Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2023.1120000

Sequential Distributed Optimization for Cooperative Collision Avoidance via Agent-Wise CCP-PSM

GYUBIN PARK[1,2], (Student Member, IEEE), DOHOON LEE[1,2], (Student Member, IEEE) and JONG-HAN KIM[1,2,3], (Member, IEEE)

[1]Department of Aerospace Engineering, Inha University, Incheon, South Korea
[2]Program in Aerospace Systems Convergence, Inha University, Incheon, South Korea
[3]Aerospace Systems Research Institute, Inha University, Incheon, South Korea

Corresponding author: Jong-Han Kim (e-mail: jonghank@inha.ac.kr)

This work was supported in part by Korea Research Institute for defense Technology Planning and advancement (KRIT) Grant funded by Defense Acquisition Program Administration(DAPA) (No. KRIT-CT-22-083, Cloud Computing-Based Learning/Inference Engine Technology), and in part by Unmanned Vehicles Core Technology Research and Development Program through the National Research Foundation of Korea (NRF) and Unmanned Vehicle Advanced Research Center (UVARC) funded by the Ministry of Science and ICT, Republic of Korea (No. NRF-2021M3C1C1A02099428).

ABSTRACT

This paper proposes Agent-wise CCP-PSM, a distributed optimization framework for collision avoidance in high-density multi-agent systems. The proposed method linearizes non-convex distance constraints using the Convex–Concave Procedure (CCP) and solves each agent's local problem via a computationally efficient Projected Subgradient Method (PSM). By incorporating collision avoidance as soft penalties, the framework improves numerical robustness and maintains solution availability in congested scenarios. Furthermore, a Gauss–Seidel-based sequential update enables each agent to utilize the most recent information from others while keeping the subproblem dimension independent of the total number of agents, improving coordination efficiency.

Within a receding-horizon framework, the method generates collision-avoidance trajectories in real time while empirically reducing minimum-separation violations. Its computational complexity with respect to the number of agents and planning horizon is analyzed and validated through simulations. Results demonstrate that the proposed framework provides a favorable balance among minimum-separation performance, computational efficiency, and scalability across diverse scenarios.

INDEX TERMS Multi-Agent System, Cooperative Collision Avoidance, Distributed Optimization, Convex-Concave Procedure, Projected Subgradient Method.

I. INTRODUCTION

THE rapid evolution of Unmanned Aircraft Systems (UAS) and the emergence of Urban Air Mobility (UAM) have led to increasingly congested airspaces. In such environments, multiple aircraft operate in close proximity, making scalable and computationally tractable collision avoidance a central requirement for safe operation. Over the past decades, a broad spectrum of approaches has been developed, ranging from reactive collision avoidance strategies to optimization-based trajectory planning frameworks.

Reactive methods initiate avoidance maneuvers when conflicts become imminent. Representative examples include the Velocity Obstacle (VO) framework [1] and its extensions such as Reciprocal Velocity Obstacles (RVO) [2], Optimal Reciprocal Collision Avoidance (ORCA) [3], and Hybrid RVO (HRVO) [4]. Artificial Potential Field (APF) methods [5], [6] also provide lightweight mechanisms for real-time navigation. While these techniques are computationally efficient and well suited for decentralized implementation, their performance may degrade in dense or highly interactive scenarios due to their susceptibility to local minima and a lack of explicit anticipatory coordination among agents, which can lead to avoidance failure in congested airspaces.

To enhance global consistency and trajectory smoothness, optimization-based planning methods have gained increasing attention. Approaches such as TrajOpt [7] and CHOMP [8]

employ gradient-based optimization to compute collision-free trajectories, while GPMP [9] leverages probabilistic inference for efficient motion planning. In aerospace applications, Sequential Convex Programming (SCP) and Successive Convexification (SCvx) [10], [11] have demonstrated effectiveness in handling nonlinear dynamics and constraints. Despite these advances, extending such frameworks to cooperative multi-agent collision avoidance remains challenging, primarily because inter-agent separation constraints are inherently non-convex and strongly coupled across agents. Furthermore, these frameworks typically treat separation requirements as hard constraints and rely on heavy-duty numerical solvers. In densely populated environments, such formulations frequently encounter numerical infeasibility, where no valid solution exists for the strict constraints, potentially causing system-level failures.

From a coordination perspective, existing distributed planning strategies [12]–[14], including DMPC and cooperative trajectory generation, typically rely on parallel (Jacobi-style) updates. While theoretically sound, such schemes often require multiple communication rounds to propagate consistent information, limiting convergence speed in communication-constrained or high-speed scenarios. Recent consensus studies show that asynchronous communication, cooperation–competition interactions, switching topology, and communication delays can affect information propagation and convergence rates in distributed multi-agent networks [15], [16]. Although these works focus on consensus tracking rather than trajectory-level collision avoidance, they motivate coordination structures that reduce repeated communication rounds. However, distributed trajectory-planning approaches based on repeated parallel updates still often struggle to simultaneously ensure feasibility, scalability, and real-time implementability under tightly coupled multi-agent interactions.

Motivated by these limitations, this paper proposes a distributed cooperative collision avoidance framework termed Agent-wise CCP-PSM. The proposed method is designed to bridge the gap between the theoretical rigor of Difference-of-Convex (DC) programming [17] and the requirements of embedded, real-time implementation. Unlike traditional parallel schemes, the proposed framework adopts an agent-wise sequential (Gauss–Seidel) update structure. This enables each agent to incorporate the most recently updated trajectories within the same coordination cycle, improving information propagation while reducing the need for repeated consensus-style communication rounds.

To enhance numerical robustness, the inter-agent separation requirements are reformulated as rectifier-based soft penalty functions. This formulation alleviates the strict infeasibility issue that frequently arises in adversarial initializations or highly congested configurations where hard constraints cannot be simultaneously satisfied. As a result, the penalized optimization problem remains well posed and can continue to produce dynamically feasible best-effort trajectories, allowing graceful degradation in separation performance rather than abrupt infeasibility or solver failure. By exploiting the underlying DC structure [18], [19], we integrate a Projected Subgradient Method (PSM) directly into the Convex-Concave Procedure (CCP), resulting in a lightweight implementation grounded in classical convex optimization theory [20]. Unlike conventional CCP or sequential convexification methods such as SCP that rely on repeatedly solving convex subproblems, the proposed approach avoids external solvers and significantly reduces computational overhead. Furthermore, in contrast to distributed MPC-based approaches that require receding-horizon optimization and frequent information exchange, the proposed framework directly refines trajectories with reduced communication burden. This integrated design addresses coupling, infeasibility, and computational efficiency in a unified and practically scalable manner.

The novelty of the proposed framework lies in this integrated algorithmic structure rather than in any single component alone. Compared with ORCA-type reactive methods, the proposed method optimizes finite-horizon trajectories and explicitly accounts for future interactions. Compared with CBF-based safety filters, it addresses trajectory-level coordination rather than applying a myopic local correction, while acknowledging that the soft-penalty formulation does not provide a hard safety certificate. Compared with ADMM- or DMPC-based distributed optimization methods, which often require iterative coordination, consensus updates, or frequent information exchange, the proposed Gauss–Seidel coordination propagates updated trajectory information sequentially within each coordination cycle. Finally, compared with conventional CCP, SCP, or SCvx implementations that repeatedly invoke external convex solvers, the proposed CCP–PSM solver exploits the affine dynamics constraints to perform closed-form projections, resulting in a lightweight implementation.

The main contributions of this paper are summarized as follows:

Structural Efficiency: We introduce a sequential agent-wise Gauss–Seidel coordination structure that propagates the most recently updated trajectories within each coordination cycle. This structure reduces the need for repeated consensus-style communication rounds while keeping each local subproblem dimension independent of the total number of agents.

Computational Tractability: By combining CCP with a tailored PSM and closed-form projection onto the affine dynamics manifold, the proposed framework eliminates the need for external convex solvers. This enables predictable low-latency computation suitable for real-time receding-horizon implementation.

Numerical Robustness: The rectifier-based soft-penalty formulation mitigates infeasibility of the penalized optimization problem in high-density scenarios. As a result, the framework maintains solution availability and reduces minimum-separation violations in a best-effort manner, while exact satisfaction of the minimum-separation requirement remains dependent on the penalty weight and finite iteration budget.

Quantitative Validation: Extensive simulations demonstrate that the proposed framework achieves a favorable

trade-off among minimum separation distance, minimum-separation violations, control effort, computation time, and scalability compared with representative reactive and optimization-based baselines.

II. PRELIMINARIES

A. CONVEX-CONCAVE PROCEDURE

The Convex-Concave Procedure (CCP) is an iterative method designed to solve difference-of-convex (DC) problems. It decomposes a non-convex function into the difference of two convex functions and then searches for the optimal solution through the convexification process:

$$\underset{x}{\text{minimize}} \quad g(x) - h(x).$$

Here, the objective function $f(x) = g(x) - h(x)$ is a non-convex function, expressed as the difference between two convex functions, $g(x)$ and $h(x)$. By linearizing the function $h(x)$ around a reference point z, we obtain the following:

$$\hat{h}(x;z) = h(z) + \nabla h(z)^\top (x - z).$$

Since the function $h(x)$ is convex, the linearized function $\hat{h}(x;z)$ satisfies the inequality $\hat{h}(x;z) \leq h(x)$ for all x. By substituting $\hat{h}(x;z)$ for $h(x)$ in the original objective function $f(x) = g(x) - h(x)$, we obtain the convexified objective function $\hat{f}(x;z)$, which is written as:

$$\hat{f}(x;z) = g(x) - \hat{h}(x;z) \geq f(x).$$

At each CCP iteration, the convexified surrogate problem is solved, or approximately solved, to obtain the next linearization point $x^{(k+1)}$. This process is repeated to seek a stationary point of the original non-convex objective $f(x)$:

$$\begin{aligned} x^{(k+1)} &= \underset{x}{\operatorname{argmin}}\, \hat{f}(x;x^{(k)}) \\ &= \underset{x}{\operatorname{argmin}}\, g(x) - \hat{h}(x;x^{(k)}). \end{aligned}$$

Since $f(x)$ is non-convex, CCP does not guarantee convergence to the global minimum. Under standard DC assumptions, CCP is commonly used to seek a stationary point of the penalized local problem. In this work, this property is invoked only for the local subproblem with fixed neighboring trajectories, while the full distributed receding-horizon scheme is evaluated empirically.

B. PENALTY FUNCTION FOR COLLISION AVOIDANCE

In multi-agent systems, collision avoidance constraints typically exhibit a non-convex nature, which constitutes a major factor that degrades convergence stability in numerical optimization processes. To address this issue, this study introduces a rectifier function in the form of a hinge loss, which is widely used in optimization theory, and incorporates a penalty into the objective function that is proportional to the degree of constraint violation. The penalty term $\mathcal{P}$ is defined as:

$$\mathcal{P} = \left(d_{\text{safety}} - \|p^{(k)} - p^{(l)}\|\right)_+ = \max\left(0,\ d_{\text{safety}} - d_t\right),$$

where $p^{(k)}$ and $p^{(l)}$ denote the positions of agents k and l, respectively, and $d_t = \|p^{(k)} - p^{(l)}\|$ represents the Euclidean distance between the two agents. The above expression can be decomposed into the DC structure using the following mathematical identity:

$$\max\left(0,\ d_{\text{safety}} - d_t\right) = \max\left(d_{\text{safety}},\ d_t\right) - d_t.$$

Here, d_t is an ℓ_2-norm and thus a convex function, and the max operator is also convex since it represents the pointwise maximum of a constant and an ℓ_2-norm. Consequently, the rectifier-based penalty function associated with the collision avoidance constraint admits a difference-of-convex structure of the form $g(x) - h(x)$. This exact reformulation enables the resulting penalized problem to be efficiently handled using the convex–concave procedure.

Throughout this paper, a minimum-separation violation refers to the violation of the prescribed separation requirement d_{safety}. Given the minimum pairwise distance $d_{\min}$ over a trajectory, the violation magnitude is measured as $\max(0, d_{\text{safety}} - d_{\min})$. This metric quantifies violation of the prescribed separation requirement and should be distinguished from a hard physical collision claim.

C. PROJECTED GRADIENT DESCENT WITH LINEAR EQUALITY CONSTRAINT

Consider a convex optimization problem consisting of a differentiable convex function $f(x)$ and linear equality constraints $g(x) = Ax + b$:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & g(x) = 0. \end{aligned}$$

While the global optimum admits a closed-form solution when $f(x)$ is quadratic, such a solution is generally unavailable for more general convex objectives. Since the problem includes linear equality constraints, a Projected Gradient Descent (PGD) scheme is employed. In this approach, an intermediate solution $x_{j+1,\text{temp}}$ is first obtained via a gradient descent step, which is then projected onto the feasible set to satisfy the constraints:

$$x_{j+1} = \Pi_{\mathcal{C}}\left(x_j - \alpha \nabla_x f(x_j)\right) = \Pi_{\mathcal{C}}\left(x_{j+1,\text{temp}}\right),$$

where α is the step size, $\mathcal{C} = \{x \mid Ax + b = 0\}$ denotes the feasible set, and $\Pi_{\mathcal{C}}(\cdot)$ represents the projection operator onto $\mathcal{C}$. The projection process can be defined as the following optimization problem:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \frac{1}{2}\|x - x_{j+1,\text{temp}}\|^2 \\ \text{subject to} \quad & Ax + b = 0. \end{aligned}$$

Assuming that A has full row rank, the above problem admits a closed-form solution as follows:

$$x_{j+1} = x_{j+1,\text{temp}} - A^\top \left(AA^\top\right)^{-1} \left(Ax_{j+1,\text{temp}} + b\right).$$

III. DISTRIBUTED COLLISION AVOIDANCE

A. PROBLEM FORMULATION

The cooperative collision avoidance problem for K aircraft can be formulated as a finite-horizon optimization problem, as follows:

$$\begin{aligned}
\text{minimize} \quad & \sum_{t=0}^{T-1}\sum_{k=1}^{K} \|u_t^{(k)}\|^2 && \text{(1a)}\\
\text{subject to} \quad & p_{t+1}^{(k)} = p_t^{(k)} + \Delta t v_t^{(k)}, && \text{(1b)}\\
& v_{t+1}^{(k)} = v_t^{(k)} + \Delta t u_t^{(k)}, && \text{(1c)}\\
& p_0^{(k)} = p_{\text{init}}^{(k)}, \qquad v_0^{(k)} = v_{\text{init}}^{(k)}, && \text{(1d)}\\
& p_T^{(k)} = p_{\text{des}}^{(k)}, \qquad v_T^{(k)} = v_{\text{des}}^{(k)}, && \text{(1e)}\\
& \|p_t^{(k)} - p_t^{(l)}\| \geq d_{\text{safety}}, \quad \forall l \neq k. && \text{(1f)}
\end{aligned}$$

In problem (1), the subscript of each variable denotes the timestep, while the superscript denotes the index of the agent. Here, $p_t^{(k)} \in \mathbb{R}^2$, $v_t^{(k)} \in \mathbb{R}^2$ and $u_t^{(k)} \in \mathbb{R}^2$ are position, velocity and control input of k-th agent at time t, while Δt and T denote the timestep and the horizon size, respectively. Thus, the objective function in (1a) minimizes the control effort for each aircraft, and the constraints (1b) and (1c) represent the linear discrete-time dynamics of the aircraft. The constraints (1d)-(1e) specify the initial and final states of the aircraft, while (1f) ensures that the distance between any two aircraft is at least d_{safety}, which serves as the collision avoidance constraint. By eliminating the state variables using the lifted system representation, the initial conditions, system dynamics, and terminal boundary conditions can be aggregated into a single linear equality constraint $Mu^{(k)} + n^{(k)} = 0$ where $M \in \mathbb{R}^{4\times 2T}$ and $n^{(k)} \in \mathbb{R}^4$. If the objective function (1a) is separated with respect to the control inputs of each agent, $u^{(k)} \in \mathbb{R}^{2T}$, problem (1) can be represented as a subproblem in the following decoupled form based on the linearized dynamics equations:

$$\begin{aligned}
\underset{u^{(k)}}{\text{minimize}} \quad & \|u^{(k)}\|^2 \\
\text{subject to} \quad & Mu^{(k)} + n^{(k)} = 0, \\
& d_t \geq d_{\text{safety}}, \quad \forall l \neq k,
\end{aligned} \qquad \text{(2)}$$

where

$$d_t = \Big\| \underbrace{(G_t u^{(k)} + h_t^{(k)})}_{p_t^{(k)}} - \underbrace{(G_t u^{(l)} + h_t^{(l)})}_{p_t^{(l)}} \Big\|.$$

In problem (2), the matrix $G_t \in \mathbb{R}^{2\times 2T}$ and the vector $h_t^{(k)} \in \mathbb{R}^2$ are used to map the control input $u^{(k)}$ to the k-th agent's position $p_t^{(k)}$ through the lifted system representation. A fundamental challenge in solving (1) is that the collision avoidance constraint is intrinsically coupled with the decision variables of other agents, $u^{(l)}$, which prevents direct decomposition of the centralized problem.

To enable a decentralized solution, each agent k treats the trajectories of neighboring agents as fixed parameters rather than decision variables. This parameterization allows the coupled multi-agent constraint to be evaluated locally, yielding an optimization problem that depends only on the individual decision variable $u^{(k)}$. To avoid oscillatory behaviors or deadlocks that may arise from simultaneous updates across agents, a sequential coordination strategy based on a Gauss–Seidel scheme is adopted with a predefined priority ordering, such as the agent index. This design improves information propagation within a coordination cycle because later agents can exploit the most recently updated trajectories of earlier agents. However, it also introduces an inherent trade-off: the resulting trajectories may depend on the selected update order.

In the present implementation, a fixed index-based ordering is used to keep the coordination protocol simple, deterministic, and reproducible. The order dependence can be mitigated in practice by cyclically shifting the priority order, randomizing the update order between coordination cycles, or assigning priority according to conflict severity. A detailed investigation of adaptive ordering strategies is left for future work.

Under this scheme, agents perform optimization sequentially within each coordination cycle. When agent k solves its subproblem at the $(m+1)$-th coordination iteration, it utilizes the most recent trajectory information available from the network:

$$u^{(l)} = \begin{cases} u^{(l),m+1}, & l < k \quad \text{(already updated)} \\ u^{(l),m}, & l > k \quad \text{(not yet updated)} \end{cases} \qquad \text{(3)}$$

where the superscript m denotes the index of the global coordination loop. The coordination procedure at iteration $m+1$ consists of the following steps:

1) *Local Optimization:* Each agent k solves its subproblem to obtain an updated trajectory $u^{(k),m+1}$, while the trajectories of other agents are held fixed as defined above.
2) *Information Exchange:* After completing the local optimization, the updated trajectory $u^{(k),m+1}$ is immediately communicated to subsequent agents in the sequence.

Consequently, this sequential coordination process significantly reduces the computational burden compared to a fully centralized optimization problem (1). In a centralized formulation, the number of decision variables and coupled constraints grows rapidly with the total number of agents, which can become prohibitive for large-scale systems. In contrast, the proposed approach decomposes the global problem into K smaller subproblems, each involving a fixed number of decision variables. This decentralized formulation provides a structural basis that admits game-theoretic interpretations [21], [22], where the sequential updates can be viewed as a form of best-response dynamics. By leveraging this framework, the proposed approach effectively enhances scalability and real-time feasibility for multi-agent trajectory planning in dense environments.

While the proposed Gauss–Seidel coordination enables decentralized trajectory updates with improved scalability, its practical implementation requires repeated execution in

Algorithm 1 Distributed Gauss–Seidel Coordination

Input: Initial trajectories $\{u^{(k),0}\}_{k=1}^{K}$
Output: $\{u^{(k),m}\}_{k=1}^{K}$
for $m = 0, 1, 2, \ldots$ **do** ▷ Global coordination cycle
  Execute buffered control inputs from $\{u^{(k),m}\}$
  for $k = 1$ to K **do** ▷ Sequential agent update
    Fix trajectories of other agents via (3)
    Solve local subproblem for agent k
    Obtain updated trajectory $u^{(k),m+1}$
    Broadcast $u^{(k),m+1}$ to the network
  end for
end for

a real-time setting. To this end, the coordination scheme is embedded within a receding-horizon framework, as detailed in the following subsection.

B. LOCAL TRAJECTORY OPTIMIZATION

We now apply the DC framework introduced in Section II to the distributed subproblem of agent k, treating neighboring trajectories as fixed parameters as defined in previous section:

$$\begin{aligned} \underset{u^{(k)}}{\text{minimize}} \quad & (1-\lambda)\|u^{(k)}\|^2 + \lambda \sum_{l\neq k} \sum_{t=1}^{T-1} \left(d_{\text{safety}} - d_t\right)_+ \\ \text{subject to} \quad & Mu^{(k)} + n^{(k)} = 0. \end{aligned} \tag{4}$$

We handle the nonconvex collision avoidance constraints in subproblem (2) by incorporating them into the objective function and formulating the problem as a difference-of-convex function. The difference between d_{safety} and the relative distance between two aircraft is passed through a rectifier function $(\cdot)_+$, which outputs zero if the value is negative and increases the cost otherwise. This leads to the objective function shown in problem (4). The rectifier function can be separated into two convex functions, as shown below:

$$\begin{aligned} \left(d_{\text{safety}} - d_t\right)_+ &= \max\left(d_{\text{safety}} - d_t, 0\right), \\ &= \underbrace{\max\left(d_{\text{safety}},\, d_t\right)}_{\text{convex}} - \underbrace{d_t}_{\text{convex}}. \end{aligned} \tag{5}$$

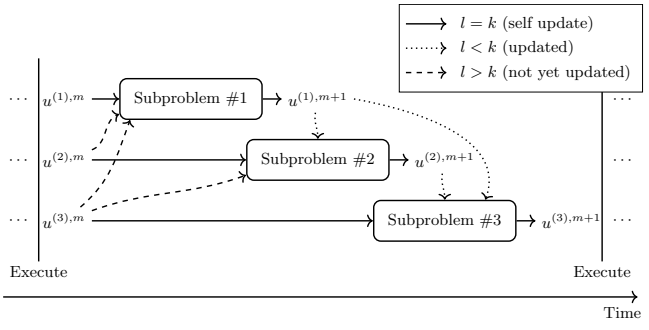


FIGURE 1. Sequential decentralized coordination via Gauss–Seidel scheme ($K = 3$).

It is important to emphasize that the proposed rectifier-based penalty formulation introduces an explicit trade-off between control effort and enforcement of the minimum-separation requirement, governed by the penalty weight $\lambda \in [0, 1]$. For λ close to 1, violations of the minimum separation distance are heavily penalized, and the resulting trajectories empirically exhibit smaller minimum-separation violations. However, for λ close to 0, the optimization may admit solutions with temporary distance violations in exchange for reduced control effort.

Therefore, the proposed framework maintains feasibility of the penalized optimization formulation, while exact satisfaction of the minimum-separation requirement depends on λ and the finite iteration limits. In practice, λ should be selected according to the desired trade-off between control efficiency and separation enforcement. A larger value of λ generally produces more conservative avoidance maneuvers by prioritizing separation recovery, whereas a smaller value reduces control effort but may increase the magnitude or frequency of minimum-separation violations. For dense scenarios, λ should be chosen close to one and jointly tuned with the prediction horizon and the CCP–PSM iteration limits. This guideline is not intended as a closed-form optimal rule; rather, it provides a practical design principle consistent with the sensitivity analysis in Section IV.

$$\begin{aligned} \underset{u^{(k)}}{\text{minimize}} \quad & (1-\lambda)\|u^{(k)}\|^2 \\ & + \lambda \sum_{l\neq k} \sum_{t=1}^{T-1} \{\max\left(d_{\text{safety}},\, d_t\right) - d_t\} \\ \text{subject to} \quad & Mu^{(k)} + n^{(k)} = 0. \end{aligned} \tag{6}$$

Since the objective function in problem (6) consists of terms involving the ℓ_2-norm and the maximum function, and has linear constraints, problem (6) becomes a difference-of-convex problem, which can be solved by applying the CCP algorithm.

First, decompose the objective function $f(u^{(k)})$ into two convex functions, $g(u^{(k)})$ and $h(u^{(k)})$:

$$\begin{aligned} f(u^{(k)}) = \overbrace{(1-\lambda)\|u^{(k)}\|^2 + \lambda \sum_{l\neq k} \sum_{t=1}^{T-1} \max\left(d_{\text{safety}},\, d_t\right)}^{g(u^{(k)})} \\ \underbrace{- \lambda \sum_{l\neq k} \sum_{t=1}^{T-1} d_t}_{h(u^{(k)})}. \end{aligned} \tag{7}$$

Now, the convexification of the objective function can be achieved via linearized $h(u^{(k)})$ around reference point z:

$$\begin{aligned} \hat{f}\left(u^{(k)}; z\right) &= g(u^{(k)}) - h(z) - s_z^\top (u^{(k)} - z), \qquad s_z \in \partial h(z) \\ &= g(u^{(k)}) \\ &\quad - \lambda \sum_{l\neq k} \sum_{t=1}^{T-1} \left\{ \|d_{z,t,l}\| + \frac{d_{z,t,l}^\top G_t (u^{(k)} - z)}{\|d_{z,t,l}\|} \right\}, \end{aligned} \tag{8}$$

where $d_{z,t,l} = G_t(z - u^{(l)}) + h_t^{(k)} - h_t^{(l)}$. Here, $u^{(l)}$ is treated as a constant as defined in (3). Note that when $\|d_{z,t,l}\| = 0$, any vector in the unit ball is a valid subgradient; in practice, a small regularization parameter $\epsilon > 0$ is added to the denominator, i.e., $\|d_{z,t,l}\| \leftarrow \|d_{z,t,l}\| + \epsilon$, to ensure numerical stability.

The CCP reference point is updated at every outer iteration as $z \leftarrow u^{(k)}|^{[i]}$, as summarized in Algorithm 2. This linearization yields a convex surrogate for the local penalized objective under fixed neighboring trajectories. Because the collision-avoidance term is included as a soft penalty rather than a hard constraint, the subproblem remains well defined even when the initial or warm-start trajectory violates the minimum-separation requirement. In such initially infeasible or colliding configurations, the rectifier penalty introduces a nonzero penalty contribution that encourages reduction of minimum-separation violations, while the projection step preserves the affine dynamics and boundary constraints. The regularization parameter ϵ prevents numerical singularities near $\|d_{z,t,l}\| = 0$, but it does not convert the soft-penalty formulation into a hard safety guarantee. Therefore, the modified subproblem can be expressed as:

$$\begin{aligned} \underset{u^{(k)}}{\text{minimize}} \quad & \hat{f}(u^{(k)}; u^{(k)}|^{[i]}) \\ \text{subject to} \quad & Mu^{(k)} + n^{(k)} = 0. \end{aligned} \tag{9}$$

In problem (9), the superscript $[i]$ denotes the index of each CCP iteration. While this convexified subproblem involves a non-differentiable objective function and linear equality constraints, it can be solved using standard constrained optimization techniques, such as Lagrange multipliers or interior-point methods.

However, we employ the Projected Subgradient Method (PSM), which can be viewed as an extension of projected gradient descent to non-smooth convex objectives by replacing the gradient with a subgradient. In contrast to interior-point methods with cubic complexity, each PSM iteration requires only matrix-vector multiplications and a single projection, making it suitable for real-time implementation. The rationale for selecting PSM is that the dynamics and boundary constraints $Mu^{(k)} + n^{(k)} = 0$ define an affine subspace, allowing the projection step to be computed analytically.

To ensure stable convergence under the non-smooth objective, a diminishing step size is adopted in the PSM iterations. Specifically, the step size at the j-th inner iteration is defined as $\alpha_j = \alpha_0/(1+j)$, where α_0 is the initial step size parameter [23].

By iteratively updating the control input along the subgradient and projecting it onto the affine constraint manifold, the solution at the $[i+1]$-th CCP iteration, $u^{(k)}|^{[i+1]}$, can be obtained as follows:

1) *Subgradient Step:* Let $u^{(k)}|_j^{[i+1]}$ denote the control input at the j-th inner iteration of the $(i+1)$-th CCP cycle. The intermediate control input $u^{(k)}_{j+1,\text{temp}}$ is updated along the negative subgradient of the convexified objective $\hat{f}$:

$$u^{(k)}|_{j+1,\text{temp}}^{[i+1]} = u^{(k)}|_j^{[i+1]} - \alpha_j \xi_j, \tag{10}$$

where the subgradient $\xi_j \in \partial_{u^{(k)}} \hat{f}(u^{(k)}|_j^{[i+1]})$ is given by:

$$\xi_j = 2(1-\lambda)u^{(k)}|_j^{[i+1]} + \lambda \sum_{l \neq k} \sum_{t=1}^{T-1} \gamma_{j,t,l}. \tag{11}$$

Based on the DC decomposition and linearizations in (8), the term $\gamma_{j,t,l}$ is determined by the collision state:

$$\gamma_{j,t,l} = \begin{cases} -\dfrac{G_t^\top d_{z,t,l}}{\|d_{z,t,l}\|}, & \|d_{j,t,l}\| < d_{\text{safety}} \\ \beta \dfrac{G_t^\top d_{j,t,l}}{\|d_{j,t,l}\|} - \dfrac{G_t^\top d_{z,t,l}}{\|d_{z,t,l}\|}, & \|d_{j,t,l}\| = d_{\text{safety}} \\ \dfrac{G_t^\top d_{j,t,l}}{\|d_{j,t,l}\|} - \dfrac{G_t^\top d_{z,t,l}}{\|d_{z,t,l}\|}, & \|d_{j,t,l}\| > d_{\text{safety}} \end{cases} \tag{12}$$

where $d_{j,t,l} = G_t(u^{(k)}|_j^{[i+1]} - u^{(l)}) + h_t^{(k)} - h_t^{(l)}$ is the relative distance vector at the current PSM iteration, and $d_{z,t,l}$ is the reference distance vector from the previous CCP iteration as defined in (8), while $\beta \in [0,1]$ corresponds to a valid convex combination of subgradients at the non-differentiable point $\|d_{j,t,l}\| = d_{\text{safety}}$. When $\|d_{j,t,l}\| < d_{\text{safety}}$, the max term becomes constant, and only the linearized concave term contributes to the subgradient.

2) *Projection Step:* After the subgradient update, the intermediate solution is projected onto the affine subspace defined by the linearized dynamics and boundary constraints $Mu^{(k)} + n^{(k)} = 0$. Since $M \in \mathbb{R}^{4\times 2T}$ has full row rank, $(MM^\top)^{-1}$ exists and the projection can be computed in closed form as:

$$\begin{aligned} u^{(k)}|_{j+1}^{[i+1]} = u^{(k)}|_{j+1,\text{temp}}^{[i+1]} \\ - M^\top (MM^\top)^{-1} \left(Mu^{(k)}|_{j+1,\text{temp}}^{[i+1]} + n^{(k)} \right). \end{aligned} \tag{13}$$

Here, the projection matrix $P = I - M^\top (MM^\top)^{-1} M$ and the term $(MM^\top)^{-1}$ depend only on the system dynamics, they can be pre-computed to accelerate the algorithm. In the finite-iteration implementation used in this work, the CCP update is taken as the final PSM iterate,

$$u^{(k)}|^{[i+1]} = u^{(k)}|_{N_{\text{PSM}}}^{[i+1]}. \tag{14}$$

Since $\hat{f}$ involves non-smooth max-norm terms, it is Lipschitz continuous but not smooth. In practice, a small step size is used and the inner loop is terminated after a finite number of iterations.

Consequently, the pseudocode for local trajectory optimization algorithm can be written as Algorithm 2.

Under fixed neighboring trajectories, subproblem (6) is a DC program, and the CCP iterations for this local penalized subproblem seek a stationary point of the corresponding local objective under standard DC assumptions. The projected subgradient and sequential update structure is related to distributed subgradient methods for multi-agent convex optimization [23]. However, because the complete framework involves nonconvex collision penalties, finite inner PSM itera-

Algorithm 2 Local Trajectory Optimization for Agent k

Input: Initial trajectory $u^{(k)}|^{[0]}$, neighbor trajectories $\{u^{(l)}\}_{l\neq k}$
Parameters: d_{safety}, λ, N_{CCP}, N_{PSM}, initial step size α_0
Output: Optimized local control $u^{(k)}$
for $i = 0$ to $N_{\text{CCP}} - 1$ **do**
  $z \leftarrow u^{(k)}|^{[i]}$ ▷ CCP reference point
  $u_0 \leftarrow z$ ▷ Initialize PSM
  for $j = 0$ to $N_{\text{PSM}} - 1$ **do**
    Compute a subgradient ξ_j ▷ via (11)
    $\alpha_j \leftarrow \alpha_0/(1+j)$
    $u_{j+1}^{\text{temp}} \leftarrow u_j - \alpha_j \xi_j$
    $u_{j+1} \leftarrow \Pi_{\mathcal{C}}(u_{j+1}^{\text{temp}})$ ▷ via (13)
  end for
  $u^{(k)}|^{[i+1]} \leftarrow u_{N_{\text{PSM}}}$
end for
return $u^{(k)} \leftarrow u^{(k)}|^{[N_{\text{CCP}}]}$

tions, Gauss–Seidel trajectory updates, and receding-horizon replanning, this connection does not imply global optimality, convergence to a Nash equilibrium, or formal convergence of the full distributed algorithm. Accordingly, the convergence behavior of the overall framework is assessed empirically in Section IV.

C. RECEDING-HORIZON FRAMEWORK

To enable real-time implementation of the proposed distributed optimization scheme, the coordination process is embedded within a receding-horizon execution framework. While a single coordination cycle yields a well-defined penalized solution under fixed neighboring information, its performance in dynamic environments can degrade due to execution latencies and the sequential nature of information propagation. This is addressed by repeatedly executing the coordination cycles in a closed-loop manner, where the previous coordination results serve as an informed warm-start for the subsequent cycle.

Let $m \in \{0, 1, 2, \dots\}$ denote the index of the global coordination cycle. At the beginning of cycle m, each agent k holds a nominal control sequence $u^{(k),m}$. During the current cycle, agents sequentially update their trajectories using the Gauss–Seidel coordination scheme described in Algorithm 1.

To synchronize the distributed updates with a physical sampling clock, it is assumed that each local subproblem is solved within a dedicated time slot Δt. Consequently, a complete global update for K agents is finalized after $K\Delta t$, defining the total coordination latency of one cycle. During this period, each agent continues to execute control inputs from the previous trajectory buffer. Specifically, instead of waiting for the completion of the current coordination cycle, agents apply the remaining control sequence (e.g., $u_1^{(k),m}, u_2^{(k),m}, \dots$) from the prior solution. This execution-ahead strategy ensures control continuity even when the total coordination time scales with the number of agents.

A critical consequence of this delay-aware structure is the induced coupling between the coordination latency and the effective planning horizon. Let T_f denote the fixed terminal time. Due to the delay incurred during sequential updates, the remaining prediction horizon for the next subproblem, N_{rem}, is effectively constrained by the coordination time:

$$N_{\text{rem}} = \frac{T_f - (t + K\Delta t)}{\Delta t} \tag{15}$$

where t denotes the current time. As K increases, the coordination cycle $K\Delta t$ occupies a larger portion of the remaining mission time, leading to a more pronounced reduction in the dimensionality of the local optimization problems. This shrinking-horizon property partially mitigates the computational growth associated with agent scaling, as theoretically analyzed in Section III-D.

Moreover, the repeated coordination cycles allow interaction information among agents to accumulate iteratively, improving the consistency of the predicted trajectories through a process analogous to warm-starting. The feedback nature of the receding-horizon framework enhances robustness against modeling errors and external disturbances, continuously refining the control trajectories based on the most recent system state and neighboring information.

D. COMPUTATIONAL COMPLEXITY ANALYSIS

The computational complexity of the proposed CCP-PSM framework is analyzed to evaluate its scalability for multi-agent systems. The complexity of a single local subproblem for agent k at a given coordination cycle depends on the number of decision variables, which is determined by the remaining prediction horizon N_{rem} defined in Section III-C.

1) Local Subproblem Complexity

The local optimization involves nested loops: an outer CCP loop and an inner PSM loop. Given the fixed maximum iterations N_{CCP} and N_{PSM}, the computational effort within each coordination cycle is primarily characterized by two primary operations:

- *Gradient Computation:* Evaluating the subgradient of the collision penalty requires computing relative distances to $K-1$ neighboring agents over N_{rem} timesteps. This operation scales as $\mathcal{O}((K-1)\cdot N_{\text{rem}})$, or $\mathcal{O}(K \cdot N_{\text{rem}})$.
- *Projection onto Linear Constraints:* The projection step (13) is performed onto an affine subspace defined by the system dynamics. Since the projection matrix is

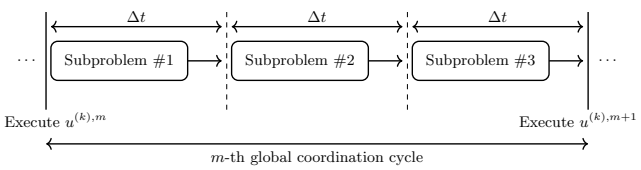


FIGURE 2. Global coordination cycle timeline ($K = 3$).

pre-computed, the operation reduces to matrix–vector multiplications. As the dimension of the optimization variable $u^{(k)}$ is $2N_{\mathrm{rem}}$, the projection complexity scales quadratically with the horizon length, i.e., $\mathcal{O}(N_{\mathrm{rem}}^2)$.

Therefore, the computational complexity of solving a single local subproblem is given by

$$\mathcal{O}\big(N_{\mathrm{CCP}}N_{\mathrm{PSM}}(KN_{\mathrm{rem}}+N_{\mathrm{rem}}^2)\big).$$

2) Global Coordination Cycle Complexity

A full coordination cycle involves sequential updates for all K agents. Thus, the total complexity of one global coordination cycle is:

$$\mathcal{O}\big(N_{\mathrm{CCP}}N_{\mathrm{PSM}}(K^2N_{\mathrm{rem}}+KN_{\mathrm{rem}}^2)\big),$$

where the term K^2N_{rem} arises from pairwise interaction evaluations across all agents, and KN_{rem}^2 corresponds to the cumulative projection effort.

3) Impact of the Shrinking Horizon

A critical feature of the proposed framework is the structural coupling between the number of agents K and the effective planning horizon N_{rem}. As discussed in Section III-C, each coordination cycle incurs a delay proportional to $K\Delta t$. Substituting (15) into the complexity formulation, the remaining prediction horizon effectively decreases with K:

$$N_{\mathrm{rem}} \propto T_{\mathrm{total}} - K.$$

This coupling introduces a computational trade-off that explains the non-monotonic trend observed in our numerical experiments:

- *Small K:* The coordination delay is minimal, leaving a large N_{rem}. The total cost is dominated by the KN_{rem}^2 term, as agents must optimize trajectories over a long remaining planning horizon.
- *Medium K:* The reduction in N_{rem} significantly decreases the dimensionality of the projection operations. Due to the quadratic dependence on N_{rem}, this horizon-shrinking effect outweighs the linear growth in K, resulting in an overall reduction in latency.
- *Large K:* As N_{rem} reaches a lower bound (or a minimum required horizon), the K^2N_{rem} interaction term becomes dominant, leading to an increase in total computational cost.

IV. NUMERICAL SIMULATIONS

A. SIMULATION SETUP

Numerical simulations are conducted to validate the effectiveness of the proposed agent-wise CCP-PSM algorithm under the receding-horizon coordination framework described in Section III-C. The evaluation is structured to demonstrate not only the basic avoidance capability but also the scalability and robustness required for real-time multi-agent systems. The specific configurations for the scenarios, baseline methods, and implementation details are as follows:

1) Scenario Descriptions:

Two distinct scenarios are designed to test different aspects of the algorithm under the aforementioned framework:

- *Circle Swap Scenario:*
  K agents are equidistantly placed on a circle with a radius of 50 m and tasked with moving to their respective antipodal points. This scenario is a representative benchmark for evaluating the algorithm's ability to resolve highly symmetrical conflicts at a single central junction, where the risk of livelock or deadlock is highest.
- *Dense Crossing Scenario:*
  To evaluate the algorithm across a broad range of encounter geometries, K agents are assigned random start and target locations within a confined square area. To avoid initial separation violations, the start and target positions are randomly sampled from a grid with intervals of d_{safety}. This scenario serves as a Monte Carlo benchmark to assess how the algorithm handles complex and unstructured interaction patterns that arise in increasingly congested environments.

2) Baseline Algorithms:

For a comprehensive evaluation, the following two methods are implemented as baselines, representing the reactive and optimization-based paradigms, respectively:

- *ORCA:*
  The Optimal Reciprocal Collision Avoidance (ORCA) framework is adopted as a reactive baseline. At each time step, a preferred velocity $v_{t,\mathrm{pref}}^{(k)}$ is defined for each agent k toward its goal $p_{\mathrm{des}}^{(k)}$, incorporating a rotation matrix $R(\theta)$ for symmetry breaking:

$$v_{t,\mathrm{pref}}^{(k)} = R(\theta)\frac{p_{t,\mathrm{go}}^{(k)}}{\|p_{t,\mathrm{go}}^{(k)}\|}\min\left(v_{\max}, \frac{\|p_{t,\mathrm{go}}^{(k)}\|}{\tau}\right)$$

  where $p_{t,\mathrm{go}}^{(k)} = p_{\mathrm{des}}^{(k)} - p_t^{(k)}$ and τ denotes the collision-avoidance time horizon. Pairwise collision avoidance is enforced through linear half-plane constraints. For each pair (k,l), we impose:

$$n_{kl,t}^{\top}\left(v - v_{kl,t}^{\mathrm{ORCA}}\right) \geq 0$$

  Here, $u_{kl,t}$ is the minimum velocity adjustment to reach the boundary of the relative velocity obstacle, and $v_{kl,t}^{\mathrm{ORCA}} = v_t^{(k)} + 0.5u_{kl,t}$ ensures reciprocal adaptation. Unlike solver-based optimization, the feasible velocity $v_{t,*}^{(k)}$ is selected from a discrete candidate set $\mathcal{C}_t^{(k)}$. This set includes (i) $v_{t,\mathrm{pref}}^{(k)}$, (ii) its projections onto ORCA line boundaries, and (iii) all relevant intersections among constraints and the speed limit $\|v\| = v_{\max}$. The final control input is derived via $u_t^{(k)} = (v_{t,*}^{(k)} - v_t^{(k)})/\Delta t$.
- *Centralized SCP:*
  Sequential Convex Programming (SCP) is adopted as the optimization-based baseline because it is a standard approach for handling non-convex collision avoidance

by iteratively solving convex subproblems. In this study, SCP solves for all agents' trajectories in a single centralized framework, providing a performance reference. The non-convex safety constraint (1f) is linearized using a first-order Taylor approximation around the reference trajectory $(\bar{p}_t^{(k)}, \bar{p}_t^{(l)})$ from the previous iteration. The resulting convex subproblem at each iteration i is formulated as:

$$
\begin{aligned}
\text{minimize} \quad & \sum_{t=0}^{T-1}\sum_{k=1}^{K} \|u_t^{(k)}\|^2 \\
& + w_{\text{tr}}^{[i]} \sum_{t=0}^{T}\sum_{k=1}^{K} \|p_t^{(k)} - \bar{p}_t^{(k)}\|^2 \\
\text{subject to} \quad & p_{t+1}^{(k)} = p_t^{(k)} + \Delta t v_t^{(k)}, \\
& v_{t+1}^{(k)} = v_t^{(k)} + \Delta t u_t^{(k)}, \\
& p_0^{(k)} = p_{\text{init}}^{(k)}, \qquad v_0^{(k)} = v_{\text{init}}^{(k)}, \\
& p_T^{(k)} = p_{\text{des}}^{(k)}, \qquad v_T^{(k)} = v_{\text{des}}^{(k)}, \\
& 2(\bar{p}_t^{(k)} - \bar{p}_t^{(l)})^\top (p_t^{(k)} - p_t^{(l)}) \\
& \quad \geq d_{\text{safety}}^2 + \|\bar{p}_t^{(k)} - \bar{p}_t^{(l)}\|^2, \quad \forall l \neq k.
\end{aligned}
$$

To ensure the validity of the linearization and facilitate stable convergence, a trust-region term is incorporated into the objective function. The trust-region weight $w_{\text{tr}}^{[i]}$ is halved at each successive SCP iteration, progressively relaxing the penalty on deviations from the reference trajectory as the solution converges. The algorithm terminates when the ℓ_2-norm of the state changes falls below a predefined threshold.

3) Implementation Details:

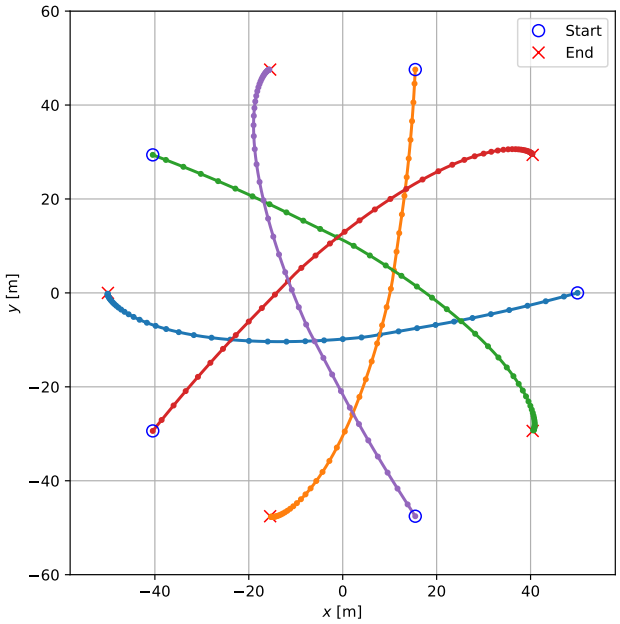
All simulations are implemented in Python 3.11 and executed on a desktop system equipped with an Intel Core i5 processor and 32 GB DDR4 RAM. For all experiments, a fully connected communication topology is assumed, where each agent has instantaneous access to the states of all other agents at each time step. The key simulation parameters, including the solver settings and algorithm-specific constants, are summarized in Table 1.

B. COMPARATIVE PERFORMANCE ANALYSIS

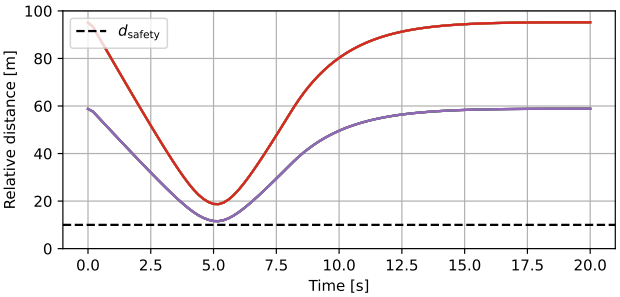
1) Deterministic Analysis: Circle Swap Scenario

Figs. 3 and 4 illustrate the collision avoidance results of the baseline algorithms for $K = 5$. The circle swap scenario presents a significant challenge due to its highly symmetric geometry, where all agents' intended paths intersect at a single central point, often leading to numerical instability or deadlocks.

As shown in Fig. 3, the reactive ORCA framework successfully resolves the conflict but exhibits relatively conservative detours. While ORCA is computationally efficient, its lack of future trajectory prediction results in less-than-optimal paths in such complex symmetric settings. On the



(a) Trajectories.

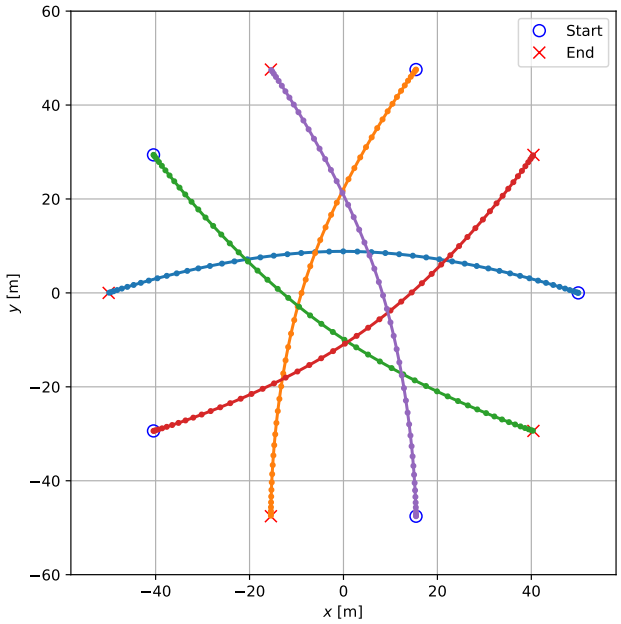


(b) Relative distance.

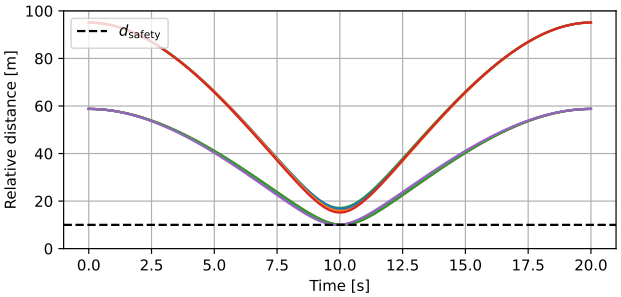
FIGURE 3. ORCA, Circle swap, $K = 5$.

TABLE 1. Simulation parameters.

Parameter	Value	Unit
Horizon length, T	100	–
Time step, Δt	0.2	s
Safe distance, d_{safety}	10	m
Penalty weight, λ	0.9	–
Initial PSM stepsize, α_0	0.5	–
Maximum CCP iterations, N_{CCP}	10	–
Maximum PSM iterations, N_{PSM}	10	–
ORCA time horizon, τ	2.5	s
ORCA speed limit, $v_{\max}$	10	m/s
ORCA bias angle, θ	15.0	deg
SCP solver	ECOS	–
Maximum SCP iterations	30	–
Initial trust-region weight, $w_{\text{tr}}^{[0]}$	1.0	–
SCP tolerance	0.1	–

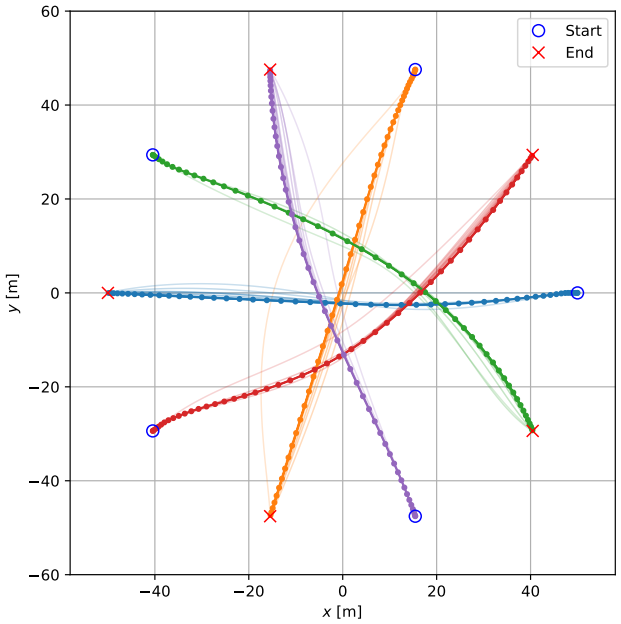


(a) Trajectories.

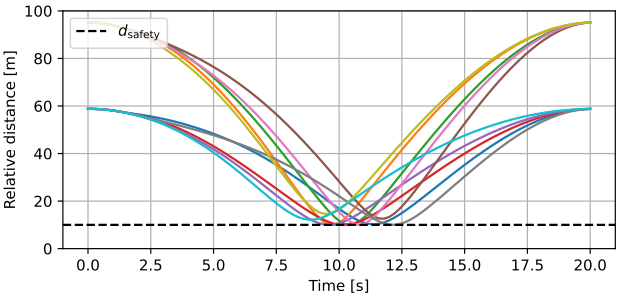


(b) Relative distance.

FIGURE 4. Centralized SCP, Circle swap, $K = 5$.



(a) Trajectories.



(b) Relative distance.

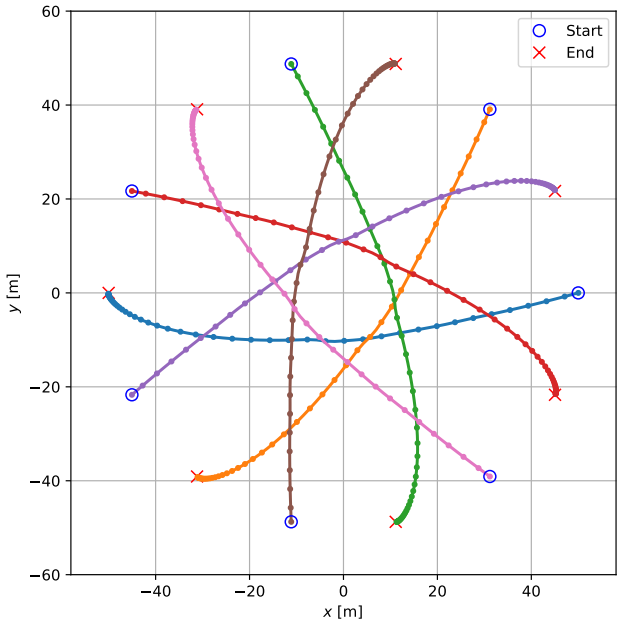
FIGURE 5. CCP-PSM, Circle swap, $K = 5$.

other hand, the centralized SCP (Fig. 4) generates the most efficient trajectories by considering the entire system's state simultaneously. However, this optimality comes at the cost of high computational demand; the centralized SCP requires an average of 26.861 s to complete 20 iterations for a single optimization instance.

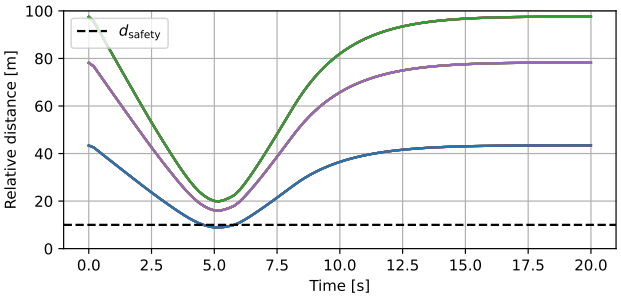
Fig. 5 presents the performance of the proposed receding-horizon CCP-PSM. A distinctive feature in Fig. 5(a) is the inclusion of intermediate solutions, depicted as faint lines, which represent the predicted trajectories generated at each receding-horizon step. This visualization demonstrates the active re-planning process where agents dynamically refine their future paths based on updated information from others.

While the proposed method effectively resolves the symmetric conflict, its resulting trajectories exhibit distinct patterns compared to the centralized SCP. This divergence stems from the sequential coordination process inherent in the distributed Gauss–Seidel update, where agents adapt their paths based on the most recently shared information from others. Despite these differences in path geometry, the proposed method maintains the prescribed minimum separation in this case. As depicted in the relative distance plots (Fig. 5(b)), multiple agent pairs strictly maintain the safety boundary, demonstrating that the algorithm utilizes the available feasible space as efficiently as a centralized solver.

The quantitative comparison is summarized in Table 2. To ensure a fair comparison, the computation time for ORCA and CCP-PSM is measured as the total execution time, while for SCP, it represents the total time required for 20 iterations. The results are organized to highlight the trade-offs between optimality and efficiency. Notably, CCP-PSM achieves a control cost (487.67) that is significantly lower than that of ORCA (13161.05) and maintains the prescribed minimum-separation boundary in this case ($d_{\min} = 10.00$ m), matching the SCP result. The strikingly high control cost of ORCA is primarily attributed to its reactive nature; the conversion of its velocity-derived outputs into acceleration inputs for the double-integrator dynamics leads to impulsive control signals

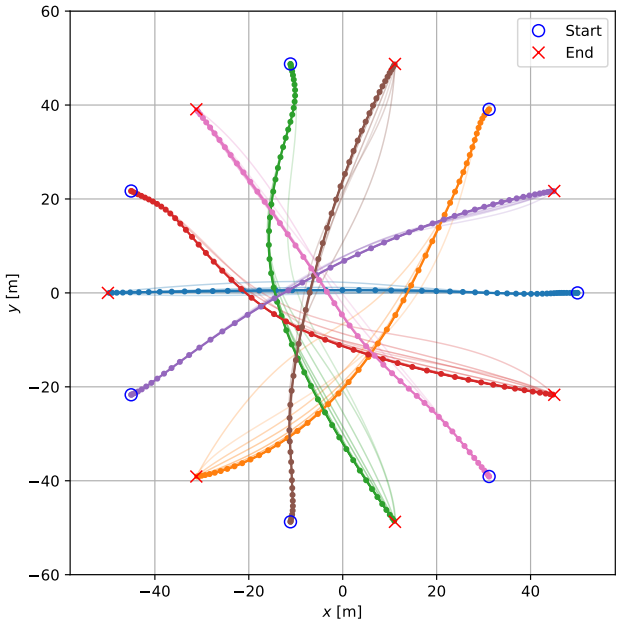


(a) Trajectories.

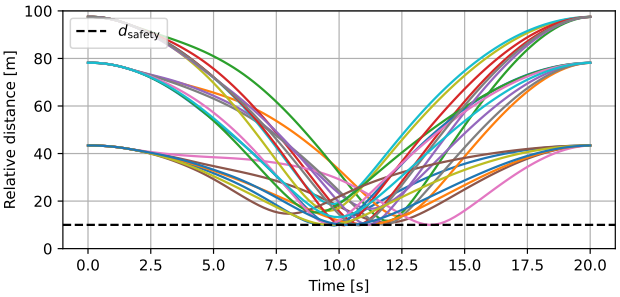


(b) Relative distance.

FIGURE 6. ORCA, Circle swap, $K = 7$.



(a) Trajectories.



(b) Relative distance.

FIGURE 7. CCP-PSM, Circle swap, $K = 7$.

and high-frequency oscillations in dense environments. Conversely, SCP and CCP-PSM proactively plan energy-efficient trajectories by directly incorporating control effort into their optimization processes. Furthermore, the proposed method demonstrates superior real-time feasibility by completing the entire re-planning process in a fraction of the time required by the centralized SCP.

As the number of agents increases to $K = 7$, the performance gap between the algorithms becomes more pronounced, particularly in terms of solver feasibility and minimum-separation performance. A critical finding in this scenario is the total failure of the centralized SCP solver. Specifically, the SCP solver fails at the initial iteration, as it is unable to find even an initial feasible trajectory that satisfies the coupled non-convex safety constraints. This failure is primarily driven by the extreme geometric symmetry of the circle swap, where the linearized safety constraints for all agent pairs become simultaneously active at the central junction. This indicates that as agent density and symmetry increase, the feasible search space for a centralized framework shrinks to the point of numerical intractability, rendering standard optimization-based approaches inapplicable for such complex coordination tasks.

TABLE 2. Quantitative performance comparison ($K = 5$).

Metric	ORCA	SCP	CCP-PSM
Total control cost	13161.05	418.42	487.67
Min. distance (m)	11.57	10.00	10.00
Avg. CPU time (s)	0.090	26.861	1.100
CPU time std. dev.	0.002	0.111	0.008

In contrast, the proposed CCP-PSM and the reactive ORCA framework continue to provide collision-avoidance solutions, as illustrated in Fig. 6 and 7. However, the quality of avoidance maneuvers differs significantly. The trajectories generated by ORCA, while computationally fast, exhibit increased

instability in this highly congested symmetric setup. Most notably, as summarized in Table 3, ORCA fails to maintain the required safety distance, with the minimum relative distance dropping to 8.86 m ($< d_{\text{safety}}$). This violation occurs because the reactive velocity-obstacle approach, with the fixed parameters used in this study, struggles to resolve dense, multi-way conflicts where the feasible velocity space becomes extremely restricted.

The proposed CCP-PSM, integrated with the receding-horizon framework, demonstrates improved robustness in terms of minimum separation and control effort compared with ORCA. Despite the lack of a centralized coordinator, CCP-PSM resolves the $K = 7$ conflict while keeping the minimum distance at 9.70 m. While a minor deviation from the strict $d_{\text{safety}} = 10$ m is observed due to the limited number of iterations for real-time feasibility, it substantially reduces the minimum-separation violation compared with ORCA. Furthermore, the control cost of CCP-PSM (780.77) is orders of magnitude lower than that of ORCA (19613.29), indicating that the proposed method finds far more efficient avoidance paths even in environments where traditional optimization-based solvers fail to initiate. These results indicate that the agent-wise decomposition of CCP-PSM provides a practical compromise between the low latency of reactive methods and the trajectory efficiency of centralized solvers.

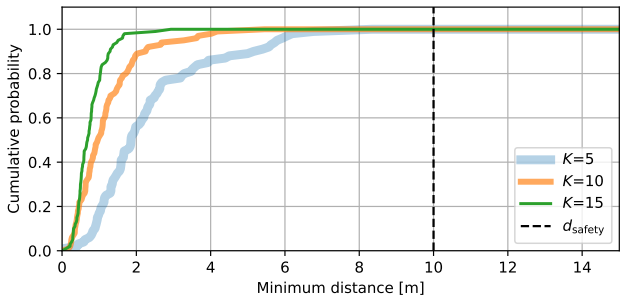
2) Statistical Analysis: Dense Crossing Scenario

To evaluate the statistical performance and scalability of the proposed CCP-PSM framework, 100 Monte Carlo simulations were conducted for each configuration. The agents were assigned to random start and goal locations within square domains of 30×30, 40×40, and 50×50 m^2 for $K = 5, 10,$ and 15, respectively. These settings correspond to theoretical spatial saturation levels of approximately 43.6%, 49.1%, and 58.9%, representing increasingly congested environments.

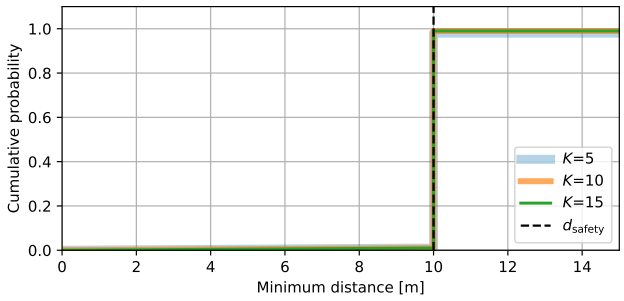
The cumulative distribution function (CDF) of the minimum relative distance in Fig. 8 provides insight into the minimum-separation performance of each method. For a planner that strictly satisfies the prescribed minimum-separation requirement, the CDF would ideally remain zero below $d_{\text{safety}} = 10$ m and increase rapidly near that threshold. ORCA exhibits poor minimum-separation performance across all tested scenarios, with the minimum distance frequently falling below the safety threshold. While its reactive nature and dynamics mismatch contribute to this degradation

TABLE 3. Quantitative performance comparison ($K = 7$).

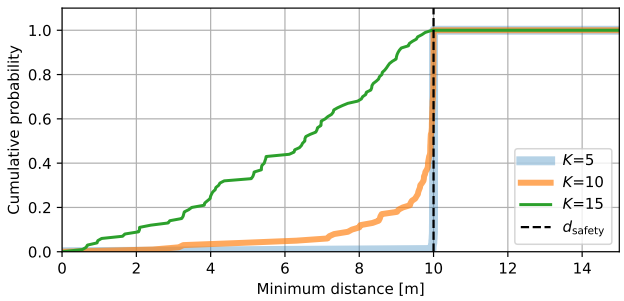
Metric	ORCA	CCP-PSM
Total control cost	19613.29	780.77
Min. distance (m)	8.86	9.70
Avg. CPU time (s)	0.217	1.172
CPU time std. dev.	0.001	0.006



(a) ORCA.



(b) SCP.



(c) CCP-PSM.

FIGURE 8. Minimum distance cumulative probability.

as discussed in the previous section, the primary cause in this statistical evaluation is the sensitivity to parameter tuning. The ORCA parameters, which were calibrated to resolve the structured conflicts in the circle swap scenario, proved to be ill-suited for the unstructured and stochastic interactions of the dense crossing scenario. This highlights a critical practical limitation of reactive baselines, as they require meticulous, case-specific re-tuning to maintain minimum-separation performance whenever the encounter geometry changes.

Centralized SCP shows a near-ideal CDF shape, reflecting its superior ability to jointly optimize all trajectories. Unlike the circle swap scenario, SCP maintains feasibility for $K = 10$ in random encounters due to the broken geometric symmetry, which reduces the likelihood of simultaneous activation of conflicting constraints. However, its main limitation lies in computational scalability. As the number of agents increases, the centralized optimization problem becomes increasingly

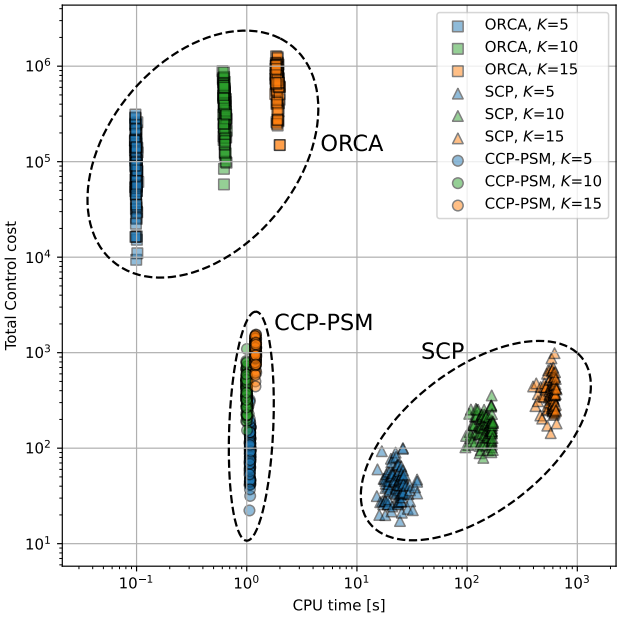
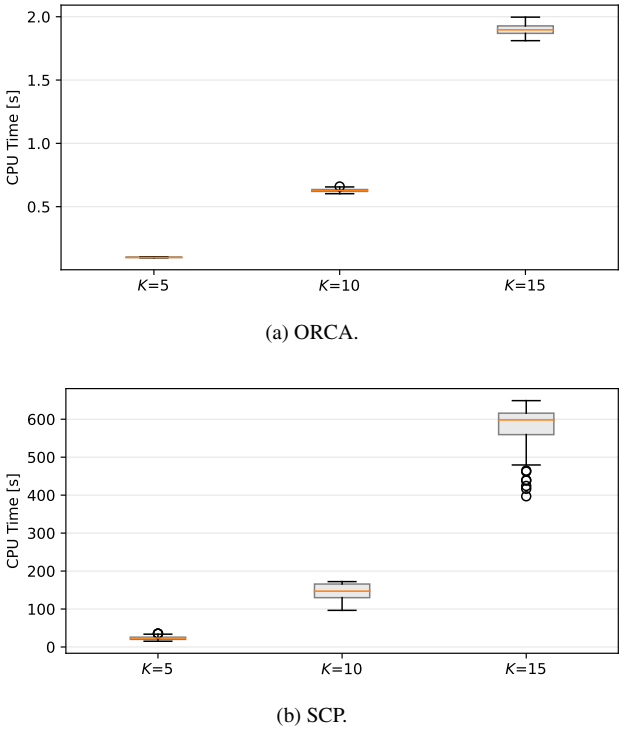


FIGURE 9. **Control cost vs. Computation time trade-off.**



(a) ORCA.

(b) SCP.

(c) CCP-PSM.

FIGURE 10. **Computation time.**

expensive to solve, resulting in a rapid growth of computation time. This makes SCP less suitable for real-time applications in dense multi-agent systems, despite its strong optimality performance.

To ensure mission continuity in such congested environments, the proposed CCP-PSM adopts a soft penalty approach. This design choice provides numerical robustness by allowing the algorithm to generate a "best-effort" solution rather than terminating, even when the prescribed minimum-separation requirement is difficult to satisfy. Under this framework, CCP-PSM achieves near-zero mean minimum-separation violation for $K = 5$, indicating that the observed violations are negligible in magnitude. While separation violations are observed at $K = 10$ and become more pronounced at $K = 15$, these occur in highly congested scenarios where the feasible maneuvering space becomes physically limited by agent density. The observed violations do not lead to numerical divergence or loss of trajectory coherence, but the $K = 15$ case also reveals the limitation of the soft-penalty formulation under extreme density. Nevertheless, the proposed method maintains solution availability and generates dynamically feasible control trajectories across all tested scenarios, without requiring scenario-specific parameter tuning.

TABLE 4. **Minimum-separation violation statistics in dense crossing scenarios.**

Algorithm	K	$\bar{d}_{\min}$ (m)	Violation Rate (%)	Mean Violation (m)
ORCA	5	2.35	100.0	7.65
	10	1.20	100.0	8.80
	15	0.77	100.0	9.23
SCP*	5	10.00	0.0	0.00
	10	10.00	0.0	0.00
	15	10.00	0.0	0.00
CCP-PSM	5	10.00	40.0	0.002
	10	9.34	90.0	0.66
	15	6.06	100.0	3.94

*SCP statistics use successful solver runs only.

To explicitly quantify constraint violations, Table 4 reports minimum-separation violation statistics. The violation rate is defined as the percentage of Monte Carlo trials in which $d_{\min} < d_{\text{safety}}$, and the mean violation is computed as the average of $\max(0, d_{\text{safety}} - d_{\min})$ over all trials. A small numerical tolerance is used when evaluating violations near d_{safety} to avoid counting solver-level round-off errors. Because the agents are modeled as point masses with a prescribed separation requirement rather than finite-size rigid bodies, we report minimum-separation violation statistics instead of a separate physical collision rate.

Fig. 9 illustrates the trade-off between computational latency and control effort, where the algorithms form distinct clusters. ORCA occupies the top-left region, indicating low computational cost but extremely high control effort due to its reactive and jittery maneuvers. SCP is located in the bottom-right, achieving the lowest control cost at the expense of prohibitive computational time. CCP-PSM forms a cluster in the bottom-left, demonstrating that it can generate trajectories with efficiency comparable to SCP while maintaining a computational latency close to that of the reactive ORCA. This positioning indicates that CCP-PSM provides a favorable trade-off for real-time applications where both trajectory efficiency and computational latency are important.

The computational efficiency highlighted in the Pareto analysis is further substantiated by the detailed statistical data in Table 5 and the latency distributions in Fig. 10. While centralized SCP suffers from a rapid escalation in latency as the system scales, the proposed CCP-PSM maintains a remarkably stable and low computational overhead.

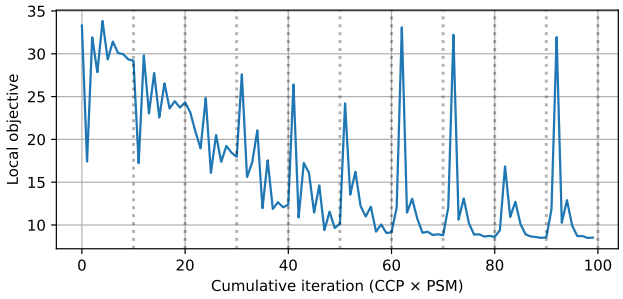
Specifically, CCP-PSM exhibits a unique non-monotonic trend where the average computation time slightly decreases as K increases from 5 to 10, before rising again at $K = 15$. This behavior is a direct empirical validation of the structural trade-off theoretically derived in Section III-D. According to the complexity $\mathcal{O}(K^2 N_{\mathrm{rem}} + K N_{\mathrm{rem}}^2)$, the computational burden is highly sensitive to the horizon length due to its quadratic dependency. In our framework, an increase in K induces a longer coordination delay, which effectively reduces the remaining optimization horizon N_{rem} for each agent.

For the transition from $K = 5$ to $K = 10$, this horizon-shrinking effect dominates the scaling behavior; the reduction in the KN_{rem}^2 term (projection cost) outweighs the linear growth in K, resulting in a net computational speedup. However, as K reaches 15, the quadratic growth of the interaction term $K^2 N_{\mathrm{rem}}$ begins to surpass the benefits of a shorter horizon. Despite this variation, the absolute latency of CCP-PSM remains consistently low with a narrow distribution and minimal outliers, as shown in Fig. 10(c). This high predictability supports the real-time computational reliability of CCP-PSM in dense environments.

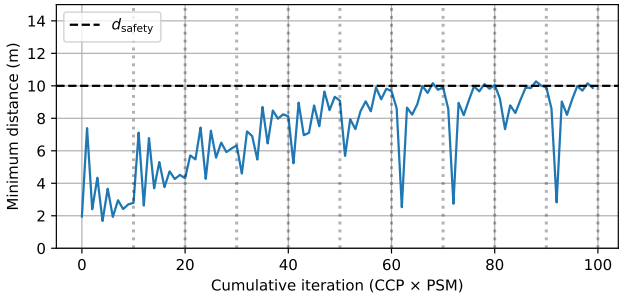
TABLE 5. Computational performance comparison (Dense Crossing scenarios).

Algorithm	K	Avg. CPU (s)	Max. CPU (s)	Std. Dev.
ORCA	5	0.099	0.104	0.001
	10	0.629	0.659	0.012
	15	1.898	1.997	0.042
SCP*	5	23.217	35.241	4.339
	10	144.352	172.196	21.039
	15	582.124	648.965	52.599
CCP-PSM	5	1.071	1.097	0.007
	10	1.001	1.043	0.010
	15	1.200	1.223	0.008

*Excluding instances where the solver failed to find an initial feasible solution.



(a) Local objective value.



(b) Minimum separation distance.

FIGURE 11. Numerical convergence, $K = 3, m = 0, k = 2$.

C. NUMERICAL ASSESSMENT OF CONVERGENCE BEHAVIOR

A formal convergence proof for the coupled CCP–PSM–Gauss–Seidel scheme is analytically challenging due to the inherent nonconvexity of collision constraints, finite inner PSM iterations, and the sequential nature of decentralized updates. Instead, we provide a systematic numerical assessment to demonstrate the convergence characteristics of the proposed solver. For this evaluation, a representative circle-swap scenario with $K = 3$ is selected as a benchmark. This configuration allows us to examine the local numerical behavior without the additional effects of severe spatial saturation that occur in higher-density scenarios.

We specifically focus on the third agent ($k = 2$) during the initial coordination cycle ($m = 0$). Since the sequential update follows the Gauss–Seidel manner, the optimization for $k = 2$ is performed after incorporating the updated trajectory intents of all preceding agents. Thus, this case provides a representative example of how the local solver behaves after receiving the most recently updated trajectories under a high-conflict configuration.

As illustrated in Fig. 11(a), the penalty-augmented objective function exhibits a consistent downward trend over the cumulative iterations. While transient fluctuations are observed within the proximal subproblem (PSM) steps, reflecting the trade-off between the soft penalty and the control effort, the nested CCP–PSM iterations stabilize numerically.

The feasibility recovery is further examined through the minimum separation distance in Fig. 11(b). Despite the ini-

tial guess violating the minimum-separation requirement, the distance progressively increases and stabilizes near the prescribed threshold. This indicates that the soft-penalty formulation reduces minimum-separation violations in a best-effort manner and allows the optimization process to continue from an initially conflicting trajectory. These results support the numerical stability of the local CCP–PSM solver in the tested scenario, while the convergence of the complete distributed receding-horizon framework remains an empirical property rather than a formal guarantee.

D. SCALABILITY ANALYSIS

To validate the scalability characteristics of the proposed CCP-PSM framework, we empirically analyze the computational complexity with respect to both the number of agents K and the remaining prediction horizon length N_{rem}. While the theoretical analysis in Section III-D establishes a composite complexity of $\mathcal{T} = \mathcal{O}(K^2 N_{\mathrm{rem}} + K N_{\mathrm{rem}}^2)$, the actual runtime behavior is influenced by the receding-horizon coordination mechanism, in which N_{rem} is dynamically coupled with the sequential update schedule.

All reported computational times correspond to the execution time of the agent-wise local subproblem solver, which includes both CCP and PSM iterations. To ensure statistical reliability, each configuration is evaluated over 10 independent trials with randomized initialization, and the results are averaged.

Fig. 12 illustrates the relationship between computational time and N_{rem} for different values of K. Across all configurations, the runtime exhibits a consistent quadratic dependence on N_{rem}, confirming that projection onto the affine constraint manifold dominates the computational cost. The fitted curves closely match the empirical measurements, demonstrating strong consistency with the theoretical $\mathcal{O}(K N_{\mathrm{rem}}^2)$ term.

In contrast, Fig. 13 presents the runtime variation with respect to K for fixed representative snapshots of N_{rem}. It is important to note that this plot corresponds to a *slice of the full two-dimensional complexity surface*, rather than an isolated dependency on K. As a result, the observed curvature does not directly reflect a monotonic scaling law in K, but instead arises from the coupling between K and N_{rem} embedded in the receding-horizon coordination structure.

Specifically, the effective horizon available for each agent is indirectly influenced by the coordination delay induced by increasing K, leading to a reduction in N_{rem} in practice. Since the dominant computational term scales quadratically with N_{rem}, this coupling introduces a nonlinear distortion in the projected K-axis response. Consequently, the apparent non-monotonic trend in Fig. 13 should be interpreted as a projection effect of the joint complexity surface, rather than a direct computational speedup with increasing K.

To further quantify the runtime structure, we perform nonlinear regression using the following global model:

$$\mathcal{T} = c_1 K^2 N_{\mathrm{rem}} + c_2 K N_{\mathrm{rem}}^2 + c_3, \tag{17}$$

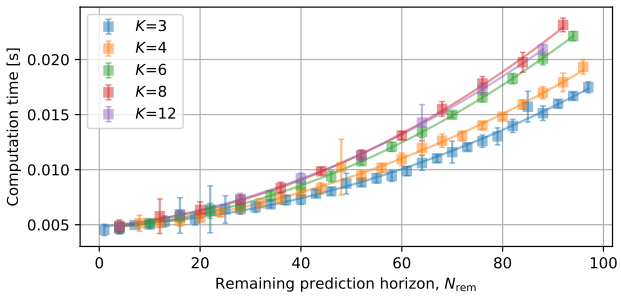


FIGURE 12. Computation time vs. Remaining horizon length N_{rem}.

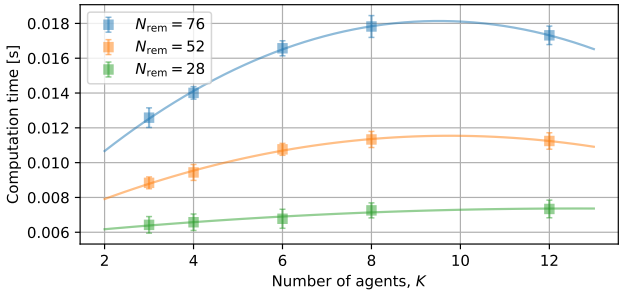


FIGURE 13. Computation time vs. Number of agents K under coupled horizon dynamics.

where c_1, c_2, and c_3 represent the interaction cost, projection cost, and fixed overhead, respectively.

The model achieves a coefficient of determination $R^2 = 0.9088$, indicating strong agreement between the proposed complexity structure and empirical observations. The projection term (c_2) is identified as the dominant contributor to computational cost. The negative value of c_1 is attributed to multicollinearity between the regression features under strongly coupled sampling of (K, N_{rem}); therefore, the fitted coefficients should be interpreted primarily as an empirical regression model rather than as independent physical cost components.

Overall, the empirical results support the proposed complexity model as a useful characterization of the observed scalability trend. The runtime behavior should be interpreted as an emergent property of the coupled (K, N_{rem}) dynamics, confirming that the proposed method remains computationally tractable in large-scale multi-agent settings.

TABLE 6. Fitted complexity model and goodness-of-fit analysis.

Component	Value
Interaction coefficient c_1 ($K^2 N_{\mathrm{rem}}$)	-1.14×10^{-6}
Projection coefficient c_2 ($K N_{\mathrm{rem}}^2$)	3.50×10^{-7}
Fixed overhead c_3	6.84×10^{-3}
R^2	0.9088

E. SENSITIVITY ANALYSIS

To evaluate the influence of key design parameters, we conducted a sensitivity analysis using 100 Monte Carlo simulations for each parameter configuration. The simulations were performed under the $K = 10$ dense crossing scenario, with the remaining parameters fixed as listed in Table 1. The analysis focuses on the mission horizon T and the penalty weight λ, which directly affect temporal flexibility, control effort, and enforcement of the prescribed minimum-separation requirement. The statistical results are summarized in Table 7.

1) Impact of Horizon Length T

The mission horizon T determines the available temporal window for resolving inter-agent conflicts, as shown in Fig. 14. As T increases, agents are provided with greater flexibility to resolve conflicts in a smoother and more anticipatory manner. This generally improves minimum-separation performance, reflected in higher minimum separation distances, and reduces control effort by avoiding impulsive, last-minute maneuvers. For example, the mean violation decreases from 6.711 m at $T = 50$ to 0.152 m at $T = 150$, showing that longer horizons improve minimum-separation performance in the tested dense scenario.

In contrast, shorter horizons ($T \leq 75$) restrict the available maneuvering time, leading to more aggressive responses and greater sensitivity to initial configurations. As T increases beyond a moderate range ($T \geq 125$), the improvement becomes less pronounced, indicating that the benefit of additional prediction horizon saturates once sufficient temporal flexibility is available for conflict resolution.

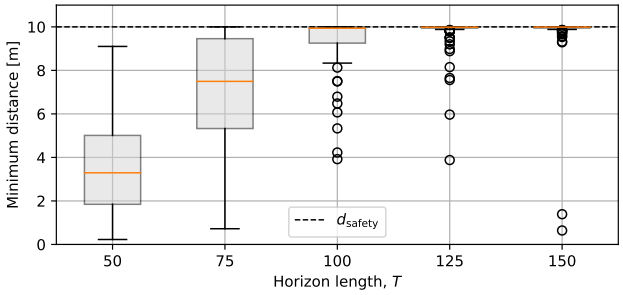
These results suggest that T should be selected large enough to provide sufficient coordination time, but excessively long horizons may provide diminishing returns while increasing computational burden.

2) Impact of Penalty Weight λ

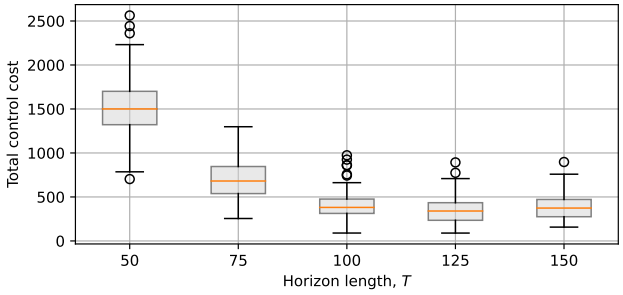
The penalty weight λ controls the trade-off between control effort minimization and enforcement of the prescribed minimum-separation requirement in the soft-penalty formulation. As shown in Fig. 15, smaller values of λ reduce the emphasis on separation enforcement, which can lead to larger variability in the minimum separation distance and larger minimum-separation violations. In contrast, larger values of λ place greater weight on reducing separation violations

TABLE 7. Sensitivity analysis of minimum-separation performance.

Param.	Value	$\bar{d}_{\min}$ (m)	Violation Rate (%)	Mean Violation (m)
T	50	3.289	100.0	6.711
	75	7.832	98.0	2.168
	100	9.039	92.0	0.961
	125	9.724	77.0	0.276
	150	9.848	76.0	0.152
λ	0.50	7.788	100.0	2.212
	0.70	8.641	100.0	1.359
	0.90	9.343	93.0	0.657
	0.95	9.367	74.0	0.633
	0.99	9.567	50.0	0.433

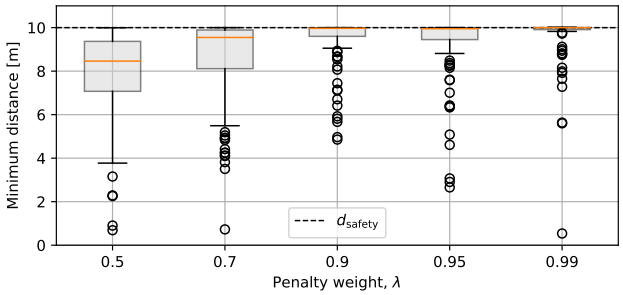


(a) Minimum separation distance distribution.

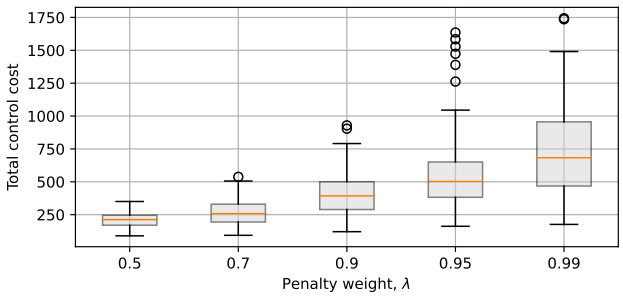


(b) Total control cost distribution.

FIGURE 14. Sensitivity analysis with varying horizon length T.

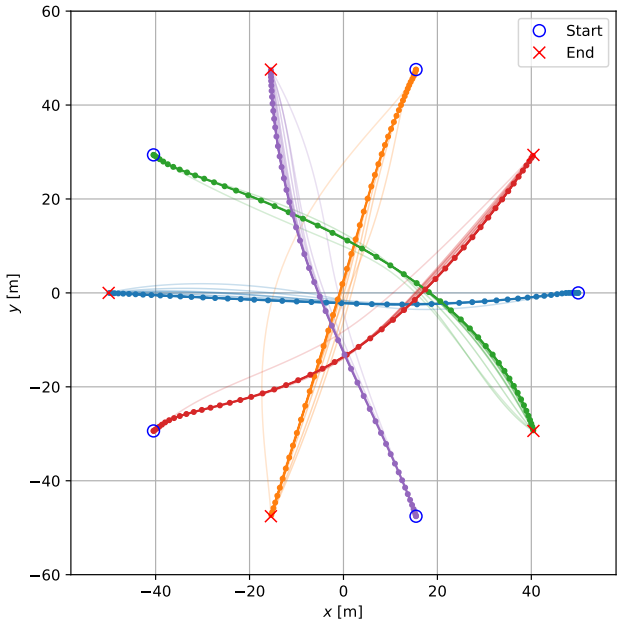


(a) Minimum separation distance distribution.

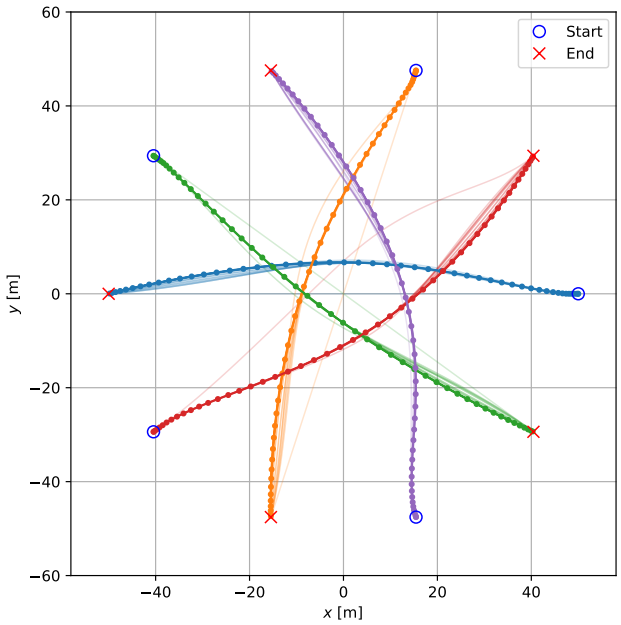


(b) Total control cost distribution.

FIGURE 15. Sensitivity analysis with varying penalty weight λ.

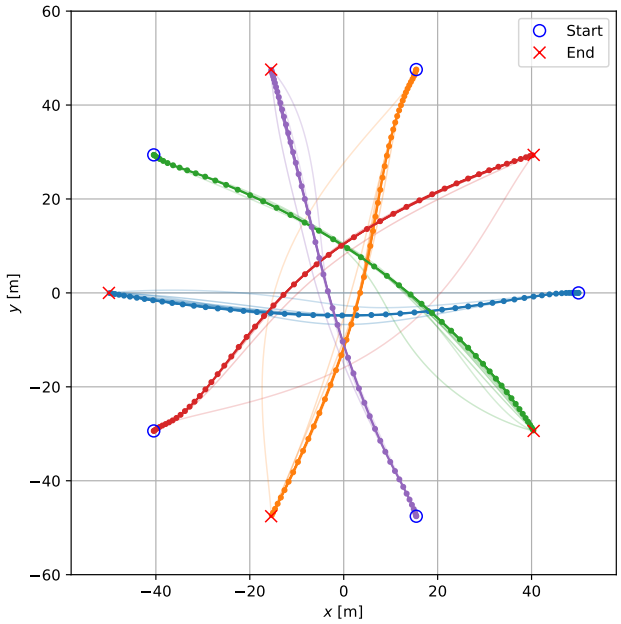


(a) $p_{\text{loss}} = 0.1$.

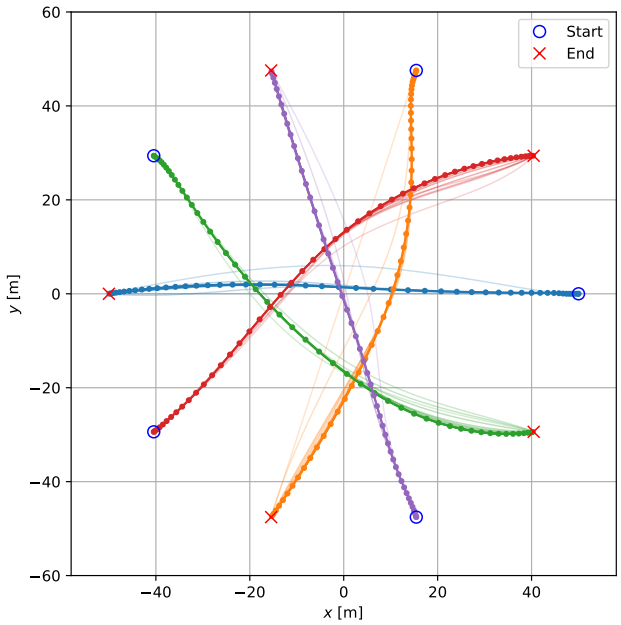


(b) $p_{\text{loss}} = 0.2$.

FIGURE 16. Trajectory example under packet loss conditions, $p_{\text{loss}} = 0.1, 0.2$.



(a) $p_{\text{loss}} = 0.3$.



(b) $p_{\text{loss}} = 0.4$.

FIGURE 17. Trajectory example under packet loss conditions, $p_{\text{loss}} = 0.3, 0.4$.

and generally produce more conservative avoidance behavior, while increasing control effort. This trend is reflected in Table 7, where the mean violation decreases from 2.212 m at $\lambda = 0.50$ to 0.433 m at $\lambda = 0.99$.

Overall, λ serves as a key design parameter that controls the trade-off among control efficiency, minimum-separation enforcement, and trajectory conservativeness. The results indicate that λ should be interpreted as a practical tuning parameter rather than a closed-form optimal quantity. For dense scenarios, larger values of λ are generally preferable when violation reduction is prioritized, whereas smaller values may be used when control efficiency is more important and small separation violations are acceptable. A systematic selection rule based on agent density, desired separation requirement, and interaction intensity remains an important direction for future work.

3) Discussion on Feasibility and Structural Limitations

The results in Table 7 indicate that the proposed CCP-PSM framework may exhibit minimum-separation violations in highly dense configurations, even when dynamically feasible trajectories continue to be generated.

This behavior should be interpreted in terms of the structural differences between centralized and distributed optimization. Centralized approaches such as SCP enforce global consistency within a single optimization problem, enabling strict adherence to coupled separation constraints upon convergence, provided that a feasible solution can be found.

In contrast, the proposed CCP-PSM framework relies on an agent-wise sequential update structure combined with soft penalty relaxation. While this helps maintain numerical robustness and solution availability in challenging scenarios, it may introduce local inconsistency during intermediate coordination steps. Specifically, each agent solves a locally approximated subproblem based on partially updated trajectories of other agents, which may temporarily reduce separation enforcement in highly dense interaction regions.

As a result, separation violations in extreme cases reflect a trade-off between distributed real-time feasibility and strict satisfaction of the global separation requirement. The observed violations do not lead to numerical divergence or loss of trajectory coherence, but they indicate that the soft-penalty formulation alone should not be interpreted as a strict safety guarantee.

Therefore, for strictly safety-critical applications such as UAV systems, additional safety layers such as barrier functions or hard-constraint filters would be required to enforce strict separation guarantees.

F. ROBUSTNESS UNDER COMMUNICATION PACKET LOSS

To evaluate the robustness of the proposed CCP-PSM algorithm, we conducted 100 Monte Carlo simulations for each case under the $K = 5$ circle swap scenario. Packet loss was modeled as an independent Bernoulli trial with a probability $p_{\text{loss}} \in \{0.1, \ldots, 0.5\}$ per communication event. In instances of packet loss, agents utilize the previously computed plans stored in their local buffers to predict the trajectories of others. This effectively introduces a bounded delay in the communication loop, allowing the sequential Gauss–Seidel update to proceed without interruption.

Figures 16 and 17 illustrate the resulting trajectory examples under increasing p_{loss}, while the quantitative performance metrics are summarized in Table 8. The results indicate that the algorithm largely preserves minimum-separation performance despite communication degradation. The average minimum separation distance $d_{\min}$ remained close to the prescribed separation requirement $d_{\text{safety}} = 10$ m, with a value of 9.89 m even at $p_{\text{loss}} = 0.5$.

TABLE 8. Minimum separation distance and control effort under packet loss.

Packet Loss Rate	0.1	0.2	0.3	0.4	0.5
Avg. Min Dist (m)	9.93	9.95	9.91	9.86	9.89
Std. Min Dist	0.14	0.11	0.16	0.39	0.40
Avg. Control Cost	496.76	504.54	515.50	529.99	535.14
Std. Control Cost	17.35	22.21	30.57	40.30	48.57

Although a slight reduction in the minimum separation distance is observed, the deviation remains small, suggesting that the use of previously computed buffered trajectories provides robustness against communication uncertainty. However, a control effort increase is observed as p_{loss} rises, reflecting additional corrective actions required under outdated information. Overall, the proposed framework maintains solution availability and dynamically feasible trajectories even under significant packet loss.

V. CONCLUSION

This paper presents Agent-wise CCP-PSM, a distributed optimization-based collision avoidance algorithm designed for real-time trajectory generation in high-density multi-agent environments. By linearizing non-convex collision avoidance constraints through the Convex-Concave Procedure (CCP) and employing a Projected Subgradient Method (PSM), the framework enables each agent to solve its local subproblem efficiently within a receding-horizon structure.

The primary contributions of the proposed algorithm lie in its structural decomposition and numerical robustness. By adopting a sequential agent-wise update scheme that optimizes local control variables independently, the framework avoids the dimensional explosion inherent in fully centralized formulations. The introduction of a rectifier-based soft penalty improves solution availability, enabling continued trajectory generation even in congested scenarios where traditional hard-constrained methods frequently encounter numerical infeasibility. This formulation provides a controllable trade-off, allowing the system to balance control efficiency and enforcement of the prescribed minimum-separation requirement.

Simulation results demonstrate that the proposed method maintains solution availability and real-time computational feasibility across the tested scenarios. Extensive evaluations confirmed that CCP-PSM achieves a favorable balance between the high optimality of centralized solvers and the low computational latency of reactive baselines. The scalability analysis further revealed that the coupling between coordination latency and the shrinking-horizon property effectively regulates the computational burden as the system scales, as supported by both theoretical complexity analysis and empirical results. In addition, sensitivity analyses indicate that the horizon length and penalty weight affect minimum-separation performance, control effort, and trajectory conservativeness, confirming their role as practical tuning parameters. The framework also demonstrated resilience to communication uncertainties, maintaining minimum-separation performance in most cases by leveraging previously computed buffered trajectories even under degraded communication conditions.

While the present study validates the effectiveness of

the approach through numerical simulations, the achieved computational efficiency and predictable runtime behavior make the method suitable for implementation on resource-constrained embedded platforms. Future work will focus on conducting a formal game-theoretic analysis to theoretically investigate the stability and properties of the Nash equilibrium within the sequential coordination scheme. Additionally, we plan to perform high-fidelity Processor-in-the-Loop Simulations (PILS) and real-time experimental validation using multiple UAV systems to further assess field applicability in unstructured and dynamic environments.

DECLARATION OF GENERATIVE AI AND AI-ASSISTED TECHNOLOGIES IN THE WRITING PROCESS

During the preparation of this work, the authors used Google Gemini 3 to check grammar, spelling, phrasing, and equation formatting. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article.

REFERENCES

[1] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *The international journal of robotics research*, vol. 17, no. 7, pp. 760–772, 1998.
[2] J. Van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *2008 IEEE international conference on robotics and automation*, pp. 1928–1935, Ieee, 2008.
[3] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Robotics research: the 14th international symposium ISRR*, pp. 3–19, Springer, 2011.
[4] J. Snape, J. Van Den Berg, S. J. Guy, and D. Manocha, "The hybrid reciprocal velocity obstacle," *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 696–706, 2011.
[5] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.
[6] S. S. Ge and Y. J. Cui, "New potential functions for mobile robot path planning," *IEEE Transactions on robotics and automation*, vol. 16, no. 5, pp. 615–620, 2002.
[7] J. Schulman, J. Ho, A. X. Lee, I. Awwal, H. Bradlow, and P. Abbeel, "Finding locally optimal, collision-free trajectories with sequential convex optimization.," in *Robotics: science and systems*, vol. 9, pp. 1–10, Berlin, Germany, 2013.
[8] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "Chomp: Gradient optimization techniques for efficient motion planning," in *2009 IEEE international conference on robotics and automation*, pp. 489–494, IEEE, 2009.
[9] M. Mukadam, X. Yan, and B. Boots, "Gaussian process motion planning," in *2016 IEEE international conference on robotics and automation (ICRA)*, pp. 9–15, IEEE, 2016.
[10] Y. Mao, M. Szmuk, and B. Açıkmeşe, "Successive convexification of non-convex optimal control problems and its convergence properties," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 3636–3641, IEEE, 2016.
[11] M. Szmuk and B. Acikmese, "Successive convexification for 6-dof mars rocket powered landing with free-final-time," in *2018 AIAA Guidance, Navigation, and Control Conference*, p. 0617, 2018.
[12] D. Morgan, G. P. Subramanian, S.-J. Chung, and F. Y. Hadaegh, "Swarm assignment and trajectory optimization using variable-swarm, distributed auction assignment and sequential convex programming," *The International Journal of Robotics Research*, vol. 35, p. 1261–1285, Feb. 2016.
[13] J. Bento, N. Derbinsky, J. Alonso-Mora, and J. S. Yedidia, "A message-passing algorithm for multi-agent trajectory planning," *Advances in neural information processing systems*, vol. 26, 2013.
[14] E. Feo-Flushing, L. M. Gambardella, and G. A. D. Caro, "Spatially-distributed missions with heterogeneous multi-robot teams," *IEEE Access*, vol. 9, p. 67327–67348, 2021.
[15] W. Li, S. Yan, L. Shi, J. Yue, M. Shi, B. Lin, and K. Qin, "Multiagent consensus tracking control over asynchronous cooperation–competition networks," *IEEE Transactions on Cybernetics*, 2025.
[16] L. Shi, S. Yan, and W. Li, "Consensus and products of substochastic matrices: Convergence rate with communication delays," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2025.
[17] L. T. H. An and P. D. Tao, "The dc (difference of convex functions) programming and dca revisited with dc models of real world nonconvex optimization problems," *Annals of operations research*, vol. 133, no. 1, pp. 23–46, 2005.
[18] A. L. Yuille and A. Rangarajan, "The concave-convex procedure," *Neural computation*, vol. 15, no. 4, pp. 915–936, 2003.
[19] T. Lipp and S. Boyd, "Variations and extension of the convex–concave procedure," *Optimization and Engineering*, vol. 17, no. 2, pp. 263–287, 2016.
[20] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.
[21] M. H. Vahidnia, A. A. Alesheikh, and S. K. Alavipanah, "A multi-agent architecture for geosimulation of moving agents," *Journal of Geographical Systems*, vol. 17, no. 4, pp. 353–390, 2015.
[22] Y. Guo, Q. Pan, Q. Sun, C. Zhao, D. Wang, and M. Feng, "Cooperative game-based multi-agent path planning with obstacle avoidance," in *2019 IEEE 28th international symposium on industrial electronics (ISIE)*, pp. 1385–1390, IEEE, 2019.
[23] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on automatic control*, vol. 54, no. 1, pp. 48–61, 2009.

GYUBIN PARK (Student member, IEEE) received his B.S. degree in Electronic Engineering from Kyung Hee University in 2020. Since 2021, he has been pursuing his Ph.D. degree in the Department of Aerospace Engineering at Inha University. His research interests include convex optimization and aerospace control.

DOHOON LEE (Student member, IEEE) received his B.S. degree in Aerospace Engineering from Inha University in 2025. He is currently pursuing his Ph.D. degree in the Department of Aerospace Engineering at Inha University since 2025. His research interests include convex optimization algorithms and applications to aerospace systems.

JONG-HAN KIM (Member, IEEE) received the B.S. and M.S. degrees in aerospace engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 1999 and 2001, respectively, and the Ph.D. degree in aeronautics and astronautics from Stanford University, Stanford, CA, USA, in 2012.

He is an associate professor in the Department of Aerospace Engineering at Inha University, Incheon, South Korea. Previously he was an assistant professor in the Department of Electronic Engineering at Kyung Hee University, Yongin, South Korea, and prior to that he was a senior researcher at the Agency for Defense Development (ADD), Daejeon, South Korea, being in charge of developing guidance and control techniques for missile systems development programs. His research interests include advanced optimization techniques, inference and learning, and aerospace applications of advanced guidance and control techniques.