

QP Chaser: Polynomial Trajectory Generation for Autonomous Aerial Tracking

Yunwoo Lee¹, Jungwon Park², Seungwoo Jung³, Boseong Jeon⁴, Dahyun Oh⁵, and H. Jin Kim⁶

Abstract—Maintaining the visibility of the target is one of the major objectives of aerial tracking missions. This paper proposes a target-visible trajectory planning pipeline using quadratic programming (QP). Our approach can handle various tracking settings, including 1) single- and dual-target following and 2) both static and dynamic environments, unlike other works that focus on a single specific setup. In contrast to other studies that fully trust the predicted trajectory of the target and consider only the visibility of the target's center, our pipeline considers error in target path prediction and the entire body of the target to maintain the target visibility robustly. First, a prediction module uses a sample-check strategy to quickly calculate the reachable areas of moving objects, which represent the areas their bodies can reach, considering obstacles. Subsequently, the planning module formulates a single QP problem, considering path homotopy, to generate a tracking trajectory that maximizes the visibility of the target's reachable area among obstacles. The performance of the planner is validated in multiple scenarios, through high-fidelity simulations and real-world experiments.

Note to Practitioners—This paper proposes an aerial target tracking framework applicable to both single- and dual-target following missions. This paper proposes the prediction of the reachable area of moving objects and the generation of a target-visible trajectory, both of which are computed in real-time. Since the proposed planner considers the possible reach area of moving objects, the generated trajectory of the drone is robust to the prediction inaccuracy in terms of the target visibility. Our system can be utilized in crowded environments with multiple moving objects and extended to multiple-target scenarios. We extensively validate our system through several real-world experiments to show practicality.

Received 11 July 2024; revised 6 November 2024, 22 May 2025, and 1 September 2025; accepted 30 October 2025. Date of publication 5 November 2025; date of current version 24 November 2025. This article was recommended for publication by Associate Editor T. Zhang and Editor P. Rocco upon evaluation of the reviewers' comments. This work was supported by the 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 Information and Communication Technology (ICT) under Grant NRF-2020M3C1C1A010864. (*Corresponding author: H. Jin Kim.*)

Yunwoo Lee was with the Artificial Intelligence Institute, Seoul National University, Seoul 08826, South Korea. He is now with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: yunwool@andrew.cmu.edu).

Jungwon Park is with the Department of Mechanical System Design Engineering, Seoul National University of Science and Technology, Seoul 01181, South Korea (e-mail: jungwonpark@seoultech.ac.kr).

Seungwoo Jung, Dahyun Oh, and H. Jin Kim are with the Department of Aerospace Engineering and Automation and Systems Research Institute, Seoul National University, Seoul 08826, South Korea (e-mail: tmddn833@snu.ac.kr; qlass33@snu.ac.kr; hjinkim@snu.ac.kr).

Boseong Jeon is with the Samsung Research, Samsung Electronics, Seoul 06765, South Korea (e-mail: junbs95@gmail.com).

This article has supplementary downloadable material available at <https://doi.org/10.1109/TASE.2025.3629046>, provided by the authors.

Digital Object Identifier 10.1109/TASE.2025.3629046

Index Terms—Aerial tracking, visual servoing, trajectory planning, vision-based autonomous aerial vehicles.

I. INTRODUCTION

VISION-AIDED multi-rotors [1], [2], [3] are widely employed in applications such as surveillance [4] and cinematography [5], and autonomous target chasing is essential in such tasks. In target-chasing missions, various situations exist in which a single drone has to handle both single- and multi-target scenarios without occlusion. For example, in film shooting, there are scenes in which one or several actors should be shot in a single take, without being visually disturbed by structures in the shooting set. Moreover, the occlusion of main actors by background actors is generally prohibited. Additionally, autonomous shooting can be used in sporting events to provide aerial views of athletes' dynamic play. Similarly, the athletes' movement should not be obstructed by crowds. Therefore, a tracking strategy capable of handling both single and multiple targets among static and dynamic obstacles can benefit various scenarios in chasing tasks.

Despite great attention and research over the recent decade, aerial target tracking remains a demanding task for the following reasons. First, the motion generator in the chasing system needs to account for visibility obstruction caused by environmental structures and the limited camera field-of-view (FOV), in addition to typical considerations in UAV motion planning, such as collision avoidance, quality of the flight path, and dynamical limits. Since the sudden appearance of unforeseen obstacles can cause target occlusion, a trajectory that satisfies all these considerations needs to be generated quickly. Second, it is difficult to forecast accurate future paths of dynamic objects due to perceptual error from sensors, surrounding environmental structures, and unreliable estimation of intentions. Poor predictions can negatively impact trajectory planning performance and lead to tracking failures.

In order to address the factors above, we propose a real-time single- and dual-target tracking strategy that 1) can be adopted in both static and dynamic environments and 2) enhances the target visibility against prediction error, as illustrated in Fig. 1. The proposed method consists of two parts: *prediction* and *chasing*. In the *prediction* module, we calculate reachable areas of moving objects, taking into account the obstacle configuration. For efficient calculations, we use a sample-check strategy. First, we sample motion primitives representing possible trajectories, generated by quadratic programming (QP), which is solved in closed form. Then, collisions between primitives and obstacles are checked while leveraging the properties of Bernstein polynomials. We then calculate reachable areas enclosing the collision-free primitives.

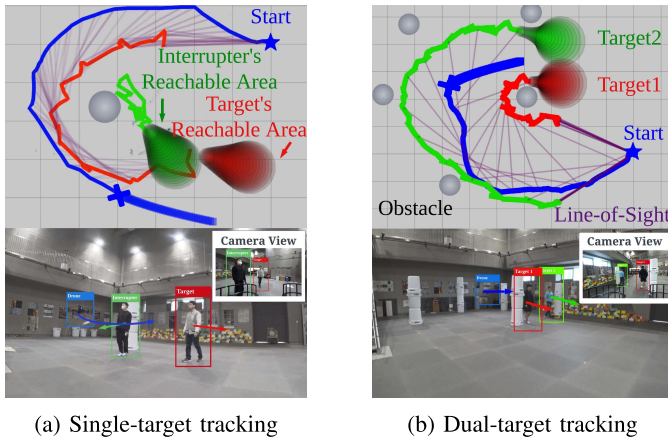


Fig. 1. Target tracking mission in a realistic situation. (a): A target (red) moves in an indoor arena, and a dynamic obstacle (green) interrupts the visibility of the target. (b): Two targets (red and green) move among stacked bins (grey). A chaser drone (cross) generates a trajectory (blue) consistently tracking the targets without occlusion.

In the *chasing* module, we design a chasing trajectory via a single QP. QP is known for being solvable in polynomial time and guaranteeing global optimality. We adopted this method because the optimization was solved within tens of milliseconds during numerous tests. The key idea of our trajectory planning is to define a target-visible region (TVR), a time-dependent spatial set that keeps the visibility of targets. Two essential factors of TVR enhance the target visibility. First, analyzing the homotopy classes of the targets' path helps avoid situations where target occlusion inescapably occurs. Second, TVR is designed to maximize the visible area of the target's reachable area, considering potential visual obstructions from obstacles and ensuring robust target visibility against the prediction error. Moreover, by allowing the entire target body to be viewed instead of specific points, the TVR enhances the target visibility. Additionally, to handle dual-target scenarios, we define an area where two targets can be simultaneously viewed with a limited FOV camera. Table I presents the comparison with the relevant works, and our main contributions are summarized as follows.

- We propose a QP-based trajectory planning framework capable of single and dual target-chasing missions amidst static and dynamic obstacles, in contrast with existing works that address either single-target scenarios or static environments only.
- (*Prediction*) We propose an efficient sample-check-based method to compute reachable areas of moving objects, leveraging Bernstein polynomials. This method reflects both static and dynamic obstacles, which contrasts with other works that do not fully consider the obstacle configurations in motion predictions.
- (*Chasing*) We propose a target-visible trajectory generation method by designing a target-visible region (TVR), in which the target visibility is robustly maintained. This approach considers 1) the path homotopy, 2) the entire body of targets, and 3) the prediction error of moving objects. This contrasts with other works that fully trust potentially erroneous path predictions.

The remainder of this paper is arranged as follows. We review the relevant References in Section II, and the relationship between the target visibility and the path homotopy is studied in Section III. The problem statement and the pipeline of the proposed system are presented in Section IV, and Section V describes the prediction of the reachable areas of moving objects. Section VI describes TVR, designs the reference trajectory for the chasing drone, and completes QP formulation. The validation of the proposed pipeline is demonstrated with high-fidelity simulations and real-world experiments in Section VII.

II. RELATED WORKS

In this section, we review UAV trajectory planning research on target tracking.

A. Single Target Chasing in Empty Space

Research on motion planning to follow a single target in a empty space has been widely studied. References [6], [7], and [8] propose vision-based controllers to prevent the target from moving out of the camera's FOV. References [9] and [10] propose viewpoint planners for high-quality video creation and accurate 2D human pose estimation, respectively. Although all these works improve target perception, they are only applicable in obstacle-free environments, making them unsuitable for real-world scenarios.

B. Single Target Chasing in Static Obstacle Environments

There have been various studies that take the target visibility into account in single-target chasing in static environments.

1) *Chasing Trajectory Planning*: [3] and [5] design cost functions that penalize target occlusion by obstacles, while [18] designs an environmental complexity cost function to adjust the distance between a target and a drone, implicitly reducing the probability of occlusion. These methods combine the functions with several other objective functions related to actuation efficiency and collision avoidance. Such conflicting objectives can yield a sub-optimal solution that compromises tracking motion. Therefore, the visibility of the target is not ensured. In contrast, our approach ensures the target visibility by incorporating it as a hard constraint in the QP problem, which can be solved quickly. On the other hand, there are works that explicitly consider visibility as hard constraints in optimization. Reference [19] designs sector-shaped visible region and [20] designs select the view region among annulus sectors centered around the target. They use their target-visible regions in unconstrained optimization to generate a chasing trajectory. However, these works focus on the visibility of the center of a target, which may result in partial occlusion. In contrast, our method considers the visibility of the entire body of the target and accounts for the prediction error, thus enhancing robustness in target tracking.

2) *Moving Target Prediction*: Also, there are works that handle target path prediction for aerial tracking. References [21] and [22] use past target observations to predict the future trajectory via polynomial regression. On the other hand, [23] propose prediction method leveraging advantages of

Gaussian process regression and gated recurrent unit. However, their methods can generate erroneous results by insufficiently considering surroundings, resulting in path conflicts with obstacles. With inaccurate target path prediction, planners may not produce effective chasing paths and fail to chase a target without occlusion. Some works [11] and [24] predict the future target's movements by formulating optimization problems that encourage the target to move towards free spaces. However, these approaches are limited to static environments. In contrast, our framework predicts the target's movement considering both static and dynamic obstacles.

C. Single-Target Chasing in Dynamic Environments

Some studies treat tracking a target in dynamic environments. There are works that attempt to solve the occlusion problem by approximating the future motion of both dynamic obstacles and targets using a constant velocity model. Reference [12] designs a target-visibility cost that is inspired by GPU ray casting model. Reference [13] uses a learned occlusion model to evaluate the target visibility and tests the planner in dynamic environments. [14] aims to avoid occlusion by imposing a constraint that prevents a segment connecting a target and the drone from colliding with dynamic obstacles. Since [12] and [13] use the constant velocity model to predict the movements of dynamic objects, there are cases where the target and obstacles overlap, leading to incorrect target-visibility cost evaluation, which can ruin the planning results. Furthermore, the constraint in [14] cannot be satisfied if the predicted path of the target traverses occupied space (e.g. obstacles), which makes the optimization problem infeasible. In contrast, our approach predicts the movements of objects by fully considering the environment and incorporates the prediction error into planning to enhance the target visibility.

On the other hand, the approach in [15] makes a set of polynomial motion primitives and selects the best path under safety and target visibility constraints to acquire a non-colliding target path and generate a chasing trajectory. However, when the target and drone are near an obstacle, the planner [15] may select the path toward a region where occlusion is inevitable from the perspective of path homotopy, as will be shown in Section VII-C. In contrast, our method directly applies concepts of the path homotopy to prevent inescapable occlusion.

D. Aerial Tracking of More Than One Target

Aerial tracking of more than one target using a single drone has been studied in a few works. References [25] and [26] minimize the change in the target's position in the camera image; however, they do not consider obstacles. Reference [16] designs a dual visibility score field to handle the visibility of two targets among obstacles and a camera field-of-view (FOV) limit. Because the field is constructed heuristically, the success rate of tracking heavily depends on parameter settings. In contrast, our work handles these issues by applying them as hard constraints that must be satisfied. Reference [17] tracks multiple targets by simultaneously controlling the camera's intrinsic and extrinsic parameters. However, it is limited to

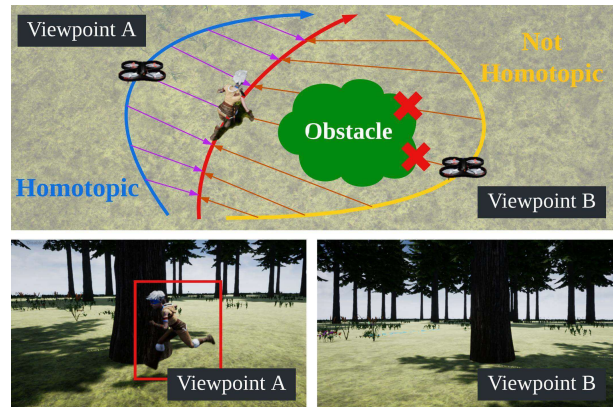


Fig. 2. Comparison between two viewpoints. The target can be observed at Viewpoint A (on a blue path), and the target is occluded by an obstacle at Viewpoint B (on a yellow path). Each arrow (purple, orange) represents the *Line-of-Sight* between the target and the drone. A red cross means target occlusion by an obstacle.

systems where the intrinsic parameters can be controlled. In contrast, we ensure simultaneous observation of the targets using fixed and limited FOV cameras.

E. Target Following Using Quadratic Programming

Few aerial tracking works utilize QP for chasing trajectory planning: [24] and [27]. Reference [24] calculates a series of viewpoints and uses QP optimization to interpolate them, but it does not ensure the target visibility when moving between the viewpoints. Reference [27] generates safe flight corridors along the target's path and pushes the drone's trajectory toward the safe regions, but the target visibility is not considered in its QP problem. In contrast, we formulate the target visibility constraints for the full planning horizon. Also, in conventional drone trajectory generation methods using QP, the trajectory is confined to a static half-space over time intervals. This makes it particularly difficult to keep the target visibility in the presence of dynamic obstacles. In contrast, by incorporating time-varying spatial constraints, our method allows the drone to operate within a broader region, offering an advantage in target tracking.

III. PRELIMINARY

This section presents the relationship between target occlusion and path homotopy. In an obstacle environment, there exist multiple path homotopy classes [28]. As shown in Fig. 2, the visibility of the target is closely related to path homotopy. We analyze the relation between them and take it into consideration when generating the chasing trajectory.

As stated in [29], the definition of path homotopy is as follows.

Definition 1: Paths $\sigma_1, \sigma_2 : I \rightarrow X \subset \mathbb{R}^3$ are path-homotopic if there is a continuous map $F : I \times I \rightarrow X$ such that

$$F(s, 0) = \sigma_1(s) \text{ and } F(s, 1) = \sigma_2(s), \quad (1)$$

where I is unit interval $[0, 1]$.

In this paper, *Line-of-Sight* is of interest, which is defined as follows.

TABLE I
COMPARISON WITH THE STATE-OF-THE-ART ALGORITHMS

Method	Scenarios		Prediction	Planning	
	Dual-target	Dynamic obstacles	Environ. informed	PE-aware	WB of target
[11]	×	×	✓	×	×
[12]	✓	✓	×	×	✓
[13]	×	✓	×	×	×
[14]	×	✓	×	×	×
[15]	×	✓	✓	×	×
[16]	✓	×	✓	×	×
[17]	✓	×	×	×	✓
Ours	✓	✓	✓	✓	✓

✓ means that the algorithm explicitly considers the corresponding item. (PE: Perception-error, and WB: Whole body)

Definition 2: Line-of-Sight is a segment connecting two objects $\mathbf{x}_1, \mathbf{x}_2 : [0, \infty) \rightarrow \mathbb{R}^3$.

$$\mathcal{L}(\mathbf{x}_1(t), \mathbf{x}_2(t)) = (1 - \epsilon)\mathbf{x}_1(t) + \epsilon\mathbf{x}_2(t), \quad \forall \epsilon \in I \quad (2)$$

where t represents time.

Based on the definitions above, we derive a relation between the path homotopy and visibility between two objects.

Theorem 1: When two objects are reciprocally visible, the paths of objects are path-homotopic.

Proof: Suppose that two objects $\mathbf{x}_1(t), \mathbf{x}_2(t)$ move along paths σ_1, σ_2 in free space $\mathcal{F} \subset \mathbb{R}^3$ during time interval $[t_0, t_f]$, respectively. *i.e.* $\sigma_1 = \mathbf{x}_1 \circ \xi, \sigma_2 = \mathbf{x}_2 \circ \xi$, where $\xi : I \rightarrow [t_0, t_f]$ is the time mapping function such that $t = \xi(s)$. If the visibility between \mathbf{x}_1 and \mathbf{x}_2 is maintained, the *Line-of-Sight* does not collide with obstacles: $\mathcal{L}(\mathbf{x}_1(t), \mathbf{x}_2(t)) \subseteq \mathcal{F}$ for $\forall t \in [t_0, t_f]$. Then, by definition, $\mathcal{L}(\sigma_1(s), \sigma_2(s)) = (1 - \epsilon)\sigma_1(s) + \epsilon\sigma_2(s) \in \mathcal{F}$ for $\forall \epsilon, s \in I$. Since such a condition satisfies the definition of continuous mapping in Definition 1, the two paths are homotopic.

From Theorem 1, when the drone chooses a path with a different homotopy class from the target path, target occlusion inevitably occurs. Therefore, we explicitly consider path-homotopy when planning a chasing trajectory.

Throughout this article, we use the notation in Table II. The bold small letters represent vectors, calligraphic capital letters denote sets, and italic lowercase letters mean scalar values.

IV. OVERVIEW

A. Problem Setup

In this section, we formulate the trajectory planning problem for a tracking drone with firmly attached camera sensors that have limited FOVs $\theta_f \in (0, \pi)$ [rad]. We consider an environment \mathcal{W} , which consists of separate cylindrical static and dynamic obstacles. Our goal is to generate a trajectory of the drone so that it can see the single and dual targets ceaselessly in \mathcal{W} over the time horizon $[0, T]$. To achieve the goal, the drone has to predict the future movement of moving objects, such as targets and dynamic obstacles, and generate a target-chasing trajectory. To reflect the prediction error, we compute the reachable areas of moving objects. Then, based on the reachable areas, the planner generates a continuous-time trajectory that satisfies dynamical feasibility and avoids

TABLE II
NOMENCLATURE

Symbol	Definition
$\mathbf{p}_c(t)$	A trajectory of the drone.
$\underline{\mathbf{c}}$	An optimization variable that consists of Bernstein coefficients representing $\mathbf{p}_c(t)$.
n_c	The degree of a polynomial trajectory $\mathbf{p}_c(t)$.
T	Planning horizon
θ_f	FOV of the camera built on the drone.
v_{\max}, a_{\max}	Maximum drone speed and acceleration
\mathcal{F}	Free space
$\mathcal{O}, \mathcal{O}_j$	A set of obstacles. An j -th obstacle in \mathcal{O}
N_o, N_{samp}	The number of elements of \mathcal{O} and samples of end-points in the prediction.
$\mathcal{R}_{q_j}(t), \hat{\mathbf{q}}_j(t), r_{q_j}(t)$	A reachable area of an j -th target. A center trajectory and radius of $\mathcal{R}_{q_j}(t)$. We omit a subscript j to handle an arbitrary target.
$\mathcal{R}_{o_j}(t), \hat{\mathbf{o}}_j(t), r_{o_j}(t)$	A reachable area of an j -th obstacle. A center trajectory and radius of $\mathcal{R}_{o_j}(t)$. We omit a subscript j to handle an arbitrary obstacle.
$S_p, \mathcal{P}_p, \mathcal{P}_s$	<i>End Point Set</i> , a set of candidate trajectories and a set of non-colliding candidate trajectories.
$\mathcal{L}(\mathbf{x}_1, \mathbf{x}_2)$	Line-of-Sight between \mathbf{x}_1 and \mathbf{x}_2 .
$\mathcal{V}_O(t), \mathcal{V}_F(t)$	Target visible region against a single obstacle (TVR-O) and target visible region that can see the two targets simultaneously with camera FOV (TVR-F).
$\mathcal{D}_z(t)$	A region where the drone cannot see both targets with the limited camera FOV.
$\psi(\mathbf{p}_c(t); \hat{\mathbf{q}}(t), \mathcal{O}_i), \psi(\mathbf{p}_c(t); \hat{\mathbf{q}}(t), \mathcal{O})$	Visibility score against the i -th obstacle. Visibility score against all obstacles.
$\mathbf{x}_2 \mathbf{x}_1(t)$	Relative position between \mathbf{x}_1 and \mathbf{x}_2 at time t . $\mathbf{x}_1(t) - \mathbf{x}_2(t)$.
$\ \mathbf{x}\ _p$	L_p norm of \mathbf{x}
$\mathbf{x}'(t), \mathbf{x}''(t), \mathbf{x}'''(t)$	First, second, third time derivatives of $\mathbf{x}(t)$
\mathbf{a}^T, B^T	Transposed vector \mathbf{a} and matrix B
$<, >, \leq, \geq$	elementwise inequality symbols
$\mathbf{x}_{(x)}, \mathbf{x}_{(y)}$	x - and y -components of \mathbf{x} .
$\binom{n}{k}$	Binomial coefficients, the number of k -combinations from a set of n elements.
$\mathbf{0}_{m \times n}$	An $m \times n$ matrix with all-zero elements.
$I_{m \times m}$	An identity matrix with rank m .
$\text{tr}(\cdot), \det(\cdot)$	Trace and determinant of a matrix.
$\partial \mathcal{A}$	A boundary of a closed set \mathcal{A} .
$\mathcal{B}(\mathbf{x}, r)$	A ball with center at \mathbf{x} and radius r .
$[A; B]$	Row-wise concatenation of matrices A and B
$ \mathcal{X} $	The number of elements in set \mathcal{X}
$\overline{AB}, \angle ABC$	Segment connecting points A and B . Angle between \overline{BA} and \overline{BC} .

collision while preserving the target visibility. The whole pipeline is summarized in Fig. 3. Within this pipeline, we set up two problems: 1) *Prediction* and 2) *Chasing problem*.

1) *Prediction Problem:* The prediction module forecasts reachable areas of moving objects such as targets and dynamic obstacles, over a time horizon $t \in [0, T]$. The reachable areas $\mathcal{R}_q(t)$ and $\mathcal{R}_o(t)$ are sets that encompass the future positions of the targets $\mathbf{q}(t)$ and obstacles $\mathbf{o}(t)$, respectively. The goal is to calculate the reachable areas considering an obstacle set \mathcal{O} . Specifically, we represent the j -th target ($j = 1, 2$) and k -th obstacle ($k \in \{1, \dots, N_o := |\mathcal{O}|\}$) as \mathbf{q}_j and \mathbf{o}_k respectively. Since the methods to compute $\mathcal{R}_o(t)$ and $\mathcal{R}_q(t)$ are equivalent, we use a symbol \mathbf{p} instead of \mathbf{q} and \mathbf{o} to represent information of reachable areas of moving objects, in Section V.

2) *Chasing Problem:* Given the reachable areas of the target and the obstacles obtained by the prediction module,

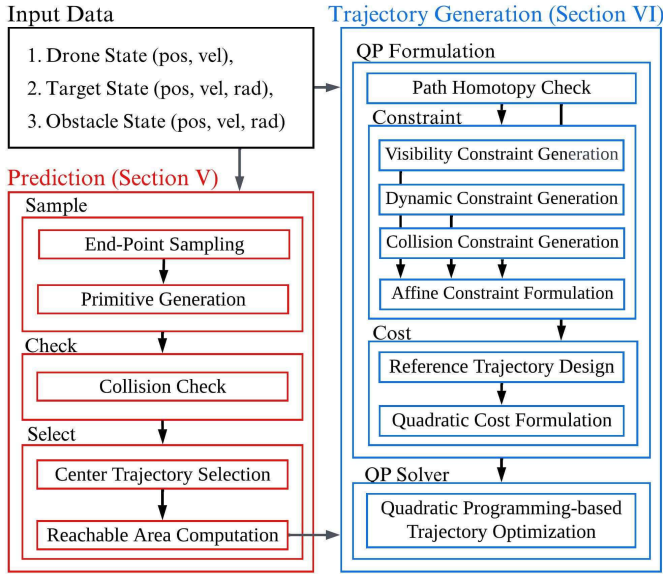


Fig. 3. Pipeline of the proposed chasing system.

we generate a trajectory of the drone $\mathbf{p}_c(t)$ that accounts for the following factors.

- **Occlusion avoidance:** To robustly prevent target occlusion by obstacles, we make the *Lines-of-Sight* between the drone and all points within the target's reachable area $\mathcal{R}_q(t)$ do not intersect with the obstacle's reachable area $\mathcal{R}_o(t)$.

$$\mathcal{L}(\mathbf{p}_c(t), \mathcal{R}_q(t)) \cap \mathcal{R}_o(t) = \emptyset, \quad \forall t \in [0, T] \quad (3)$$

- **Field-of-view:** While tracking two targets, the drone should keep both targets within the limited FOV θ_f of its camera. The angle made by the two *Lines-of-Sight*, $\mathcal{L}(\mathbf{p}_c(t), \mathbf{q}_1(t))$ and $\mathcal{L}(\mathbf{p}_c(t), \mathbf{q}_2(t))$ should not exceed θ_f .

$$\cos^{-1} \left(\frac{\mathbf{p}_c \mathbf{q}_1(t)^\top \mathbf{p}_c \mathbf{q}_2(t)}{\|\mathbf{p}_c \mathbf{q}_1(t)\|_2 \|\mathbf{p}_c \mathbf{q}_2(t)\|_2} \right) \leq \theta_f, \quad \forall t \in [0, T] \quad (4)$$

- **Collision avoidance:** To ensure safety, the drone should not collide with both targets and obstacles.

$$\mathcal{B}(\mathbf{p}_c(t), r_c) \cap \{\mathcal{R}_o(t) \cup \mathcal{R}_q(t)\} = \emptyset, \quad \forall t \in [0, T] \quad (5)$$

r_c is the radius of the drone, and $\mathcal{B}(\cdot, \cdot)$ is the operator representing a ball with a center and radius.

- **Dynamic feasibility:** Considering the drone's actuator limits, the planned trajectory should not exceed the maximum velocity v_{\max} and acceleration a_{\max} .

$$\|\mathbf{p}'_c(t)\|_2 \leq v_{\max}, \quad \|\mathbf{p}''_c(t)\|_2 \leq a_{\max}, \quad \forall t \in [0, T] \quad (6)$$

3) *Assumptions:* For the *Prediction problem*, we assume that (API) the moving objects do not collide with obstacles, and (AP2) they do not move in a jerky way. In the *Chasing problem*, we assume that (AC1) the maximum velocity v_{\max} and the maximum acceleration a_{\max} of the drone are higher than the target and obstacles. Furthermore, based on Theorem 1, when the targets move along a path with different homotopy classes, occlusion unavoidably occurs, so we

assume that (AC2) all targets move along homotopic paths against obstacles.

In addition, we set the flying height of the drone to a fixed level for the acquisition of consistent images of the target. From the problem settings, we focus on the design of chasing trajectory in the $x - y$ plane.

4) *Trajectory Representation:* Due to the virtue of differential flatness of quadrotor dynamics [30], the trajectory of multi-rotors can be expressed with a polynomial function of time t . In this paper, Bernstein basis is employed to express polynomials. Bernstein bases of n -th order polynomial for time interval $[t_a, t_b]$ are defined as follows.

$$b_{k,n}(t; t_a, t_b) = \binom{n}{k} \frac{(t_b - t)^{n-k} (t - t_a)^k}{(t_b - t_a)^n}, \quad 0 \leq k \leq n \quad (7)$$

Since the bases defined above are non-negative in the time interval $[t_a, t_b]$, a linear combination with non-negative coefficients makes the total value become non-negative. We utilize this property in the following sections.

The trajectory of the drone, $\mathbf{p}_c(t) \in \mathbb{R}^2$, is represented as an M -segment piecewise Bernstein polynomial.

$$\mathbf{p}_c(t) = \begin{cases} \mathbf{C}_1^\top \mathbf{b}_{n_c,1}(t) & t \in [T_0, T_1] \\ \mathbf{C}_2^\top \mathbf{b}_{n_c,2}(t) & \text{quadt} \in [T_1, T_2] \\ \dots \\ \mathbf{C}_M^\top \mathbf{b}_{n_c,M}(t) & t \in [T_{M-1}, T_M] \end{cases} \quad (8)$$

$\mathbf{b}_{n_c,m}(t) \in \mathbb{R}^{(n_c+1)}$ is a vector that stacks the Bernstein basis functions $b_{k,n_c}(t; T_{m-1}, T_m)$ vertically for $k = 0$ to n , and $\mathbf{C}_m = [\mathbf{c}_{m(x)}, \mathbf{c}_{m(y)}] \in \mathbb{R}^{(n_c+1) \times 2}$ is a coefficient matrix with $n_c + 1$ control points stacked row-wise, where each column represents the x and y coordinates of the trajectory: $\mathbf{p}_c(t) = [\mathbf{c}_{m(x)}^\top \mathbf{b}_{n_c,m}(t), \mathbf{c}_{m(y)}^\top \mathbf{b}_{n_c,m}(t)]^\top$ for $t \in [T_{m-1}, T_m]$. We define $\mathbf{c}_m = [\mathbf{c}_{m(x)}^\top, \mathbf{c}_{m(y)}^\top]^\top \in \mathbb{R}^{2(n_c+1)}$, and collect them into $\underline{\mathbf{c}} = [\mathbf{c}_1^\top, \dots, \mathbf{c}_M^\top]^\top \in \mathbb{R}^{2M(n_c+1)}$, which serve as the decision variable of the polynomial trajectory optimization.

5) *Objectives:* For the *Prediction Problem*, the prediction module forecasts moving objects' reachable area $\mathcal{R}_p(t)$. Then, for the *Chasing Problem*, the chasing module formulates a QP problem with respect to $\underline{\mathbf{c}}$ and finds an optimal $\underline{\mathbf{c}}$ so that the drone chases the target without occlusion and collision while satisfying dynamical limits.

V. REACHABLE AREA PREDICTION

This section introduces a method for calculating reachable areas using the sample-check strategy. We sample a set of motion primitives and filter out the primitives that collide with obstacles. Then, we calculate a reachable area that encloses the remaining collision-free primitives. Figs. 4a-4c visualize the prediction process.

A. Candidate for Future Trajectory of Moving Object

We sample the motion primitives of the target by collecting the positions that can be reached at $t = T$ and interpolating them with the current position.

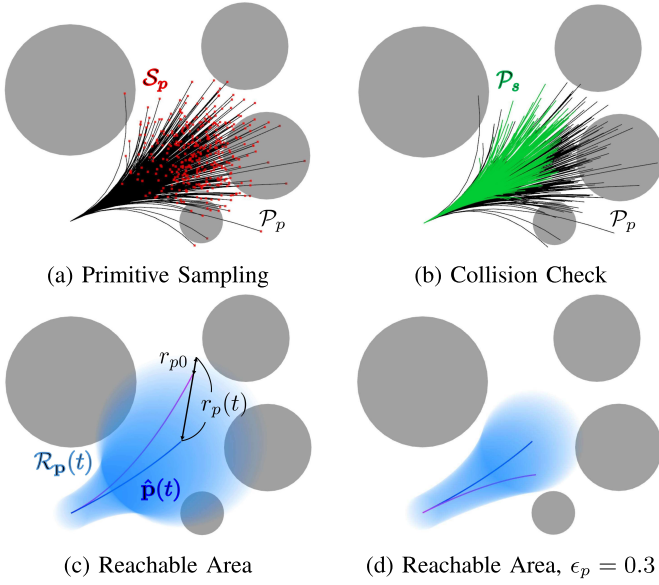


Fig. 4. (a): Sampled endpoints \mathcal{S}_p (red) and primitives \mathcal{P}_p (black) among obstacles (grey). (b): Primitives \mathcal{P}_p (black) and non-colliding primitives \mathcal{P}_s (green). (c): The best primitive $\hat{\mathbf{p}}(t)$ (blue), reachable area $\mathcal{R}_p(t)$ (light blue), and the farthest primitive (magenta). (d): The prediction with the outlier factor $\epsilon_p = 0.3$.

1) *Endpoint Sampling*: The positions of a moving object at time $t = T$ are sampled, using a constant velocity model under disturbance:

$$\begin{aligned} \begin{bmatrix} \mathbf{p}'(t) \\ \mathbf{p}''(t) \end{bmatrix} &= F_p \begin{bmatrix} \mathbf{p}(t) \\ \mathbf{p}'(t) \end{bmatrix} + G_p \mathbf{w}, \quad \mathbf{w} \sim (0, Q), \\ F_p &= \begin{bmatrix} \mathbf{0}_{2 \times 2} & \mathbf{I}_{2 \times 2} \\ \mathbf{0}_{2 \times 2} & \mathbf{0}_{2 \times 2} \end{bmatrix}, \quad G_p = \begin{bmatrix} \mathbf{0}_{2 \times 2} \\ \mathbf{I}_{2 \times 2} \end{bmatrix} \end{aligned} \quad (9)$$

where $\mathbf{p}(t) \in \mathbb{R}^2$ is position of the moving object, and $\mathbf{w} \in \mathbb{R}^2$ is white noise with power spectral density Q . Then, with no measurement update, estimation error covariance $P(t)$ in continuous-time Kalman filter propagates with time [31].

$$P'(t) = F_p(t)P(t) + P(t)F_p^\top(t) + G_p Q G_p^\top \quad (10)$$

We collect N_{samp} points from a 2-dimensional Gaussian distribution $\mathcal{N}(\hat{\mathbf{p}}_0 + \hat{\mathbf{p}}'_0 T, P(T))$ where $\hat{\mathbf{p}}_0$ and $\hat{\mathbf{p}}'_0$ are position and velocity at current time $t = 0$, respectively. It is illustrated in Fig. 4a, where the samples are shown as red points, and we refer to the set of the samples the *Endpoint Set* $\mathcal{S}_p = \{\mathbf{s}_{p,i} \in \mathbb{R}^2 | i = 1, \dots, N_{samp}\}$.

2) *Primitive Generation*: Given the initial position, velocity, and end positions $\mathbf{s}_{p,i} \in \mathcal{S}_p$, we design trajectory candidates $\hat{\mathbf{p}}_i(t)$, $i = 1, \dots, N_{samp}$ for moving objects. Under the assumption (AP2) that moving objects do not move in a jerky way, we establish the following problem.

$$\begin{aligned} \min_{\hat{\mathbf{p}}_i(t)} & \int_0^T \|\hat{\mathbf{p}}_i'''(t)\|_2^2 dt \\ \text{s.t.} & \hat{\mathbf{p}}_i(0) = \hat{\mathbf{p}}_0, \quad \hat{\mathbf{p}}_i'(0) = \hat{\mathbf{p}}'_0, \quad \hat{\mathbf{p}}_i(T) = \mathbf{s}_{p,i} \end{aligned} \quad (11)$$

Recalling that the trajectory is represented with Bernstein polynomial, we write an i -th candidate trajectory as $\hat{\mathbf{p}}_i(t) = \mathbf{P}_i^\top \mathbf{b}_{n_p}(t)$, $i = 1, \dots, N_{samp}$ where $\mathbf{P}_i = [\mathbf{p}_{i(x)}, \mathbf{p}_{i(y)}] \in \mathbb{R}^{(n_p+1) \times 2}$ and $\mathbf{b}_{n_p}(t) = [b_{0,n_p}(t; 0, T), \dots, b_{n_p,n_p}(t; 0, T)]^\top$. By defining

$\underline{\mathbf{p}}_i := [\mathbf{p}_{i(x)}^\top, \mathbf{p}_{i(y)}^\top]^\top$ as an optimization variable, (11) becomes a QP problem

$$\begin{aligned} \min_{\underline{\mathbf{p}}_i} & \frac{1}{2} \underline{\mathbf{p}}_i^\top Q_p \underline{\mathbf{p}}_i \\ \text{s.t.} & A_p \underline{\mathbf{p}}_i = \mathbf{b}_{p,i} \end{aligned} \quad (12)$$

where Q_p is a positive semi-definite matrix, A_p is a $6 \times (n_p + 1)$ matrix, and $\mathbf{b}_{p,i}$ is a vector composed of $\hat{\mathbf{p}}_0$, $\hat{\mathbf{p}}'_0$, and $\mathbf{s}_{p,i}$. (12) can be converted into unconstrained QP, whose optimal $\underline{\mathbf{p}}_i$ is a closed-form solution as follows.

$$\begin{bmatrix} \underline{\mathbf{p}}_i \\ \lambda \end{bmatrix} = \begin{bmatrix} Q_p & A_p^\top \\ A_p & \mathbf{0}_{6 \times 6} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{0}_{2(n_p+1) \times 1} \\ \mathbf{b}_{p,i} \end{bmatrix} \quad (13)$$

where λ is a lagrange multiplier. A set of candidate trajectories of the moving object is defined as \mathcal{P}_p , and Fig. 4a shows an example, where the trajectories are shown as black splines.

B. Collision Check

Under the assumption (API) that moving objects do not collide, trajectory candidates $\hat{\mathbf{p}}_i(t) \in \mathcal{P}_p$ that violate the below condition are filtered out:

$$\|\hat{\mathbf{p}}_i(t) - \hat{\mathbf{o}}_j(t)\|_2^2 - (r_{p0} + r_{o_j}(t))^2 \geq 0, \quad \forall t \in [0, T] \quad (14)$$

where $\hat{\mathbf{o}}_j(t)$ and $r_{o_j}(t)$ denote the predicted trajectory and radius, including the prediction error bounds, of the j -th obstacle, respectively, while r_{p0} represents the radius of the dynamic object of interest. Due to the fact that all terms in (14) can be represented in polynomials, and Bernstein bases are non-negative in the time period $[0, T]$, entirely non-negative coefficients make the left-hand side of (14) non-negative. We examine all coefficients of (14) for the primitives belonging to \mathcal{P}_p . For the mathematical details, see Appendix A. We call a set of primitives that pass the test \mathcal{P}_s . Fig. 4b shows an example, where the green splines are collision-free out of all the black trajectories.

C. Prediction With Error Bounds

With the set \mathcal{P}_s , the reachable area $\mathcal{R}_p(t)$ is defined as a time-varying ball enclosing all the primitives in \mathcal{P}_s .

$$\mathcal{R}_p(t) = \mathcal{B}(\hat{\mathbf{p}}(t), r_p(t)) \quad (15)$$

We determine a center trajectory $\hat{\mathbf{p}}(t)$ as the primitive with the smallest sum of distances to the other primitives in \mathcal{P}_s :

$$\hat{\mathbf{p}}(t) = \operatorname{argmin}_{\hat{\mathbf{p}}_i(t) \in \mathcal{P}_s} \sum_{j \neq i, \hat{\mathbf{p}}_j(t) \in \mathcal{P}_s} \int_0^T \|\hat{\mathbf{p}}_i(t) - \hat{\mathbf{p}}_j(t)\|_2 dt. \quad (16)$$

Proposition 1: The optimization problem (16) is equivalent to the following problem.

$$\hat{\mathbf{p}}(t) = \hat{\mathbf{p}}_{i^*}(t), \quad i^* = \operatorname{argmin}_i \sum_{j=0}^{|\mathcal{P}_p|} \|\mathbf{s}_{p,i} - \mathbf{s}_{p,j}\|_2, \quad (17)$$

where $\mathbf{s}_{p,i} = \hat{\mathbf{p}}_i(T)$, $\mathbf{s}_{p,j} = \hat{\mathbf{p}}_j(T)$, $\hat{\mathbf{p}}_i(t), \hat{\mathbf{p}}_j(t) \in \mathcal{P}_s$

Proof: See Appendix B.

TABLE III
COMPUTATION TIME IN PREDICTION PROCESS

(N_{samp}, N_o)	(500,2)	(500,4)	(2000,2)	(2000,4)
Endpoints Sampling & Primitive Generation [μ s]	43.97	47.23	168.3	184.1
Collision Check [μ s]	146.3	185.3	487.9	750.7
Reachable Area Computation [ms]	0.245	0.172	2.921	2.073
Total Time [ms]	0.435	0.405	3.577	3.008

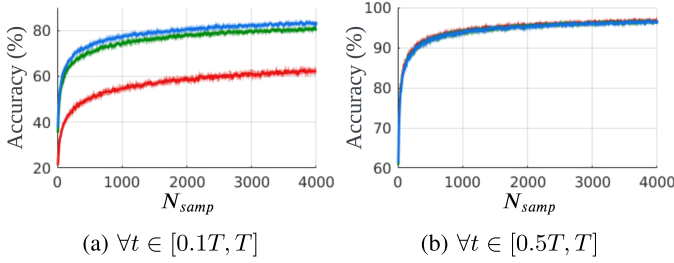


Fig. 5. Prediction accuracy. Red, green, and blue lines represent $Q = 0.1, 0.5, 1.0$ [m^2/s^3] cases, respectively.

From Proposition 1, $\hat{\mathbf{p}}(t)$ is determined by simple arithmetic operations and the process has a time complexity of $O(|\mathcal{P}_s|^2)$. Then, we define $r_p(t)$ so that $\mathcal{R}_p(t)$ encloses all the primitives in \mathcal{P}_s for $\forall t \in [0, T]$.

$$r_p(t) = \max_{\hat{\mathbf{p}}_j(t) \in \mathcal{P}_s} \|\hat{\mathbf{p}}(t) - \hat{\mathbf{p}}_j(t)\|_2 + r_{p0} \quad (18)$$

The second term in the right hand side of (18) allows the whole body to be contained in $\mathcal{R}_p(t)$. Fig. 4c illustrates how $\mathcal{R}_p(t)$, shown as the blue area, is determined. The use of $r_q(t)$ and $r_o(t)$ in *chasing problem* allows for the consideration of the visibility of the entire bodies of the targets, as well as the potential movements of moving targets and dynamic obstacles.

D. Evaluation

We measure the computation time of prediction in obstacle environments. We set n_p as 3 and test 1000 times for different scenarios $(N_{samp}, N_o) = (500, 2), (500, 4), (2000, 2), (2000, 4)$. The prediction module is computed on a laptop with an Intel i7 CPU, with a single thread implementation, and execution time is summarized in Table III. As N_{samp} and N_o increase, the time needed for calculating primitives and collision checks increases. However, environments where obstacles are densely located around moving objects, as shown in Fig. 4b, may result in a small \mathcal{P}_s . Accordingly, the time to compute a reachable area decreases as the number of close obstacles increases. On the other hand, if the object moves while staying far from the obstacles, the computation time for the reachable area increases as the number of obstacles grows.

Also, we test the accuracy of the presented prediction methods. For evaluation, discretized models of (9) with power spectral density of white noise $Q = 0.1, 0.5, 1.0$ [m^2/s^3] are considered. We confirm whether $\mathcal{B}(\mathbf{p}(t), r_{p0}) \subset \mathcal{R}_p(t)$ is satisfied in obstacle-free situations while progressively increasing the N_{samp} . For each N_{samp} , the tests are executed 10000 times, and the acquired accuracy is shown in Fig. 5. Contrary to

the assumption (AP2) that a moving object follows a smooth trajectory, the discretized model used for evaluation can exhibit jerky movements. This indicates that our prediction approach is capable of handling non-ideal target motions when a sufficient number of samples N_{samp} is used. As N_{samp} increases, the accuracy improves, as shown in Fig. 5, but the computation time also increases, as presented in Table III. Therefore, N_{samp} should be determined according to the computation resources for a balance between accuracy and running time. In this paper, we set N_{samp} as 2000. With this setting, $\mathcal{B}(\mathbf{p}(t), r_{p0}) \subset \mathcal{R}_p(t)$, $\forall t \in (0, T]$ was satisfied 98.8% empirically in the simulations in Section VII-C.

As elaborated in Section IV-A, we specifically calculate the reachable areas of moving targets and obstacles: $\mathcal{R}_q(t) = \mathcal{B}(\hat{\mathbf{q}}(t), r_q(t))$ and $\mathcal{R}_o(t) = \mathcal{B}(\hat{\mathbf{o}}(t), r_o(t))$. Subsequently, in the following section, we generate a trajectory that maximizes the visibility of $\mathcal{R}_q(t)$ while avoiding $\mathcal{R}_o(t)$.

VI. CHASING TRAJECTORY GENERATION

This section formulates a QP problem to make a chasing trajectory occlusion-free, collision-free, and dynamically feasible. We first define the target visible region (TVR), the region with maximum visibility of the target's reachable area, and formulate constraints regarding collision avoidance and dynamic limits. Then, we design the reference trajectory to enhance the target visibility. Regarding the target visibility, all the steps are based on the path homotopy.

A. Homotopy Class Check

In 2-dimensional space, there exist two classes of path homotopy against a single obstacle, as shown in Fig. 6a. As stated in Theorem 1, the drone should move along a homotopic path with the target path to avoid occlusion. Based on the relative position between the drone and the obstacle, $\delta \mathbf{p}_c(t) = \mathbf{p}_c(t) - \hat{\mathbf{o}}(t)$, and the relative position between the target and the obstacle, $\delta \hat{\mathbf{q}}(t) = \hat{\mathbf{q}}(t) - \hat{\mathbf{o}}(t)$, at the current time $t = 0$, the homotopy class of chasing path is determined as follows:

$$\begin{cases} \text{Class O1 (if } \det([\delta \mathbf{p}_c^T(t); \delta \hat{\mathbf{q}}^T(t)]) \geq 0, \text{ at } t = 0) \\ \text{Class O2 (if } \det([\delta \mathbf{p}_c^T(t); \delta \hat{\mathbf{q}}^T(t)]) < 0, \text{ at } t = 0) \end{cases} \quad (19)$$

where $[A; B]$ means the row-wise concatenation of matrices A and B . After conducting the homotopy check above, we define the visibility constraint and reference chasing trajectory.

B. Target Visible Region

To robustly maintain the target visibility despite the prediction error, the target visible region TVR is defined considering the reachable areas of the moving objects: $\mathcal{R}_q(t)$ and $\mathcal{R}_o(t)$.

1) *Target Visibility Against Obstacles*: We define the target visible region against an obstacle (TVR-O), for each obstacle. To maximize a visible area of the targets' reachable area that is not occluded by the reachable area of obstacles, TVR-O is set to a half-space so that it includes the target's reachable area and minimizes the area that overlaps with the obstacle's reachable area. There are two cases where the reachable areas of the target and obstacles overlap and do not overlap, and TVR-O is defined accordingly.

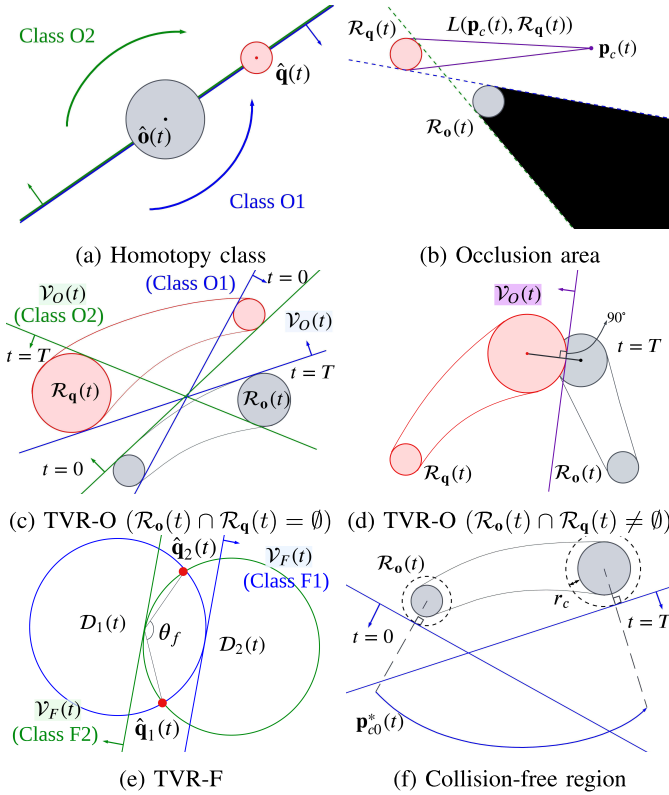


Fig. 6. (a): Homotopy class against a single obstacle. (b): A drone in a black-colored area cannot see the whole reachable area of the target. (c) TVR-O (blue: *Class O1*, green: *Class O2*) when the reachable area of the target (red) and obstacles (grey) are distant from each other. (d) TVR-O (purple) when the reachable areas of the target (red) and obstacles (grey) overlap. (e) TVR-F (blue: *Class F1*, green: *Class F2*). (f): Collision-free region (blue) against an obstacle (grey).

Case 1: In the case where the reachable areas of the target and an obstacle do not overlap, the target invisible region is made by straight lines which are tangential to both $\mathcal{R}_q(t)$ and $\mathcal{R}_o(t)$, as shown in Fig. 6b. Accordingly, we define TVR-O as a half-space made by a tangential line between $\mathcal{R}_q(t)$ and $\mathcal{R}_o(t)$ according to the homotopy class (19), as shown in Fig. 6c. $\mathcal{V}_O(t)$ denotes TVR-O and is represented as follows.

$$\mathcal{V}_O(t) = \left\{ \mathbf{x}(t) \mid \delta \hat{\mathbf{q}}^\top(t) \begin{bmatrix} r_{qo}(t) \mp d_2(t) \\ \pm d_2(t) & r_{qo}(t) \end{bmatrix} \delta \mathbf{x}(t) - r_o(t) d_1^2(t) \geq 0 \right\} \quad (20)$$

where $r_{qo}(t) = r_q(t) + r_o(t)$, $d_1(t) = \|\delta \hat{\mathbf{q}}(t)\|_2$, and $d_2(t) = \sqrt{d_1^2(t) - r_{qo}^2(t)}$. The double signs in (20) are in the same order, where the lower and upper signs are for *Class O1* and *Class O2*, respectively.

Lemma 1: If $\mathbf{p}_c(t) \in \mathcal{V}_O(t)$, $\mathcal{L}(\mathbf{p}_c(t), \mathcal{R}_q(t)) \cap \mathcal{R}_o(t) = \emptyset$ is satisfied.

Proof: $\mathcal{V}_O(t)$ is a half-space which is convex, and both $\mathbf{p}_c(t)$ and $\mathcal{R}_q(t)$ belong to $\mathcal{V}_O(t)$. By the definition of convexity, all the *Lines-of-Sight* connecting the drone $\mathbf{p}_c(t)$ and all points in the reachable area of the target $\mathcal{R}_q(t)$ are included in $\mathcal{V}_O(t)$. Since the TVR-O defined in (20) is disjoint with $\mathcal{R}_o(t)$, $\mathcal{L}(\mathbf{p}_c(t), \mathcal{R}_q(t)) \cap \mathcal{R}_o(t) = \emptyset$.

Remark 1: In dual-target scenarios, in order to avoid the occlusion of a target by another target, (20) is additionally defined by considering one of the targets as an obstacle.

Case 2: For the case where the reachable areas overlap, TVR-O becomes a half-space made by a tangential line to $\mathcal{R}_o(t)$, which is perpendicular to a line segment connecting the centers of the target $\hat{\mathbf{q}}(t)$ and the obstacles $\hat{\mathbf{o}}(t)$, illustrated as straight lines in Fig. 6d. Then the TVR-O is represented as

$$\mathcal{V}_O(t) = \{ \mathbf{x}(t) \mid \delta \hat{\mathbf{q}}^\top(t) \hat{\mathbf{q}} \mathbf{x}(t) + r_q(t) d_1(t) \geq 0 \} \quad (21)$$

The reachable areas include the volumes of the moving objects along with the prediction error. Since TVR-O is defined based on the reachable areas, it enhances the visibility of the entire body of the target against the prediction error. For detailed formulations of (20) and (21), see Appendix C-A.

2) *Bernstein Polynomial Approximation:* In defining TVR-O, all terms in (20) and (21) are polynomials except $d_1(t)$ and $d_2(t)$. To make $\mathbf{p}_c(t) \in \mathcal{V}_O(t)$ affine with respect to the optimization variable \mathbf{c} , we first convert $d_1(t)$, $d_2(t)$ into Bernstein polynomials using the following numerical technique that originates from *Lagrange interpolation* with standard polynomial representation [32].

Suppose that, by using *Lagrange interpolation*, a non-polynomial function $f(t)$ is approximated by an \tilde{n} -th degree Bernstein polynomial function $\tilde{f}(t)$ over the time interval $[t_a, t_b]$:

$$\tilde{f}(t) = \tilde{\mathbf{f}}_n^\top \mathbf{b}_{\tilde{n}}(t), \quad t \in [t_a, t_b] \quad (22)$$

where $\tilde{\mathbf{f}}_n \in \mathbb{R}^{\tilde{n}+1}$ and $\mathbf{b}_{\tilde{n}}(t) \in \mathbb{R}^{\tilde{n}+1}$ are a coefficient vector and a Bernstein basis vector for $[t_a, t_b]$, respectively. To calculate the coefficients of approximated function $\tilde{\mathbf{f}}_n$, we utilize Bernstein Vandermonde matrix $B_{\tilde{n}}$ [33] as follows.

$$\tilde{\mathbf{f}}_n = B_{\tilde{n}}^{-1} \tilde{\mathbf{f}}_n,$$

where $B_{\tilde{n}} = \{b_{j,k}\} \in \mathbb{R}^{(\tilde{n}+1) \times (\tilde{n}+1)}$,

$$b_{j,k} = \binom{\tilde{n}}{k} \left(\frac{j}{\tilde{n}} \right)^k \left(\frac{\tilde{n}-j}{\tilde{n}} \right)^{\tilde{n}-k}, \quad 0 \leq j, k \leq \tilde{n},$$

$$\tilde{\mathbf{f}}_n = [f_0, f_1, \dots, f_{\tilde{n}}]^\top, \quad f_l = f \left(\left(1 - \frac{l}{\tilde{n}} \right) t_a + \frac{l}{\tilde{n}} t_b \right) \quad (23)$$

Using this technique, $d_1(t)$ and $d_2(t)$ in (20) and (21) are approximately represented as $\tilde{d}_1(t)$ and $\tilde{d}_2(t)$, respectively. With the approximated terms, we represent the target visibility constraint as follows.

Case 1: $t \in [T_{m-1}, T_m]$, s.t. $\mathcal{R}_q(t) \cap \mathcal{R}_o(t) = \emptyset$:

$$\delta \hat{\mathbf{q}}^\top(t) \begin{bmatrix} r_{qo}(t) \mp \tilde{d}_2(t) \\ \pm \tilde{d}_2(t) & r_{qo}(t) \end{bmatrix} \delta \mathbf{p}_c(t) - r_o(t) d_1^2(t) \geq 0 \quad (24a)$$

Case 2: $t \in [T_{m-1}, T_m]$, s.t. $\mathcal{R}_q(t) \cap \mathcal{R}_o(t) \neq \emptyset$:

$$\delta \hat{\mathbf{q}}^\top(t) \hat{\mathbf{q}} \mathbf{p}_c(t) - r_q(t) \tilde{d}_1(t) \geq 0 \quad (24b)$$

3) *FOV Constraints:* In addition to (24), the drone should avoid the region where it cannot see the two targets with its limited camera FOV. As shown in Fig. 6e, circles whose inscribed angle of an arc tracing points Q_1 and Q_2 at $\hat{\mathbf{q}}_1(t)$, $\hat{\mathbf{q}}_2(t)$ equals θ_f , are represented as follows:

$$\mathcal{D}_j(t) = \{ \mathbf{x}(t) \mid \|\mathbf{x}(t) - \mathbf{f}_j(t)\|_2 \leq r_f(t) \}, \quad j = 1, 2$$

$$\begin{aligned} \mathbf{f}_j(t) &= \frac{1}{2}(\hat{\mathbf{q}}_1(t) + \mathbf{q}_2(t)) + \frac{\cot \theta_f}{2} R \left((-1)^j \frac{\pi}{2} \right) \hat{\mathbf{q}}_1 \hat{\mathbf{q}}_2(t) \\ r_f(t) &= \frac{1}{2 \sin \theta_f} \|\hat{\mathbf{q}}_1 \hat{\mathbf{q}}_2(t)\|_2, \quad \hat{\mathbf{q}}_1 \hat{\mathbf{q}}_2(t) = \hat{\mathbf{q}}_2(t) - \hat{\mathbf{q}}_1(t) \end{aligned} \quad (25)$$

where $R(\theta)$ is the rotation matrix for a rotation of θ . According to the geometric property of an inscribed angle, $\angle Q_1 X Q_2 > \theta_f$, $\angle Q_1 Y Q_2 < \theta_f$ for $\forall X \in \mathcal{D}_z(t) \setminus \partial \mathcal{D}_z(t)$, $\forall Y \notin \mathcal{D}_z(t)$, where $\mathcal{D}_z(t)$ is represented as follows.

$$\mathcal{D}_z(t) = \begin{cases} \mathcal{D}_1(t) \cap \mathcal{D}_2(t) & \left(\text{if } \theta_f \geq \frac{\pi}{2} \right) \\ \mathcal{D}_1(t) \cup \mathcal{D}_2(t) & \left(\text{if } \theta_f < \frac{\pi}{2} \right) \end{cases} \quad (26)$$

Since the camera FOV is θ_f , the drone inside $\mathcal{D}_z(t)$ misses at least one target in the camera image inevitably, and the drone outside of $\mathcal{D}_z(t)$ is able to see both targets.

We define the target visible region considering camera FOV (TVR-F) as a half-space that does not include $\mathcal{D}_z(t)$ and is tangential to $\mathcal{D}_z(t)$, as shown in Fig. 6e. $\mathcal{V}_F(t)$ denotes TVR-F and is represented as follows.

$$\mathcal{V}_F(t) = \left\{ \mathbf{x}(t) \mid \pm \det \left(\begin{bmatrix} \hat{\mathbf{q}}_1 \mathbf{x}^\top(t) \\ \hat{\mathbf{q}}_1 \hat{\mathbf{q}}_2^\top(t) \end{bmatrix} \right) - \frac{1 + \cos \theta_f}{2 \sin \theta_f} \|\hat{\mathbf{q}}_1 \hat{\mathbf{q}}_2(t)\|_2^2 \geq 0 \right\} \quad (27)$$

The upper and lower signs in (27) are for *Class F1* and *Class F2*, respectively, which are defined as

$$\begin{cases} \text{Class F1 (if } \det([\hat{\mathbf{q}}_1 \mathbf{p}_c^\top(t); \hat{\mathbf{q}}_1 \hat{\mathbf{q}}_2^\top(t)]) \geq 0, \text{ at } t = 0) \\ \text{Class F2 (if } \det([\hat{\mathbf{q}}_1 \mathbf{p}_c^\top(t); \hat{\mathbf{q}}_1 \hat{\mathbf{q}}_2^\top(t)]) < 0, \text{ at } t = 0). \end{cases} \quad (28)$$

We enforce the drone to satisfy $\mathbf{p}_c(t) \in \mathcal{V}_F(t)$ in order to see both targets. For the details, see the Appendix C-B.

C. Collision Avoidance

To avoid collision with obstacles, we define a collision-free area as a half-space made by a tangential line to a set that is inflated by r_c from $\mathcal{R}_o(t)$. The tangential line is drawn considering a trajectory $\mathbf{p}_{c0}^*(t)$, which is the previous planning result. This is illustrated in Fig. 6f and represented as follows.

$$\mathcal{F}_c(t) = \left\{ \mathbf{x}(t) \mid \frac{\delta \mathbf{p}_{c0}^*(t)}{\|\delta \mathbf{p}_{c0}^*(t)\|_2} (\mathbf{x}(t) - \hat{\mathbf{o}}(t)) \geq (r_o(t) + r_c) \right\} \quad (29)$$

where $\delta \mathbf{p}_{c0}^*(t) := \mathbf{p}_{c0}^*(t) - \hat{\mathbf{o}}(t)$. As in (22), we approximate a non-polynomial term $\|\delta \mathbf{p}_{c0}^*(t)\|_2$ to a polynomial $\tilde{d}_3(t)$. With the approximated term $\tilde{d}_3(t)$, the collision constraint is defined as follows:

$$\delta \mathbf{p}_{c0}^*(t) \delta \mathbf{p}_c(t) - (r_o(t) + r_c) \tilde{d}_3(t) \geq 0 \quad (30)$$

Due to the fact that the multiplication of Bernstein polynomials is also a Bernstein polynomial, the left-hand side of the (24), (27), and (30) can be represented in a Bernstein polynomial form. With the non-negativeness of Bernstein basis, we make coefficients of each basis non-negative in order to keep the left-hand side non-negative, and (24), (27), and (30) turn into affine constraints with the decision vector \mathbf{c} . The constraints can be written as follows, and we omit details.

$$\begin{aligned} A_{\text{TVR-O}} \mathbf{c} - \mathbf{b}_{\text{TVR-O}} &\geq 0, \\ A_{\text{TVR-F}} \mathbf{c} - \mathbf{b}_{\text{TVR-F}} &\geq 0, \\ A_{\text{Colli}} \mathbf{c} - \mathbf{b}_{\text{Colli}} &\geq 0 \end{aligned} \quad (31)$$

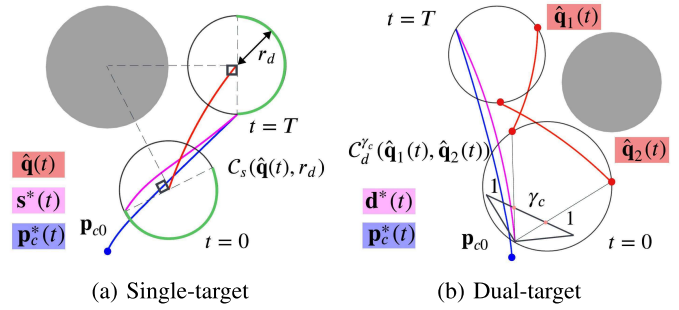


Fig. 7. Reference trajectory design in single- and dual-target (red) cases among obstacles (grey). (a): The points having the maximum visibility score ψ in $C_s(\hat{\mathbf{q}}(t), r_d)$ (green), a trajectory with the maximum ψ , $\mathbf{s}^*(t)$ (magenta), and the reference trajectory $\mathbf{p}_c^*(t)$ (blue line) considering current position of the drone (blue dot). (b): $C_d^{\gamma_c}(\hat{\mathbf{q}}_1(t), \hat{\mathbf{q}}_2(t))$ (black circle). The trajectory with high ψ , $\mathbf{d}^*(t)$ (magenta), and the reference trajectory $\mathbf{p}_c^*(t)$ (blue line) considering current position of the drone (blue dot).

D. Reference Trajectory for Target Tracking

In this section, we propose a reference trajectory for target chasing that enhances the visibility of the targets.

1) *Single-Target Case*: In designing the reference trajectory, we use the visibility score, as discussed in [34]. The definition of the visibility score ψ is the closest distance between all points in an j -th obstacle, \mathcal{O}_j , and the *Line-of-Sight* connecting the target and the drone:

$$\psi(\mathbf{p}_c(t); \hat{\mathbf{q}}(t), \mathcal{O}_j) = \min_{\substack{\mathbf{x} \in \mathcal{L}(\mathbf{p}_c(t), \hat{\mathbf{q}}(t)) \\ \mathbf{y} \in \mathcal{O}_j}} \|\mathbf{x} - \mathbf{y}\|_2. \quad (32)$$

In order to keep the projected size of the target on the camera image, we set the desired shooting distance r_d . With the desired shooting distance r_d , a viewpoint candidate set can be defined as $C_s(\hat{\mathbf{q}}(t), r_d) = \{\mathbf{x}(t) \mid \|\mathbf{x}(t) - \hat{\mathbf{q}}(t)\|_2 = r_d\}$. Under the assumption that the environment consists of cylindrical obstacles, half-circumference of $C_s(\hat{\mathbf{q}}(t), r_d)$ acquires the maximum visibility score, as illustrated in green in Fig. 7a. Therefore, the following trajectory maintains the maximum $\psi(\mathbf{p}_c(t); \hat{\mathbf{q}}(t), \mathcal{O}_j)$.

$$\mathbf{s}_j(t) = \mathbf{q}(t) + r_d R \left(\mp \frac{\pi}{2} \right) \delta_{s,j}(t), \quad \delta_{s,j}(t) = \frac{\hat{\mathbf{q}}(t) - \hat{\mathbf{o}}_j(t)}{\|\hat{\mathbf{q}}(t) - \hat{\mathbf{o}}_j(t)\|_2} \quad (33)$$

The upper and lower signs are for *Class O1* and *Class O2*, respectively. We define the reference trajectory as a weighted sum of $\mathbf{s}_j(t)$.

$$\mathbf{s}^*(t) = \sum_{j=1}^{N_o} w_{s,j} \mathbf{s}_j(t), \quad \text{where } \sum_{j=1}^{N_o} w_{s,j} = 1 \quad (34)$$

$w_{s,j}$'s are weight functions that are inversely proportional to the current distance between the target and each obstacle.

2) *Dual-Target Case*: In order to make aesthetically pleasing scenes, we aim to place two targets in a ratio of $1 : \gamma_c : 1$ on the camera image. To do so, the drone must be within a set $C_d^{\gamma_c}(\hat{\mathbf{q}}_1(t), \hat{\mathbf{q}}_2(t))$, which is defined as follows and illustrated as a black circle in Fig. 7.

$$\begin{aligned} C_d^{\gamma_c}(\hat{\mathbf{q}}_1(t), \hat{\mathbf{q}}_2(t)) &= \{\mathbf{x}(t) \mid \|\mathbf{x}(t) - \mathbf{f}_c(t)\|_2 = r_{f,c}\}, \\ \mathbf{f}_c(t) &= \frac{1}{2}(\hat{\mathbf{q}}_1(t) + \hat{\mathbf{q}}_2(t)) \pm \kappa_1 R \left(-\frac{\pi}{2} \right) \hat{\mathbf{q}}_1 \hat{\mathbf{q}}_2(t), \\ r_{f,c}(t) &= \kappa_2 \|\hat{\mathbf{q}}_1 \hat{\mathbf{q}}_2(t)\|_2, \end{aligned}$$

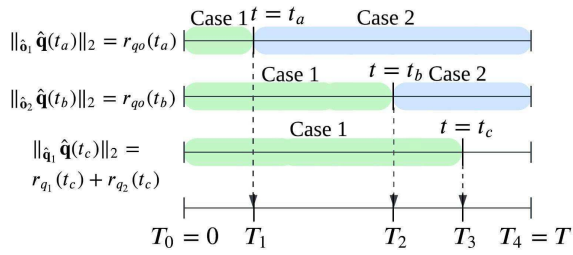


Fig. 8. Time segmentation. Green and blue regions are time intervals when the (24a) and (24b) are adopted as the target visibility constraints, respectively.

$$\begin{aligned} \kappa_1 &= \frac{\gamma_c + 2}{4\gamma_c} \cot \frac{\theta_f}{2} \left(1 - \frac{\gamma_c^2}{(\gamma_c + 2)^2} \tan^2 \frac{\theta_f}{2} \right), \\ \kappa_2 &= \frac{\gamma_c + 2}{4\gamma_c} \cot \frac{\theta_f}{2} \left(1 + \frac{\gamma_c^2}{(\gamma_c + 2)^2} \tan^2 \frac{\theta_f}{2} \right) \end{aligned} \quad (35)$$

The upper and lower signs are for *Class F1* and *Class F2*.

AA.

We define the reference trajectory $\mathbf{d}^*(t)$ to acquire a high visibility score ψ while maintaining the ratio.

$$\begin{aligned} \mathbf{d}^*(t) &= \mathbf{f}_c(t) + r_{f,c}(t) \frac{\delta_d(t)}{\|\delta_d(t)\|_2}, \\ \delta_d(t) &= \delta_b(t) + \sum_{j=1}^{N_o} w_{s1,j} \delta_{s1,j}(t) + w_{s2,j} \delta_{s2,j}(t), \\ \delta_b(t) &= R \left(\mp \frac{\pi}{2} \right) \frac{\hat{\mathbf{q}}_2(t) - \hat{\mathbf{q}}_1(t)}{\|\hat{\mathbf{q}}_2(t) - \hat{\mathbf{q}}_1(t)\|_2} \end{aligned} \quad (36)$$

The upper and lower signs in $\delta_b(t)$ are for *Class F1* and *Class F2*, respectively, and $w_{s1,j}$, $\delta_{s1,j}(t)$ and $w_{s2,j}$, $\delta_{s2,j}(t)$ correspond to the $w_{s,j}$, $\delta_{s,j}(t)$ in (33) and (34) for the target 1 and 2. $\delta_b(t)$ is defined as the farthest direction from the two targets to maximize a metric, $\psi(\mathbf{p}_c(t); \hat{\mathbf{q}}_1(t), \mathcal{B}(\hat{\mathbf{q}}_2(t), r_{q_0})) + \psi(\mathbf{p}_c(t); \hat{\mathbf{q}}_2(t), \mathcal{B}(\hat{\mathbf{q}}_1(t), r_{q_0}))$, representing the visibility score between two targets.

For the numerical stability of the QP solver, the reference trajectory is redefined by interpolating between $\boldsymbol{\mu}^*(t)$, $\boldsymbol{\mu} = \mathbf{s}$, \mathbf{d} and the current drone's position \mathbf{p}_{c0} . With a non-decreasing polynomial function $\alpha(t)$ that satisfies $\alpha(0) = 0$ and $\alpha(T) = 1$, the reference trajectory is defined as follows.

$$\mathbf{p}_c^*(t) = (1 - \alpha(t)) \mathbf{p}_{c0} + \alpha(t) \boldsymbol{\mu}^*(t) \quad (37)$$

Since trigonometric terms for $\boldsymbol{\mu}^*(t)$, $\boldsymbol{\mu} = \mathbf{s}$ or \mathbf{d} in (34), (36) are non-polynomial, the *Lagrange interpolation* is used as (22). The approximated $\mathbf{p}_c^*(t)$ is denoted as $\tilde{\mathbf{p}}_c^*(t)$. Based on the construction of reference trajectory and the target visibility constraints, we formulate the trajectory optimization problem as a QP problem.

E. QP Formulation

1) *Trajectory Segmentation*: The constraint (24) is divided into the cases: when the reachable areas are separated and when they overlap. To apply (24) according to the situation, the roots of two equations $\|\hat{\mathbf{q}}_1(t)\|_2 = r_{q_0}(t)$ and $\|\hat{\mathbf{q}}_1(t) - \hat{\mathbf{q}}_2(t)\|_2 = r_{q_1}(t) + r_{q_2}(t)$ should be investigated for $t \in (0, T)$. We define $[T_0, \dots, T_M]$ by arranging the roots of the above equations in ascending order, and we set M time intervals, as stated

in (8). Fig. 8 visualizes the above process. Thanks to *De Casteljau's* algorithm [35], the single Bernstein polynomial such as $\hat{\mathbf{q}}(t)$, $\hat{\mathbf{o}}(t)$, $r_q(t)$, $r_o(t)$ can be divided into M Bernstein polynomials. Using an M -segment-polynomial representation, we formulate a QP problem.

2) *Constraints*: The drone's trajectory is constrained to maintain the target visibility (3) and (4), avoid collisions (5), and satisfy dynamic feasibility (6). To construct the QP problem, we formulate the constraints to be affine with respect to \mathbf{c} . By leveraging the properties of Bernstein polynomials, which include *P1*) the first and last coefficients corresponding to the endpoints and *P2*) the convex hull property, we can effectively formulate dynamic constraints. Also, the l -th derivative of an n -th order Bernstein polynomial is an $(n-l)$ -th order polynomial. The coefficients of the l -th derivative of a polynomial are obtained by multiplying the coefficients of the original polynomial by the following matrix.

$$\begin{aligned} E_{n,m}^{(l)} &= \begin{cases} \frac{n-l+1}{T_m - T_{m-1}} E_{n-l} E_{n,m}^{(l-1)} & \text{if } l \geq 1 \\ \mathbf{I}_{(n+1) \times (n+1)} & \text{if } l = 0 \end{cases}, \quad n \geq l \geq 0, \\ E_k &= \begin{bmatrix} -1 & 1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & -1 & 1 \end{bmatrix} \in \mathbb{R}^{k \times (k+1)} \end{aligned} \quad (38)$$

First, the trajectory is made to satisfy the initial conditions: the current position and velocity. Using *P1*), we can get

$$\mathbf{e}_{n_c+1,l}^\top E_{n_c,m}^{(0)} \mathbf{C}_m = \mathbf{p}_{c0}^\top, \quad (39a)$$

$$\mathbf{e}_{n_c,l}^\top E_{n_c,m}^{(1)} \mathbf{C}_m = \mathbf{p}'_{c0}^\top, \quad (39b)$$

where $\mathbf{e}_{n,l}$ is an $n \times 1$ one-hot vector where the l -th element is 1, and all other elements are 0. Second, by using *P1*), the \mathcal{C}_2 continuity between consecutive segments is achieved by the following constraints.

$$\begin{aligned} \mathbf{e}_{n_c+1-l,n_c+1-l}^\top E_{n_c,m}^{(l)} \mathbf{C}_m &= \mathbf{e}_{n_c+1-l,l}^\top E_{n_c,m+1}^{(l)} \mathbf{C}_{m+1}, \\ l &= 0, 1, 2, \quad m = 1, 2, \dots, M-1 \end{aligned} \quad (40)$$

Third, by using *P2*), the constraints of velocity and acceleration limits, (6), are formulated as follows.

$$\|\mathbf{e}_{n_c,k}^\top E_{n_c,m}^{(1)} \mathbf{C}_m\|_\infty \leq \frac{1}{\sqrt{2}} v_{\max}, \quad (41a)$$

$$\begin{aligned} \|\mathbf{e}_{n_c-1,l}^\top E_{n_c,m}^{(2)} \mathbf{C}_m\|_\infty &\leq \frac{1}{\sqrt{2}} a_{\max}, \\ k &= 1, \dots, n_c, \quad l = 1, \dots, n_c - 1, \quad m = 1, \dots, M \end{aligned} \quad (41b)$$

Since \mathbf{c} can be acquired by flattening $[\mathbf{C}_1, \dots, \mathbf{C}_M]$, the dynamic constraints (39)-(41) are affine with respect to \mathbf{c} . Then, the constraints in (31) are applied to ensure the target visibility and the drone's safety.

3) *Costs*: The optimization cost is divided into two terms: J_j and J_e . First, J_j penalizes the jerkiness of the drone's path.

$$\begin{aligned} J_j &= \int_0^T \|\mathbf{p}_c'''(t)\|_2^2 dt, \\ &= \sum_{m=1}^M \int_0^T \|\mathbf{b}_{n_c-3,m}^\top E_{n_c,m}^{(3)} \mathbf{C}_m\|_2^2 dt \end{aligned}$$

TABLE IV
 COMPUTATION TIME IN QP SOLVER

	Computation time [ms]	$n_c = 4$					$n_c = 5$					$n_c = 6$							
		# polynomial segment (M)					# polynomial segment (M)					# polynomial segment (M)							
		1	2	3	4	5	1	2	3	4	5	1	2	3	4	5			
Single	# obstacle (N_o)	1	0.17	0.54	.	.	.	0.19	0.77	.	.	.	0.21	1.16	.	.	.		
		2	0.18	0.71	2.07	.	.	0.22	1.12	2.95	.	.	0.24	1.66	4.45	.	.		
		3	0.29	0.98	2.91	6.05	.	.	0.34	1.23	4.58	8.05	.	.	0.43	1.82	5.62	12.3	.
		4	0.32	1.01	2.75	7.81	12.2	.	0.39	1.38	4.24	12.6	18.8	.	.	0.55	1.88	6.81	16.8
Dual	# obstacle (N_o)	1	0.18	0.66	1.94	5.34	.	0.20	0.91	2.95	7.61	.	0.26	1.38	4.46	10.5	.		
		2	0.21	1.21	3.94	5.82	13.6	.	0.46	1.72	6.78	9.48	21.4	.	0.55	2.41	10.8	17.8	36.3
		3	0.43	1.67	3.86	8.54	17.8	.	0.78	2.51	5.62	13.5	28.6	.	0.81	3.72	8.91	21.8	43.6

$$\begin{aligned}
 &= \sum_{m=1}^M \text{tr} \left(\mathbf{C}_m^\top \mathbf{Q}_{n_c, m}^f \mathbf{C}_m \right), \text{ where } \text{tr}(\cdot) \text{ is a trace operator,} \\
 \mathbf{Q}_{n_c, m}^f &= \frac{T_m - T_{m-1}}{2n_c - 5} (\mathbf{E}_{n_c, m}^{(3)})^\top \mathbf{Q}_{n_c}^f \mathbf{E}_{n_c, m}^{(3)}, \\
 \mathbf{Q}_{n_c}^f &= \{q_{k,l}^f\} \in \mathbb{R}^{(n_c-2) \times (n_c-2)}, q_{k,l}^f = \frac{\binom{n_c-3}{k} \binom{n_c-3}{l}}{\binom{2n_c-6}{k+l}} \quad (42)
 \end{aligned}$$

Second, J_e minimizes tracking error to the designed reference trajectory $\tilde{\mathbf{p}}_c^*(t)$.

$$\begin{aligned}
 J_e &= \int_0^T \|\mathbf{p}_c(t) - \tilde{\mathbf{p}}_c^*(t)\|_2^2 dt \\
 &= \sum_{m=1}^M \int_0^T \|\mathbf{b}_{n_c, m}^\top(t) (\mathbf{C}_m - \tilde{\mathbf{C}}_m^*)\|_2^2 dt \\
 &= \sum_{m=1}^M \text{tr} \left((\mathbf{C}_m - \tilde{\mathbf{C}}_m^*)^\top \mathbf{Q}_{n_c, m}^e (\mathbf{C}_m - \tilde{\mathbf{C}}_m^*) \right)
 \end{aligned}$$

where $\mathbf{Q}_{n_c, m}^e = \frac{T_m - T_{m-1}}{2n_c + 1} \mathbf{Q}_{n_c}^e$,

$$\mathbf{Q}_{n_c}^e = \{q_{k,l}^e\} \in \mathbb{R}^{(n_c+1) \times (n_c+1)}, q_{k,l}^e = \frac{\binom{n_c}{k} \binom{n_c}{l}}{\binom{2n_c}{k+l}} \quad (43)$$

Similarly to the constraint formulation, the cost, $J = w_j J_j + w_e J_e$, can be expressed with $\underline{\mathbf{c}}$ and becomes quadratic in $\underline{\mathbf{c}}$, where w_j and w_e are nonnegative weight factors.

Since all the constraints are affine, the cost is quadratic with respect to $\underline{\mathbf{c}}$, and Hessian matrices of the cost, $w_j \mathbf{Q}_{n_c, m}^j + w_e \mathbf{Q}_{n_c, m}^e$, $m = 1, \dots, M$, are positive semi-definite, the proposed trajectory optimization problem can be formulated as a convex QP problem.

$$\begin{aligned}
 \min_{\underline{\mathbf{c}}} & \quad \underline{\mathbf{c}}^\top \mathbf{Q}_c \underline{\mathbf{c}} + \mathbf{g}_c^\top \underline{\mathbf{c}} \\
 \text{s.t.} & \quad A_c \underline{\mathbf{c}} \geq \mathbf{b}_c \quad (44)
 \end{aligned}$$

F. Evaluation

1) *Computation Time*: We set n_c as 4, 5, 6 and test 5000 times for different scenarios $N_o = 1, 2, 3, 4$. qpOASES [36] is utilized as a QP solver, and the execution time to solve the QP problem is summarized in Table IV. As shown in the table, the computation time increases as the number of segments and obstacles increases.

In the realistic simulations and hardware experiments presented in Section VII, the number of polynomial segments

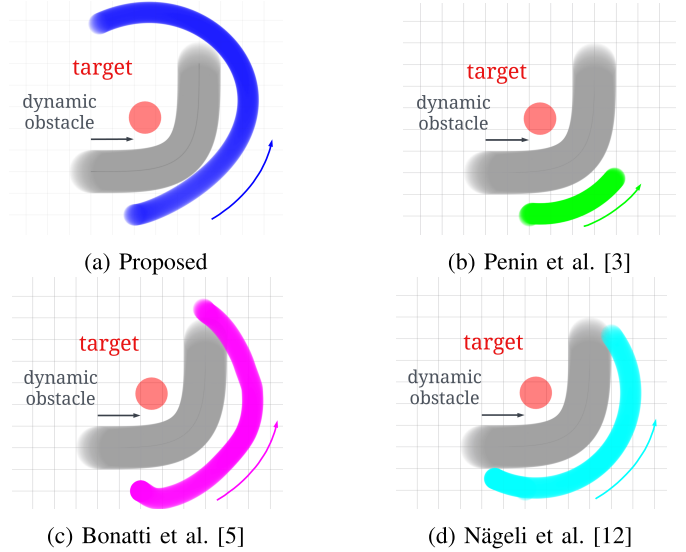


Fig. 9. Comparison results with relevant chasing planners: proposed (blue), [3] (green), [5] (magenta), and [12] (cyan).

is reported to be at most $M = 3$, and a planning frequency exceeding 40 Hz is achieved under the setting with $n_c = 6$. However, in extreme scenarios where multiple obstacles aggressively approach the target, the number of active constraints can significantly increase. Additionally, an increase in M results in a higher number of optimization variables. These factors combined may lead to a sharp increase in the planner's runtime. To satisfy real-time criteria (10 Hz) even as the number of close obstacles increases, we reduce $r_q(t)$ by defining N_ϵ furthest primitives as outliers, removing them from \mathcal{P}_s , and recalculating $\mathcal{R}_q(t)$. We determine the number of outliers $N_\epsilon = \epsilon_p |\mathcal{P}_s|$ with a factor ϵ_p . The effect of ϵ_p is visualized in Fig. 4, and it can lower the number of segments M . Although there may be a potential reduction in responsiveness to unforeseen target movements, real-time planning is achievable even in complex and dense environments with this strategy.

2) *Benchmark Test*: We now demonstrate the proposed trajectory planner's capability to maintain the target visibility against dynamic obstacles. The advantages of the proposed planner are analyzed by comparing it with non-convex optimization-based planners: [3], [5], and [12]. We consider a scenario where the target is static and a dynamic

TABLE V
CHASING PLANNER COMPARISON

	[3]	[5]	[12]	Proposed
Avg. of jerk [m/s^3]	733.9	430.4	774.1	2.379
(Safe duration) (Total duration)	1.0	0.986	0.938	1.0
(Visible duration) (Total duration)	0.587	0.092	0.046	1.0
Computation time [ms]	15.5	5.67	28.7	0.42

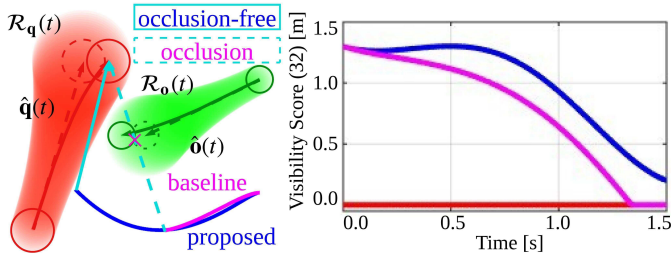


Fig. 10. Comparison of planning with and without consideration of the path prediction error. **Left:** A blue spline is a generated trajectory by the proposed planner, considering $\mathcal{R}_q(t)$ (red-shaded) and $\mathcal{R}_o(t)$ (green-shaded). A magenta spline is a trajectory generated by the baseline [15], which fully trusts $\hat{q}(t)$ (red-dashed) and $\hat{o}(t)$ (green-dashed). If a target and an obstacle move along red and green solid lines respectively, the *Line-of-Sight* (cyan, solid) connecting our trajectory and the target's trajectory does not collide with the obstacle, whereas the baseline's *Line-of-Sight* (cyan, dashed) collides, indicating occlusion. **Right:** The plot of visibility score (32) when the target and obstacle move along the solid splines. A value equal to the red line represents occlusion.

obstacle cuts in between the target and the drone. The total duration of the scenario is 10 seconds, and the position and velocity of the obstacle are updated every 20 milliseconds. We set n_c as 5 and the planning horizon as 1.5 seconds for all planners. Fig. 9 visualizes the history of the positions of the target, the drone, and the obstacles, and Table V summarizes the performance indices related to jerkiness, collisions, and occlusions.

All other nonlinear optimization-based planners fail to avoid occlusion and collision because the constraints are applied softly or the optimization converges to a sub-optimal solution. In contrast, our planner considers future obstacle movements and successfully finds trajectories by treating the drone's safety and the target visibility as hard constraints in a single QP. In addition, the computation efficiency of our method is an order of magnitude greater than other methods.

We also compare our planner with a motion-primitive-based planner [15] that does not account for the path prediction error. Fig. 10 illustrates a situation where occlusion may occur, highlighting that the success of planning can depend on whether the prediction error is considered. Even if the results of the planning method that does not consider the error are updated quickly, occlusion can still occur if the error accumulates. In contrast, our planner generates relatively large movements, which are suitable for maintaining the target visibility when obstacles approach.

3) *Weight Factor Effect:* We test the influence of the two weight factors, w_j and w_e , in the QP problem (44) on the tracking trajectory. Fig. 11 shows the resulting paths according

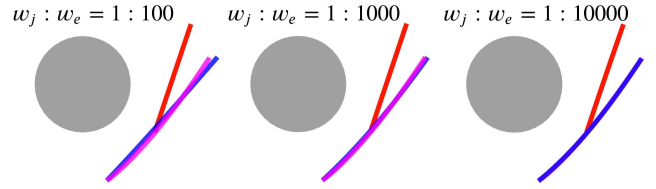


Fig. 11. Effect of weight factors (w_j and w_e) on the optimization result. The grey circle represents an obstacle, while the red, magenta, and blue splines indicate the target trajectory, reference trajectory, and optimization result, respectively.

TABLE VI
IMPACT OF WEIGHT FACTORS ON OPTIMIZATION

$w_j : w_e$	J_j [m^2/s^5]	J_e [$\text{m}^2 \cdot \text{s}$]	ψ (32) [m] (min/mean)
1:100	0.1721	0.011	0.5897/0.7982
1:1000	3.1809	0.0032	0.6071/0.8128
1:10000	9.3761	0.0097	0.6201/0.8234

to the ratio of the two weight factors, and Table VI presents the quantitative results. A higher w_j reduces jerkiness, while a higher w_e decreases the tracking error. Although there are differences in the visibility score (32), they are not significant enough to affect mission success. Within this range, the weights can be set according to the user's preference.

4) *Critical Analysis:* We now study the harsh situations that can fail to maintain the target visibility. The optimization cannot be solved for the following reasons: 1) conflicts between the target visibility constraints and actuator limit constraints and 2) incompatibility among the target visibility constraints. For example, when a dynamic obstacle cuts in between the target and the drone at high speed, the huge variation in the TVR over time conflicts with the actuator limits. Next, a situation where two dynamic obstacles cut in simultaneously from opposite directions may incur conflicts between the target visibility constraints. These obstacles force the drone to move in opposite directions. Target occlusion inevitably occurs when the target and obstacles align. In these cases, we formulate the optimization problem considering constraints, excluding the target visibility constraint. While occlusion may occur, drone safety is the top priority, and planning considering the target visibility resumes once conditions return to normal.

VII. CHASING SCENARIO VALIDATION

In this section, the presented method is validated through realistic simulators and real-world experiments.

A. Implementation Details

We perform AirSim [37] simulations for a benchmark test of our method with [15] and [16]. In real-world experiments, we install two ZED2 cameras facing opposite directions. The front-view camera is used for object detection and tracking, while the rear-view camera is used for the localization of the robot. Since the rear-view camera does not capture multiple moving objects in the camera image, the localization system is free from visual interference caused by dynamic objects, thereby preventing degradation in localization performance.

TABLE VII
PROBLEM SETTINGS

Scenario Settings		
Name	Single Target	Dual Target
drone radius (r_c) [m]	0.4	0.4
camera FOV (θ_f) [$^\circ$]	120	120
Prediction Parameters		
Name	Single Target	Dual Target
time horizon (T) [s]	1.5	1.5
# sampled points (N_{samp})	2000	2000
Planning Parameters		
Name	Single Target	Dual Target
polynomial degree (n_c)	6	6
max velocity (v_{max}) [m/s]	4.0	4.0
max acceleration (a_{max}) [m/s ²]	5.0	5.0
shooting distance (r_d) [m]	4.0	.
screen ratio (γ_c)	.	1.0
tracking weight (w_e)	10.0	10.0
jerk weight (w_j)	0.01	0.01

Additionally, we utilize 3D human pose estimation to determine the positions of the humans. The humans in experiments are distinguished by color as described in [16]. By using depth information and an algorithm from [38], we build a static map. To distribute the computational burden, we use the Jetson Xavier NX to process data from the ZED2 cameras and the Intel NUC to build maps and run the QP-Chaser algorithm. To manage the payload on the drone, we utilize a coaxial octocopter and deploy Pixhawk4 as the flight controller.

The parameters used in validations are summarized in Table VII. As the planning result is frequently updated to respond to dynamic obstacles and targets, a long planning horizon is unnecessary; therefore, T is set to a short value of 1.5 seconds. The desired distance r_d is set to 4 meters, as this value yields visually pleasing target capture with the ZED2 camera's intrinsic parameters, as determined empirically. Additionally, we set the screen ratio, γ_c , to 1 to pursue the rule of thirds in cinematography. Lastly, the QP weight parameters were set to $w_e : w_j = 1 : 1000$ based on the results in Section VI-F.3, in order to achieve an appropriate balance of visibility and smoothness.

B. Evaluation Metrics

In the validations, we evaluate the performance of the proposed planner using the following metrics: drone safety and target visibility. We measure the distances between the drone and targets (45a), and the minimum distance between the drone and obstacles (45b), to evaluate drone's safety against targets and obstacles, respectively. To evaluate the target visibility, the visibility score (32) is computed against all obstacles, and we calculate the minimum value (45c) to assess the degree of occlusion in Cartesian coordinates, as discussed in [34].

$$\chi_1(t) := \|\mathbf{q}\mathbf{p}_c(t)\|_2 - r_c - r_q \quad (45a)$$

$$\chi_2(t) := \min_{j: \mathcal{O}_j \in \mathcal{O}} \|\mathbf{o}_j \mathbf{p}_c(t)\|_2 - r_{o_j} - r_c \quad (45b)$$

$$\psi_1(t) := \min_{j: \mathcal{O}_j \in \mathcal{O}} \min_{\mathbf{x} \in \mathcal{L}(\mathbf{p}_c(t), \hat{\mathbf{q}}(t))} \min_{\mathbf{y} \in \mathcal{O}_j} \|\mathbf{x} - \mathbf{y}\|_2. \quad (45c)$$

Also, we assess the tracking performance in the image plane. First, we measure multi-object tracking accuracy (MOTA) and IDF1 to evaluate tracking accuracy. Then, we measure the visibility proportion (46) to evaluate the degree of occlusion caused by obstacles in the image plane. The score $\psi_2(t)$ means the proportion of the unobstructed part of the targets' bounding box. Since the bounding boxes can generally be obtained by most object detection algorithms, this metric is employed for generality and is formulated as follows:

$$\psi_2(t) := \begin{cases} 1 - \frac{b_q(t) \cap b_o(t)}{b_q(t)} & (\text{if } \|\mathbf{q}\mathbf{p}_c(t)\|_2 \geq \|\mathbf{o}\mathbf{p}_c(t)\|_2) \\ 1 & (\text{otherwise}) \end{cases} \quad (46)$$

where $b_q(t)$ and $b_o(t)$ represent the bounding boxes of the targets and obstacles in the camera image, respectively. In the dual-target chasing scenarios, one target can be an obstacle occluding the other target, and the above metrics are measured accordingly. Throughout all validations, we use YOLO11 [39] for the object tracking.

C. Simulations

We test the proposed planner in scenarios where two actors run in the forest. To show the robust performance of the planner, we bring the following four scenarios. In the first two scenarios, one actor is the target and the other is the interrupter. In the remaining two scenarios, both actors are targets.

- **Scenario 1:** The interrupter runs around the target, ceaselessly disrupting the shooting.
- **Scenario 2:** The interrupter intermittently cuts in between the target and the drone.
- **Scenario 3:** The two targets run while maintaining a short distance between them.
- **Scenario 4:** The two targets run while varying the distance between them.

Of the many flight tests, we extract and report some of the 30-40 seconds long flights. We compare the proposed planner with the other state-of-the-art planners [15] in single-target chasing scenarios and [16] in dual-target chasing scenarios. Fig 12 show the comparative results with key snapshots, and Table VIII shows that the proposed approach makes the drone follow the target while maintaining the target visibility safely, whereas the baselines cause collisions and occlusions. There are rare cases in which partial occlusion occurs with our planner due to the limitations of the PD controller provided by the Airsim simulator. However, in the same situation, full occlusions occur with the baselines.

In single-target scenarios, [15] fails because it alters the chasing path homotopy when obstacles are close to the drone. In dual-target scenarios, [16] calculates good viewpoints considering the visibility score (45c) and camera FOV, but the path between these viewpoints does not guarantee the target visibility. In contrast, our planner succeeds in chasing because we consider these factors as hard constraints. Moreover, despite the heavy computational demands of the physics engine, the planning pipeline is executed within 25 milliseconds, which is faster than the baselines.

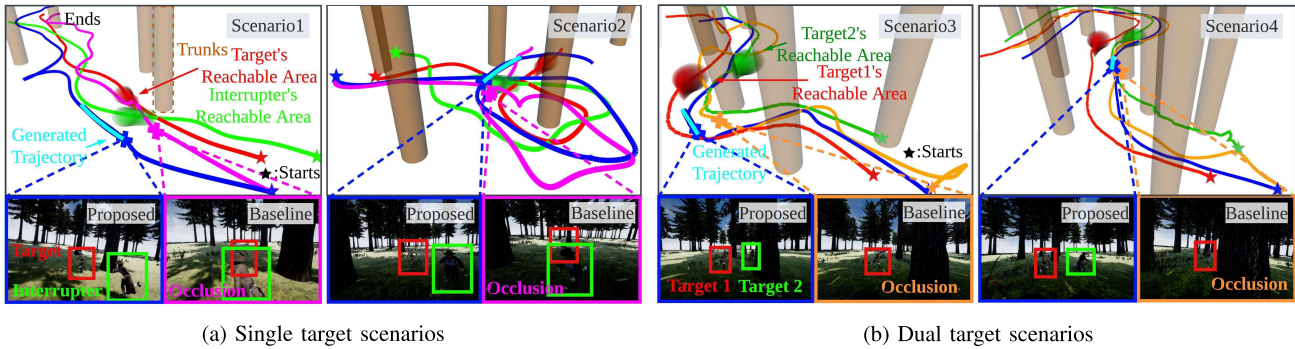


Fig. 12. Aerial tracking in AirSim environments. Scenarios 1-2 and 3-4 show simulation results for single-target and dual-target scenarios, respectively. Top figures present total position histories of moving targets and chasing drones. The bottom figures are snapshots of the camera images when the interrupter intersects the target's path and tries to conceal the target's body. The proposed planner (a blue cross) generates the trajectory (cyan) maintaining the target visibility. At the same time, [15] and [16] (magenta and orange crosses) fails to avoid target occlusion by the interrupter.

TABLE VIII
COMPARISON BETWEEN THE PROPOSED PLANNER AND BASELINES (SIMULATION)

Metrics	Planner	Single Target		Dual Target	
		Scenario 1	Scenario 2	Scenario 3	Scenario 4
Target Distance (45a) [m] (\dagger)	proposed	1.595/3.726	0.995/2.562	1.213/2.204	1.505/3.135
	baseline	1.198/3.298	-0.131/3.280	1.060/3.610	0.890/2.611
Obstacle Distance (45b) [m] (\dagger)	proposed	0.872/4.070	0.714/3.320	0.428/1.873	0.315/2.847
	baseline	-0.264/1.715	-0.364/2.520	0.302/1.901	0.059/3.207
Visibility Score (45c) [m] (\dagger)	proposed	0.687/2.369	0.028/2.529	0.193/1.187	0.151/1.975
	baseline	0.0/1.183	0.0/1.649	0.0/0.834	0.0/2.103
MOTA (\uparrow)	proposed	0.987	0.917	0.913	0.934
	baseline	0.939	0.808	0.882	0.879
IDF1 (\uparrow)	proposed	0.994	0.990	0.997	0.997
	baseline	0.969	0.894	0.974	0.972
Visibility Proportion (46) (\uparrow)	proposed	1.0/1.0	0.523/0.996	0.726/0.998	0.951/0.999
	baseline	0.0/0.954	0.0/0.926	0.0/0.942	0.0/0.951
Computation Time [ms]	proposed	13.64	15.41	20.53	21.22
	baseline	29.45	32.12	79.61	81.37

The upper and lower data in each metric represent the reported metrics of the proposed planner and the baselines (single target: [15], dual target: [16]), respectively. The values for the first, second, third, and sixth metrics indicate the (minimum/mean) performance. \dagger means that a value equal to or below 0 indicates a collision or occlusion. \uparrow means that higher is better.

D. Experiments

To validate our chasing strategy, we conduct real-world experiments with several single- and dual-target tracking scenarios. Two actors move in an 8×11 [m²] indoor space with stacked bins. As in the simulations, one actor is the target, and the other is the interrupter in single-target scenarios, while the two actors are both targets in dual-target scenarios.

For the single-target chasing, we bring the following scenarios.

- **Scenario S1:** The drone senses the bins as static obstacles and follows the target.
- **Scenario S2:** The target moves away from the drone, and the interrupter cuts in between the target and the drone.
- **Scenario S3:** Two actors rotate in a circular way, causing the target to move away from the drone, while the interrupter consistently disrupts the visibility of the target.
- **Scenario S4:** The interrupter intentionally hides behind bins and appears abruptly to obstruct the target.

For the dual-target chasing, we bring the following scenarios.

- **Scenario D1:** The targets move in a circular pattern. To capture both targets in the camera view, the drone also moves in circles.

- **Scenario D2:** The targets move while repeatedly increasing and reducing their relative distance. The drone adjusts its distance from the targets to keep them within the image.
- **Scenario D3:** The drone detects the bins and follows the targets among them.

Figs. 13 and 14 show the histories with key snapshots of the single- and dual-target experiments, respectively. The planner generates chasing trajectories in response to the targets' diverse movements and obstacles' interference. Table IX confirms that the drone successfully followed the targets without collision and occlusion.

We use compressed RGB and depth images in the pipeline to record the camera images in limited onboard computer storage and to transfer images between computers. Since the compressed data is acquired at a slow rate, information about moving objects is updated slowly. Therefore, as shown in Figs. 13 and 14, due to intermittent data, the planning module interprets the target as making sudden movements. Nevertheless, the drone keeps the target in view and ensures its own safety by quickly updating its trajectory, taking prediction error into account. This implicitly shows that the proposed system can effectively handle situations where the target actually exhibits abrupt motions.

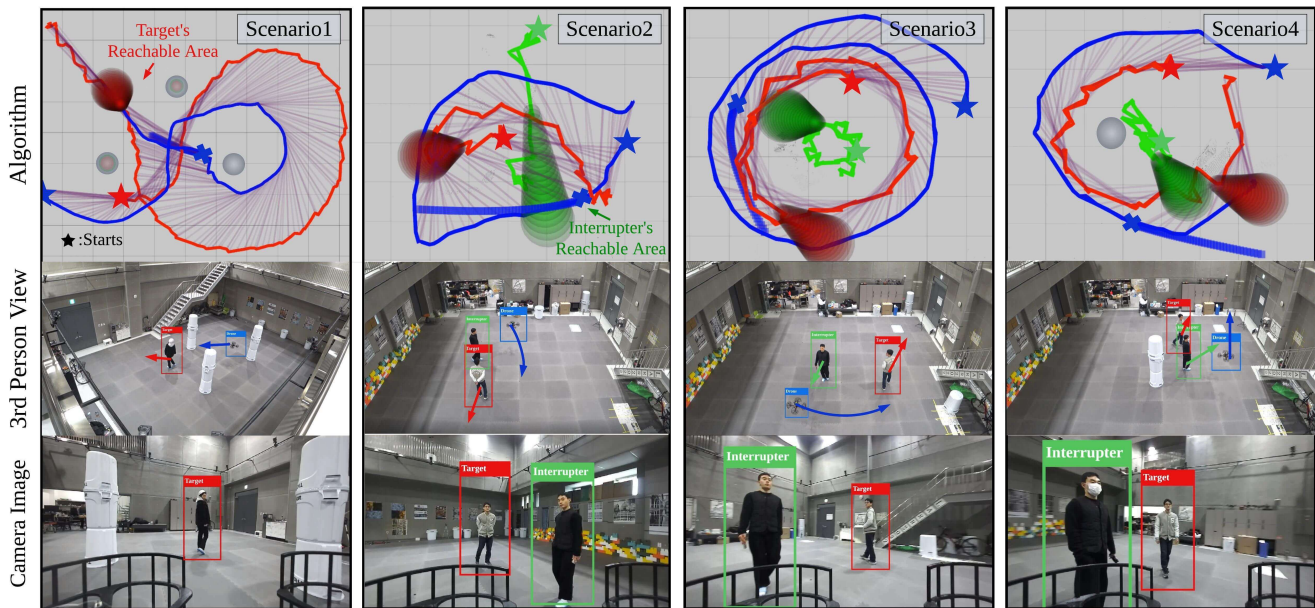


Fig. 13. Autonomous aerial tracking in an indoor environment with single-target scenarios. Blue, red, and green curves represent reported paths moved by the drone, the target, and the interrupter, respectively. Blue crosses mean the position of the drone at captured moments, and purple segments represent lines-of-sight between the target and the drone. The short, thick blue splines, which start from blue crosses, represent the generated trajectory for 1.5 seconds.

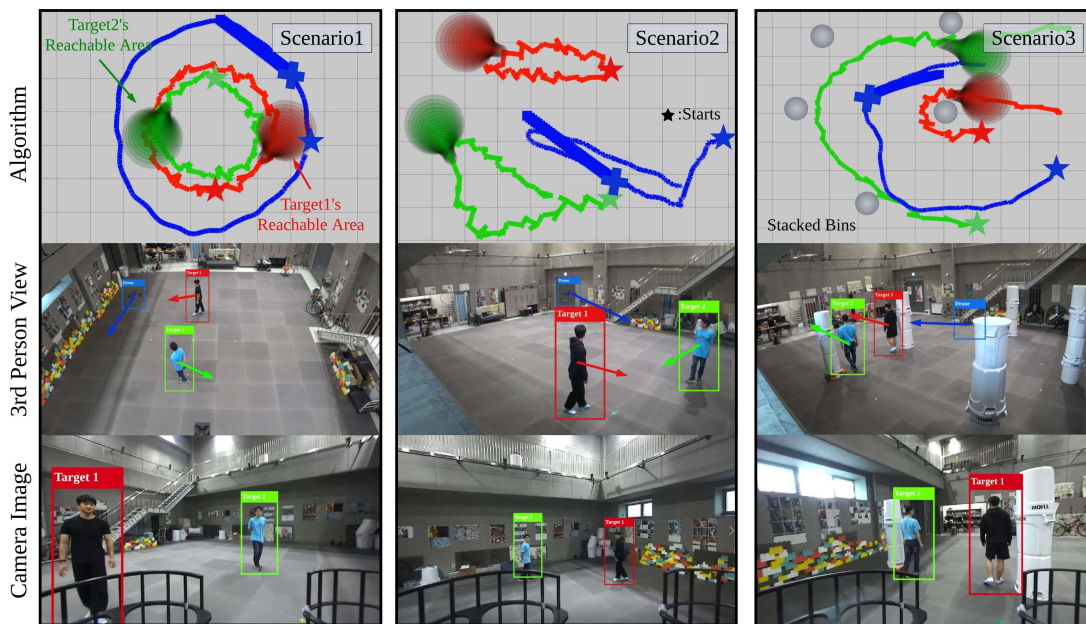


Fig. 14. Autonomous aerial tracking in an indoor environment with dual-target scenarios. Blue, red, and green curves represent reported paths moved by the drone, target 1 and target 2, respectively. Blue crosses mean the position of the drone at captured moments. The short, thick blue splines, which start from blue crosses, represent the generated trajectory for 1.5 seconds.

E. Adaptability, Scalability, and Practicality

As shown in the validations, our planner can be used in environments such as forests or crowds, where obstacles can be modeled as cylinders. Extension to environments with ellipsoidal objects is manageable, but the QP-based approach that can handle unstructured obstacles should be studied further. Extension to multi-target chasing is also feasible. As in dual-target missions, we apply occlusion avoidance and FOV constraints to all pairs of targets, but the reference

trajectory to keep high visibility of all targets needs further study.

In real-world experiments, we confirmed that despite the path prediction error, the drone successfully tracked the targets by quickly updating its chasing trajectory. However, to handle greater sensor noise and more severe unexpected target behaviors, not only is a fast planning algorithm required, but also greater maneuverability of the drone. Two cameras used for stable localization and two onboard computers to meet

TABLE IX
REPORTED PERFORMANCE IN EXPERIMENTS

Metrics	Single Target				Dual Target		
	Scenario S1	Scenario S2	Scenario S3	Scenario S4	Scenario D1	Scenario D2	Scenario D3
Target Distance (45a) [m] (†)	1.668/2.303	1.301/2.153	1.793/2.382	0.805/1.739	1.336/2.037	1.061/3.349	1.567/2.358
Obstacle Distance (45b) [m] (†)	0.534/2.289	0.701/2.285	1.405/2.109	0.545/1.862	2.316/2.877	1.094/3.277	0.209/1.889
Visibility Score (45c) [m] (†)	0.393/2.111	1.020/1.678	1.302/1.831	0.765/1.461	2.403/2.809	1.397/3.099	0.534/2.289
MOTA (†)	0.998	0.990	0.987	0.990	0.991	0.844	0.990
IDFI (†)	0.999	0.995	0.995	0.993	0.996	0.990	1.0
Visibility Proportion (46) (†)	1.0/1.0	1.0/1.0	1.0/1.0	1.0/1.0	1.0/1.0	1.0/1.0	0.901/0.995
Computation Time [ms]	12.61	14.63	16.12	15.32	14.61	13.94	20.81

The values for the first, second, third, and sixth metrics indicate the minimum/mean performance. † means that a value below 0 indicates a collision or occlusion. † means that higher is better.

the pipeline's real-time constraints significantly increase the overall weight of our system. Reducing the drone's weight will be beneficial for its agility.

VIII. CONCLUSION

We propose a real-time target-chasing planner among static and dynamic obstacles. First, we calculate the reachable areas of moving objects in obstacle environments using Bernstein polynomial primitives. Then, to prevent target occlusion, we define a continuous-time target-visible region (TVR) based on path homotopy while considering the camera field-of-view limit. The reference trajectory for target tracking is designed and utilized with TVR to formulate trajectory optimization as a single QP problem. The proposed QP formulation can generate dynamically feasible, collision-free, and occlusion-free chasing trajectories in real-time. We extensively demonstrate the effectiveness of the proposed planner through challenging scenarios, including realistic simulations and indoor experiments. In the future, we plan to extend our work to chase multiple targets in environments with moving unstructured obstacles.

APPENDIX A

IMPLEMENTATION OF COLLISION CHECK

The collision between the trajectories of the moving object and the j -th obstacle in (14) can be determined using coefficients of Bernstein polynomials.

$$\begin{aligned} & \|\hat{\mathbf{p}}_i(t) - \hat{\mathbf{o}}_j(t)\|_2^2 - (r_{p0} + r_{o_j}(t))^2 \geq 0 \\ \iff & \sum_{k=0}^{2n_p} \binom{2n_p}{k}^{-1} {}^{ij}C_k b_{k,2n_p}(t; 0, T) \geq 0, \quad \forall j \in \{1, \dots, |\mathcal{O}|\}, \end{aligned}$$

where

$$\begin{aligned} {}^{ij}C_k &= \sum_{l=\max(0, k-n_p)}^{\min(k, n_p)} \binom{n_p}{l} \binom{n_p}{k-l} ({}^i_j \mathbf{p}_{(x)}[l] {}^j_l \mathbf{p}_{(x)}[k-l] \\ & \quad + {}^i_j \mathbf{p}_{(y)}[l] {}^j_l \mathbf{p}_{(y)}[k-l] - {}^i_j \mathbf{r}[l] {}^j_l \mathbf{r}[k-l]), \\ {}^i_j \mathbf{p}_{(x)} &= \mathbf{p}_{i(x)} - \mathbf{o}_{j(x)}, \quad {}^i_j \mathbf{p}_{(y)} = \mathbf{p}_{i(y)} - \mathbf{o}_{j(y)}, \quad {}^i_j \mathbf{r} = \mathbf{r}_{o_j} + r_{p0} \end{aligned} \quad (47)$$

$[l]$ represents l -th elements of a vector, and \mathbf{r}_{o_j} is the Bernstein coefficient representing the radius of the j -th obstacle. The condition ${}^{ij}C_k \geq 0$, $k = 0, \dots, 2n_p$ makes the moving objects not collide with obstacles during $[0, T]$.

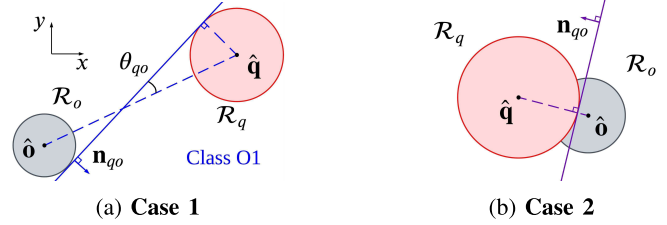


Fig. 15. TVR-O Formulation.

APPENDIX B

PROOF OF THE PROPOSITION 1

Motion primitives for object prediction derived from (11) are represented as follows.

$$\hat{\mathbf{p}}_i(t) = \mathbf{P}_{i,n_p}^\top \mathbf{b}_{n_p}(t, T)$$

where $\mathbf{P}_{i,n_p} = U_{3,n_p}^\top \mathbf{P}_{i,3}^*$,

$$\begin{aligned} \mathbf{P}_3^* &= \left[\hat{\mathbf{p}}_0, \hat{\mathbf{p}}_0 + \frac{1}{3} \hat{\mathbf{p}}_0' T, \frac{2}{3} \hat{\mathbf{p}}_0 + \frac{1}{3} \mathbf{s}_{p,i} + \frac{1}{3} \hat{\mathbf{p}}_0' T, \mathbf{s}_{p,i} \right]^\top, \\ \mathbf{b}_{n_p}(t, T) &= [b_{0,n_p}(t, 0, T), \dots, b_{n_p,n_p}(t, 0, T)]^\top, \quad n_p \geq 3, \\ U_{m,n} &= \{u_{j,k}\} \in \mathbb{R}^{(n+1) \times (m+1)}, \quad m \geq n, \\ u_{j,j+k} &= \frac{\binom{n}{j} \binom{m-n}{k}}{\binom{m}{j+k}} \end{aligned} \quad (48)$$

The difference between $\hat{\mathbf{p}}_i(t)$ and $\hat{\mathbf{p}}_j(t)$, $\|\hat{\mathbf{p}}_i(t) - \hat{\mathbf{p}}_j(t)\| = \|\mathbf{s}_{p,i} - \mathbf{s}_{p,j}\| \frac{t^2}{T^2}$. Therefore, in order to minimize distance sum in (16), it is sufficient to investigate sampled endpoints $\mathbf{s}_{p,i}$.

APPENDIX C

TVR FORMULATION

We derive TVR-O, which are represented as (20) and (21), and TVR-F, which are represented as (27). For the mathematical simplicity, we omit time t to represent variables.

A. TVR-O Formulation

As mentioned in Section. VI-B, there are two cases where the reachable areas of the target and obstacles **Case 1**: do not overlap and **Case 2**: overlap.

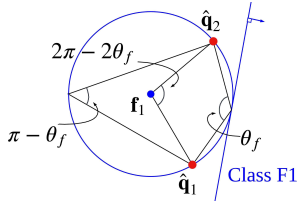


Fig. 16. TVR-F Formulation.

1) *Case 1: Non-Overlap*: Fig. 15a shows the TVR-O, when the relation between the target, the obstacle, and the drone corresponds to *Class O1*. TVR-O is made by the tangential line, and its normal vector is represented as \mathbf{n}_{qo} . TVR-O is represented as follows:

$$\mathbf{n}_{qo}^\top (\mathbf{x} - \hat{\mathbf{o}}) \geq r_o \quad (49)$$

\mathbf{n}_{qo} is perpendicular to the tangential line, and the line is rotated by θ_{qo} from a segment connecting the centers of the reachable areas. By using rotation matrices, \mathbf{n}_{qo} becomes as follows.

$$\mathbf{n}_{qo} = R\left(-\frac{\pi}{2}\right) R(\theta_{qo}) \frac{\hat{\mathbf{q}} - \hat{\mathbf{o}}}{\|\hat{\mathbf{q}} - \hat{\mathbf{o}}\|}, \quad (50a)$$

$$\sin \theta_{qo} = \frac{r_q + r_o}{\|\hat{\mathbf{q}} - \hat{\mathbf{o}}\|}, \quad \cos \theta_{qo} = \sqrt{1 - \sin^2 \theta_{qo}} \quad (50b)$$

By multiplying $\|\hat{\mathbf{q}} - \hat{\mathbf{o}}\|_2^2$ to (49), we can acquire (20).

2) *Case 2: Overlap*: Fig. 15b shows the TVR-O, when the reachable areas overlap. TVR-O is made by the tangential line that is perpendicular to the segment connecting the centers of the reachable areas. TVR-O is represented as follows:

$$\mathbf{n}_{qo}^\top (\mathbf{x} - \hat{\mathbf{q}}) + r_q \geq 0, \quad \mathbf{n}_{qo} = \frac{\hat{\mathbf{q}} - \hat{\mathbf{o}}}{\|\hat{\mathbf{q}} - \hat{\mathbf{o}}\|} \quad (51)$$

(51) is equivalent to (21).

B. TVR-F Formulation

Fig. 16 shows the TVR-F, when the relation between the targets and the drone corresponds to *Class F1*. The blue circle has inscribed angles of an arc tracing two points at $\hat{\mathbf{q}}_1$ and $\hat{\mathbf{q}}_2$ equals camera FOV θ_f . The position of its center, \mathbf{f}_1 , is obtained by a relation: rotation of the segment connecting \mathbf{f}_1 and $\hat{\mathbf{q}}_1$ by an angle $2\pi - 2\theta_f$ becomes the segment connecting \mathbf{f}_1 and $\hat{\mathbf{q}}_2$.

$$R(2\pi - 2\theta_f)(\hat{\mathbf{q}}_1 - \mathbf{f}_1) = \hat{\mathbf{q}}_2 - \mathbf{f}_1 \quad (52)$$

The angle $2\pi - 2\theta_f$ comes from the property of the inscribed angle. The radius of the circle, r_f , is equal to the distance between the points at $\hat{\mathbf{q}}_1$ and \mathbf{f}_1 .

$$\begin{aligned} r_f &= \|\mathbf{f}_1 - \hat{\mathbf{q}}_1\| = \frac{1}{2} \left\| \begin{bmatrix} 1 & \cot \theta_f \\ -\cot \theta_f & 1 \end{bmatrix} (\hat{\mathbf{q}}_2 - \hat{\mathbf{q}}_1) \right\| \\ &= \frac{1}{2 \sin \theta_f} \|\hat{\mathbf{q}}_1 - \hat{\mathbf{q}}_2\| \end{aligned} \quad (53)$$

TVR-F is made by the tangential line that is parallel to the segments connecting the two targets, and is represented as follows:

$$(\mathbf{x} - \mathbf{f}_1)^\top R\left(-\frac{\pi}{2}\right) \frac{\hat{\mathbf{q}}_2 - \hat{\mathbf{q}}_1}{\|\hat{\mathbf{q}}_2 - \hat{\mathbf{q}}_1\|} \geq r_f \quad (54)$$

By substituting \mathbf{f}_1 and r_f in (54) with terms represented by $\hat{\mathbf{q}}_1$ and $\hat{\mathbf{q}}_2$, we can acquire (27).

REFERENCES

- [1] M. Aranda, G. López-Nicolás, C. Sagüés, and Y. Mezouar, "Formation control of mobile robots using multiple aerial cameras," *IEEE Trans. Robot.*, vol. 31, no. 4, pp. 1064–1071, Aug. 2015.
- [2] A. Alcántara, J. Capitán, R. Cunha, and A. Ollero, "Optimal trajectory planning for cinematography with multiple unmanned aerial vehicles," *Robot. Autom. Syst.*, vol. 140, Jun. 2021, Art. no. 103778.
- [3] B. Penin, P. R. Giordano, and F. Chaumette, "Vision-based reactive planning for aggressive target tracking while avoiding collisions and occlusions," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 3725–3732, Oct. 2018.
- [4] H. Huang, A. V. Savkin, and W. Ni, "Online UAV trajectory planning for covert video surveillance of mobile targets," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 2, pp. 735–746, Apr. 2022.
- [5] R. Bonatti et al., "Autonomous aerial cinematography in unstructured environments with learned artistic decision-making," *J. Field Robot.*, vol. 37, no. 4, pp. 606–641, Jun. 2020.
- [6] Q. Yang and H. Li, "RMPC-based visual servoing for trajectory tracking of quadrotor UAVs with visibility constraints," *IEEE/CAA J. Autom. Sinica*, vol. 11, no. 9, pp. 2027–2029, Sep. 2024.
- [7] D. Kim, M. Pezzutto, L. Schenato, and H. J. Kim, "Visibility-constrained control of multirotor via reference governor," in *Proc. 62nd IEEE Conf. Decis. Control (CDC)*, Dec. 2023, pp. 5714–5721.
- [8] X. Yi, H. Liu, Y. Wang, H. Duan, and K. P. Valavanis, "Safe reinforcement learning-based visual servoing control for quadrotors tracking unknown ground vehicles," *IEEE Trans. Intell. Vehicles*, vol. 10, no. 6, pp. 3803–3813, Jun. 2025.
- [9] X. Wu, H. Wang, and A. K. Katsaggelos, "Automatic camera movement generation with enhanced immersion for virtual cinematography," *IEEE Trans. Multimedia*, vol. 27, pp. 5241–5254, 2025.
- [10] J. Chen, B. He, C. D. Singh, C. Fermüller, and Y. Aloimonos, "Active human pose estimation via an autonomous UAV agent," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2024, pp. 7801–7808.
- [11] S. Cui, Y. Chen, and X. Li, "A robust and efficient UAV path planning approach for tracking agile targets in complex environments," *Machines*, vol. 10, no. 10, p. 931, Oct. 2022. [Online]. Available: <https://www.mdpi.com/2075-1702/10/10/931>
- [12] T. Nägeli, J. Alonso-Mora, A. Domahidi, D. Rus, and O. Hilliges, "Real-time motion planning for aerial videography with dynamic obstacle avoidance and viewpoint optimization," *IEEE Robot. Autom. Lett.*, vol. 2, no. 3, pp. 1696–1703, Jul. 2017.
- [13] H. Masnavi, J. Shrestha, M. Mishra, P. B. Sujit, K. Kruusamäe, and A. K. Singh, "Visibility-aware navigation with batch projection augmented cross-entropy method over a learned occlusion cost," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 9366–9373, Oct. 2022.
- [14] H. Masnavi, V. K. Adajania, K. Kruusamäe, and A. K. Singh, "Real-time multi-convex model predictive control for occlusion-free target tracking with quadrotors," *IEEE Access*, vol. 10, pp. 29009–29031, 2022.
- [15] B. F. Jeon, C. Kim, H. Shin, and H. J. Kim, "Aerial chasing of a dynamic target in complex environments," *Int. J. Control, Autom. Syst.*, vol. 20, no. 6, pp. 2032–2042, Jun. 2022.
- [16] B. F. Jeon, Y. Lee, J. Choi, J. Park, and H. J. Kim, "Autonomous aerial dual-target following among obstacles," *IEEE Access*, vol. 9, pp. 143104–143120, 2021.
- [17] P. Pueyo, J. Dendarieta, E. Montijano, A. C. Murillo, and M. Schwager, "CineMPC: A fully autonomous drone cinematography system incorporating zoom, focus, pose, and scene composition," *IEEE Trans. Robot.*, vol. 40, pp. 1740–1757, 2024.
- [18] H. Wang, X. Zhang, Y. Liu, X. Zhang, and Y. Zhuang, "SVPTO: Safe visibility-guided perception-aware trajectory optimization for aerial tracking," *IEEE Trans. Ind. Electron.*, vol. 71, no. 3, pp. 2716–2725, Mar. 2024.
- [19] J. Ji, N. Pan, C. Xu, and F. Gao, "Elastic tracker: A spatio-temporal trajectory planner flexible aerial tracking," 2021, *arXiv:2109.07111*.
- [20] Z. Zhang, Y. Zhong, J. Guo, Q. Wang, C. Xu, and F. Gao, "Auto filmer: Autonomous aerial videography under human interaction," *IEEE Robot. Autom. Lett.*, vol. 8, no. 2, pp. 784–791, Feb. 2023.
- [21] Z. Han, R. Zhang, N. Pan, C. Xu, and F. Gao, "Fast-tracker: A robust aerial system for tracking agile target in cluttered environments," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 328–334.

- [22] N. Pan, R. Zhang, T. Yang, C. Cui, C. Xu, and F. Gao, "Fast-tracker 2.0: Improving autonomy of aerial tracking with active vision and human location regression," *IET Cyber-Syst. Robot.*, vol. 3, no. 4, pp. 292–301, Dec. 2021.
- [23] F. Yang, Q. Lu, N. Huang, B. Zhang, and Y. Choi, "Target tracking control of an autonomous aerial vehicle in unknown environments," *IEEE Trans. Ind. Informat.*, vol. 21, no. 6, pp. 4377–4387, Jun. 2025.
- [24] B. Jeon, Y. Lee, and H. J. Kim, "Integrated motion planner for real-time aerial videography with a drone in a dense environment," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 1243–1249.
- [25] N. R. Gans, G. Hu, K. Nagarajan, and W. E. Dixon, "Keeping multiple moving targets in the field of view of a mobile camera," *IEEE Trans. Robot.*, vol. 27, no. 4, pp. 822–828, Aug. 2011.
- [26] M. Zarudzki, H.-S. Shin, and C.-H. Lee, "An image based visual servoing approach for multi-target tracking using an quad-tilt rotor UAV," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Jun. 2017, pp. 781–790.
- [27] J. Chen, T. Liu, and S. Shen, "Tracking a moving target in cluttered environments using a quadrotor," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 446–453.
- [28] S. Bhattacharya, M. Likhachev, and V. Kumar, "Topological constraints in search-based robot path planning," *Auto. Robots*, vol. 33, no. 3, pp. 273–290, Oct. 2012.
- [29] J. R. Munkres, *Topology Prentice Hall*. Upper Saddle River, NJ, USA, 2000.
- [30] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. IEEE Int. Conf. Robot. Autom.*, Shanghai, China, May 2011, pp. 2520–2525.
- [31] D. Popa, F. L. Lewis, and L. Xie, *Optimal and Robust Estimation: With an Introduction to Stochastic Control Theory, Second Edition*. Boca Raton, FL, USA: CRC Press, 2008.
- [32] G. W. Collins, *Fundamental Numerical Methods and Data Analysis*, 2003. [Online]. Available: <http://bifrost.cwru.edu/personal/collins/numbk/Bookfmt.pdf>
- [33] A. Marco and J.-J. Martínez, "A fast and accurate algorithm for solving Bernstein–Vandermonde linear systems," *Linear Algebra Appl.*, vol. 422, nos. 2–3, pp. 616–628, Apr. 2007.
- [34] B. F. Jeon and H. J. Kim, "Online trajectory generation of a MAV for chasing a moving target in 3D dense environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 1115–1121.
- [35] C. Kielas-Jensen and V. Cichella, "BeBOT: Bernstein polynomial toolkit for trajectory generation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 3288–3293.
- [36] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "QpOASES: A parametric active-set algorithm for quadratic programming," *Math. Program. Comput.*, vol. 6, no. 4, pp. 327–363, Dec. 2014.
- [37] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "AirSim: High-fidelity visual and physical simulation for autonomous vehicles," in *Proc. Field Service Robot.*, 2017, pp. 621–635.
- [38] M. Przybyla, "Detection and tracking of 2D geometric obstacles from LRF data," in *Proc. 11th Int. Workshop Robot Motion Control (RoMoCo)*, Jul. 2017, pp. 135–141.
- [39] R. Khanam and M. Hussain, "YOLOv11: An overview of the key architectural enhancements," 2024, *arXiv:2410.17725*.



Yunwoo Lee received the B.S. degree in electrical and computer engineering and the Ph.D. degree in mechanical and aerospace engineering from Seoul National University, South Korea, in 2019 and 2025, respectively. He is currently a Post-Doctoral Fellow at the Robotics Institute, Carnegie Mellon University, USA, and previously worked at the Artificial Intelligence Institute, Seoul National University. His research interests include aerial tracking and multi-robot systems.



top prize at the 2022 KAI Aerospace Paper Award.

Jungwon Park received the B.S. degree in electrical and computer engineering and the M.S. and Ph.D. degrees in mechanical and aerospace engineering from Seoul National University, South Korea, in 2018, 2020, and 2023, respectively. He is currently an Assistant Professor with Seoul National University of Science and Technology, South Korea. His current research interests include path planning and task allocation for distributed multi-robot systems. His work was a finalist for the Best Paper Award in Multi-Robot Systems at ICRA 2020 and won the



Seungwoo Jung received the B.S. degree in mechanical engineering and artificial intelligence from Korea University, South Korea, in 2021. He is currently pursuing an integrated M.S./Ph.D. degrees in aerospace engineering with the Laboratory for Autonomous Robotics Research (LARR), Korea University. His research interests include learning based planning and control for autonomous vehicle systems.



Boseong Jeon received the B.S. degree in mechanical engineering and the Ph.D. degree in aerospace engineering from Seoul National University, Seoul, South Korea, in 2017 and 2022, respectively. He is currently a Researcher with Samsung Research, South Korea. His research interests include VLA and planning.



Dahyun Oh received the B.S. degree in mechanical engineering from Korea University, South Korea, in 2021. He is currently pursuing the Ph.D. degree in aerospace engineering with the Laboratory for Autonomous Robotics Research, under the supervision of H. Jin Kim. He is also a member with the Laboratory for Autonomous Robotics Research. His current research interests include reinforcement learning for multi-agents.



where she is currently a Professor. Her research interests include navigation and motion planning of autonomous robotic systems.

H. Jin Kim received the B.S. degree from Korean Advanced Institute of Technology, Daejeon, South Korea, in 1995, and the M.S. and Ph.D. degrees from the University of California, Berkeley, Berkeley, CA, USA, in 1999 and 2001, respectively, all in mechanical engineering. From 2002 to 2004, she was a Post-Doctoral Researcher with the Department of Electrical Engineering and Computer Science, University of California, Berkeley. In 2004, she joined the School of Mechanical and Aerospace Engineering, Seoul National University, Seoul, South Korea,