated LMPC

1: Measure or estimate all states X_n at t_n of all N agents
2: For all agents $i : 1 \rightarrow N$:
3: Project the position of agent i on all j splines
4: Regularize the s_i^i from the actual current positions
5: Compute relative velocities $||v_j^i(t_n)||_2$
6: If agent i's starting time $t_i \geq t_n$:
7: Solve Problem 4 and find u_n^i
8: Set $n+1 \leftarrow n$ and go to 1

V. ACHIEVED RESULTS

This section presents results achieved by the coupled method elaborated in this work. The first part of the numerical experiments was conducted on an Intel(R) Core i7 computer with a 2.80GHz-CPU and an 8GB-RAM, the second one on an Intel(R) Core i7 computer with a 3.60GHz-CPU and an 16GB-RAM. In table I, all relevant parameters are given for the whole algorithm. In numerical tests, a 2D setting with $d = 2$ was considered. All agents were supposed to be identical small-sized fixed-wing drones.

A. Dynamic Scheduling

Positions of agents and other objects inside the state space $\Omega = [-200, 200] \times [-200, 200]$ are defined in [m]. For $x_i \in \Omega_S(t),\ i \in \mathcal{N}$, the control set $U_i(x_i)$ is represented with the help of the polar coordinates as:

$$U_i(x_i) = \left\{ [\cos\theta_i, \sin\theta_i\ ]^\top \;\middle|\; \theta_i \in [0, 2\pi) \right\}.$$

The starting positions of agents are equal to $\overline{x}_1 = [-110, 190]^\top, \overline{x}_2 = [-80, 190]^\top, \overline{x}_3 = [-50, 190]^\top, \overline{x}_4 = [-20, 190]^\top, \overline{x}_5 = [-190, 50]^\top, \overline{x}_6 = [-180, 50]^\top, \overline{x}_7 = [-170, 50]^\top$ and $\overline{x}_8 = [-160, 50]^\top$. All agents move with a constant velocity of $v = 10\left[\frac{\text{m}}{\text{s}}\right]$. All obstacles are represented by circles with radius $30[\text{m}]$ each. They move periodically along two different circular trajectories with respective velocities $v_{Q,1}$ and $v_{Q,2}$. The centers of the trajectories are located both in the middle of the state space and have respective radii $R_{Q,1}$ and $R_{Q,2}$. The target regions are assumed to be circles with radius $10[\text{m}]$ each. Target 1 and 2 are located at $[180, -20]^\top$ and $[20, -180]^\top$, respectively. The constants $t_{i,min}$ and $t_{i,max}$ were chosen for all agents to be equal to $0[\text{s}]$ and $300[\text{s}]$, respectively. In order to force a collision, the starting times of agent 1 and 5 and their targets were predefined such that $t_1 = t_5 = 0[\text{s}]$ and $\eta_{11} = \eta_{52} = 1$.

Note that the most time-consuming part here is the computation of as many value functions as there are targets. Once these data are given, the solver requires $2.58[\text{s}]$ on average to solve the final single-level MILP.

B. Coordinated Eight-Agents LMPC

The time horizon is $t_f = 5[\text{s}]$ with 100 time points using a uniform grid. The frequency of the LMPC is therefore equal to $20[\text{Hz}]$. The length of all curves γ^i ranges from $290[\text{m}]$ to $390[\text{m}]$. The weight matrices $Y(t) \in \mathbb{R}^{4\times4}$ and $S(t) \in \mathbb{R}$ presented in Problem 3 are chosen as

$$Y = \begin{bmatrix} h \cdot 10^5 & 0 & 0 & 0 \\ 0 & h \cdot 10^4 & 0 & -h \\ 0 & 0 & 0 & 0 \\ 0 & -h & 0 & h \end{bmatrix}, \quad S = \begin{bmatrix} h \cdot 10^4 \end{bmatrix}.$$

The scenario considered by our coupled approach and some results are roughly summarized in Figure 3. The plots visualize positions of all eight agents and their collision-free trajectories at a *snapshot* time t with $t = 5, 25, 50, 100[\text{s}]$. The obstacles shown as orange circles move periodically following circular trajectories. As it can be seen, the splines followed by agents 1 and 5 cross. However, they are prevented from colliding with each other by the coordinated LMPC. The collision avoidance effects are clearly shown in Figure 4. After the collision avoidance phase, their trajectories converge to the respective reference paths. The arc length regularization introduced in Algorithm 1 is indeed essential during the collision avoidance phase. Without it, the path tracking quality would be impaired.

A full video of the scenario may be found at `https://youtu.be/RNDx6I0ezPs`. Note that in the visualization, each agent reaches the target assigned to it up to a tolerance of $\mathcal{O}(h_\Omega)$, where h_Ω is the step size of the uniform space grid in Problem 2.

TABLE I

PARAMETERS RELEVANT FOR PATH PLANNING AND TRACKING

Parameter	Value	Parameter	Value
N	8	v	$10\left[\frac{\text{m}}{\text{s}}\right]$
Q	6	$R_{Q,1}$ / $R_{Q,2}$	50 / 120[m]
T	2	$v_{Q,1}$ / $v_{Q,2}$	5.23 / 12.56 $\left[\frac{\text{m}}{\text{s}}\right]$
σ	1	β	3
t_f	5[s]	N_H	100
$h = \frac{t_f}{N_H}$	0.05[s]	$\bar{r}$ / $\underline{r}$	±3[m]
r_m	22.5[m]	R_{max} / R_{min}	±0.25[m]
u_{max} / u_{min}	$\pm0.3\left[\frac{\text{rad}}{\text{s}^2}\right]$	κ_{max} / κ_{min}	$\pm0.3\left[\frac{\text{rad}}{\text{s}}\right]$
c_{11} / c_{12}	50 / 50	c_2	4
c_ϵ	11	c_ρ	0.045

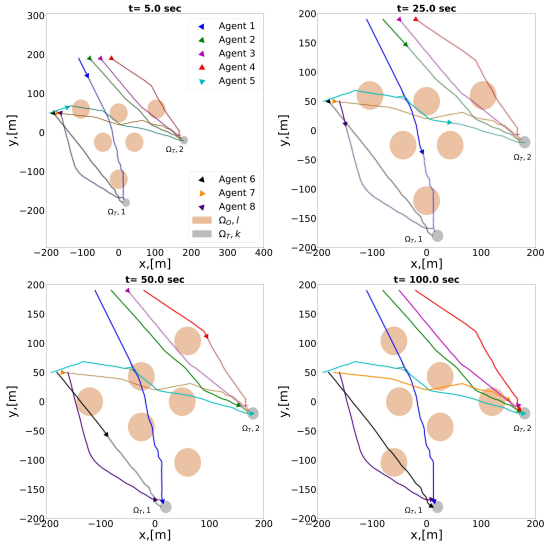

Fig. 3. Trajectories of eight agents at $t = 5.0$ seconds, $t = 25.0$ seconds, $t = 50.0$ seconds and $t = 100.0$ seconds

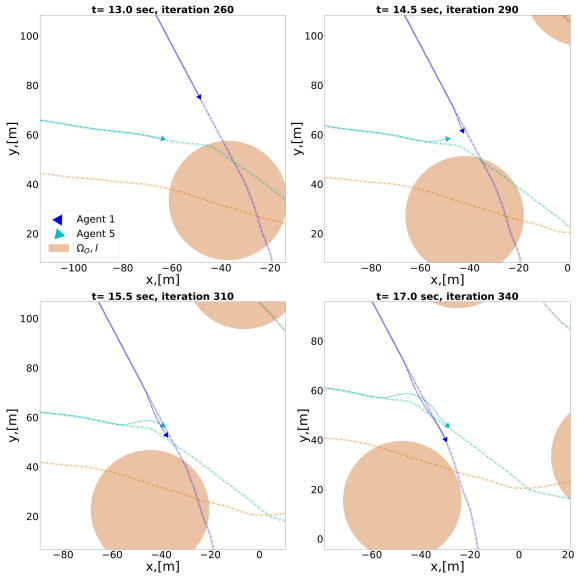

Fig. 4. Collision avoidance of two agents at $t = 13.5$ seconds, $t = 14.5$ seconds, $t = 15.5$ seconds and $t = 17.0$ seconds

In Figure 5, the states $\psi(t)$ and $\psi_r(t)$ are plotted for t ranging from $10[\text{s}]$ to $25[\text{s}]$ with a collision phase between $12.7[\text{s}]$ and $17.7[\text{s}]$. Yaw angles of agents 1 and 5 are characterized by a smooth evolution and oscillations around

the respective reference yaw angles because of $u_1(t)$ and $u_5(t)$, respectively, see Figure 6.

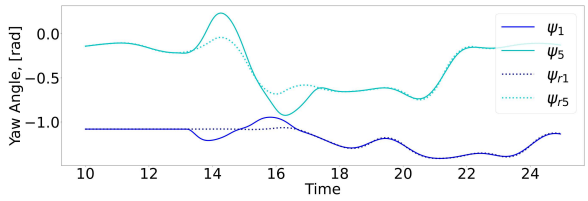

Fig. 5. States ψ and ψ_r of agents 1 and 5 during the collision avoidance phase

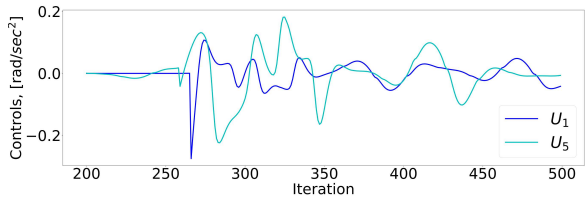

Fig. 6. Controls u of agents 1 and 5 during the collision avoidance phase

Figure 7 shows the lateral deviation $r(t)$. Outside of the collision avoidance neighborhood, the maximum deviation's absolute amplitude is about $0.53[\mathrm{m}]$ and occurs for agent 3. During collision avoidance, the maximum absolute deviation attained is equal to $3.09[\mathrm{m}]$ and $1.25[\mathrm{m}]$ for agent 1 and 5, respectively.

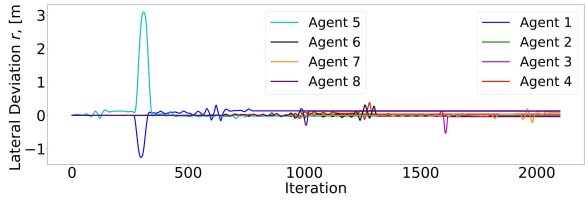

Fig. 7. Lateral deviation per iteration for each agent

For all time iterations of the discrete LMPC-OCP method, the required computational time per agent was under the LMPC limit of $0.05[\mathrm{s}]$ (i.e. $20[\mathrm{Hz}]$). For that reason, our method has proven to be fast and efficient.

VI. CONCLUSION AND OUTLOOK

This paper proposed a novel multi-agent collision-free method, which couples path planning on the basis of dynamic scheduling and MPC-based path tracking. Exploiting a coordinated MPC concept and thus avoiding complex centralized systems of equations, the method elaborated in this work has proven to be fast and efficient even during inter-agent collision avoidance maneuvers.

The approach can be extended towards systems with time-dependent flight velocity profiles. In the path planning part, the objective function of the lower level problem (2) should be reformulated. Moreover, an additional velocity control variable should be introduced. This would imply an increase in the dimension of the control space by one and hence larger computational times. In the path tracking part, non-constant velocities could be easily handled by substituting a constant velocity v by a function $v(t)$ and adding one control feedback rule to the kinematic model (4). Note that the generic kinematic model of the MPC algorithm may be adapted to multiple platforms to increase its versatility and extend its application domains. Future research projects involve a 3D extension of our approach and its experimental validity check.

REFERENCES

[1] Morgan Stanley Research, "Are flying cars preparing for takeoff?," 01 2019.

[2] F. Borrelli, T. Keviczky, G. J. Balas, G. Stewart, K. Fregene, and D. Godbole, "Hybrid decentralized control of large scale systems," in *Hybrid Systems: Computation and Control* (M. Morari and L. Thiele, eds.), (Berlin, Heidelberg), pp. 168–183, Springer Berlin Heidelberg, 2005.

[3] A. Richards and J. How, "Aircraft trajectory planning with collision avoidance using mixed integer linear programming," in *Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301)*, vol. 3, pp. 1936–1941 vol.3, 2002.

[4] V. Nikitina, S. Rogovs, and M. Gerdts, "Piecewise linear value function approximations in nonlinear dynamic scheduling problems with vtols." `https://arxiv.org/abs/2303.03351`, 2023.

[5] M. Castillo-Lopez, S. A. Sajadi-Alamdari, J. L. Sanchez-Lopez, M. A. Olivares-Mendez, and H. Voos, "Model predictive control for aerial collision avoidance in dynamic environments," *2018 26th Mediterranean Conference on Control and Automation (MED)*, pp. 1–6, 2018.

[6] B. Lindqvist, S. S. Mansouri, A.-a. Agha-mohammadi, and G. Nikolakopoulos, "Nonlinear mpc for collision avoidance and control of uavs with dynamic obstacles," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6001–6008, 2020.

[7] M. Werling and D. Liccardo, "Automatic collision avoidance using model-predictive online optimization," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pp. 6309–6314, 2012.

[8] M. A. Hurni, P. Sekhavat, and M. I. Ross, "Autonomous trajectory planning using real-time information updates," 08 2008.

[9] A. Al-Kaff, F. Garcia, D. Martin Gomez, A. de la Escalera, and J. Armingol, "Obstacle detection and avoidance system based on monocular camera and size expansion algorithm for uavs," *Sensors*, vol. 17, p. 1061, 05 2017.

[10] P. Falcone, M. Tufo, F. Borrelli, J. Asgari, and H. E. Tseng, "A linear time varying model predictive control approach to the integrated vehicle dynamics control problem in autonomous systems," in *2007 46th IEEE Conference on Decision and Control*, pp. 2980–2985, 2007.

[11] A. Britzelmeier and M. Gerdts, "A nonsmooth newton method for linear model-predictive control in tracking tasks for a mobile robot with obstacle avoidance," *IEEE Control Systems Letters*, vol. 4, no. 4, pp. 886–891, 2020.

[12] B. Gutjahr, L. Gröll, and M. Werling, "Lateral vehicle trajectory optimization using constrained linear time-varying mpc," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 6, pp. 1586–1595, 2017.

[13] R. Bellman, *Dynamic Programming*. Princeton, NJ, USA: Princeton University Press, 1 ed., 1957.

[14] S. N. Kružkov, "Generalized solutions of the hamilton-jacobi equations of eikonal type. i. formulation of the problems; existence, uniqueness and stability theorems; some properties of the solutions," *Mathematics of the USSR-Sbornik*, vol. 27, p. 406, 04 1975.

[15] M. Gerdts, S. Rogovs, and G. Valenti, *Solving Complex Intersection Management Problems Using Bi-level MINLPs and Piecewise Linearization Techniques*, pp. 255–273. Springer International Publishing, 2022.

[16] Gurobi Optimization LLC, "Gurobi optimizer reference manual 9.1," 03 2020.

[17] Hao Su, Justin Solomon, "Curves: Parametrization, curvature, frenet frame." `https://geoml.github.io/Lectures/L3_Curves_I.pdf`.

[18] R. Mifflin, "Semismooth and semiconvex functions in constrained optimization," *SIAM Journal on Control and Optimization*, vol. 15, pp. 957–972, 1977.