IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 12, 2021, accepted September 9, 2021, date of publication October 4, 2021, date of current version October 26, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3117314

Autonomous Aerial Dual-Target Following Among Obstacles

BOSEONG FELIPE JEON, YUNWOO LEE, JEONGJUN CHOI, (Graduate Student Member, IEEE), JUNGWON PARK, (Graduate Student Member, IEEE), AND H. JIN KIM, (Member, IEEE)
School of Mechanical and Aerospace Engineering, Seoul National University, Seoul 08826, South Korea

Corresponding author: H. Jin Kim (hjinkim@snu.ac.kr)

This work was supported by the Ministry of Trade, Industry and Energy (MOTIE), South Korea, through the Industrial Technology Innovation Program "Development of disaster response robot system for lifesaving and supporting fire fighters at complex disaster environment" under Grant 10067206.

ABSTRACT In contrast to recent developments in online motion planning to follow a single target with a drone among obstacles, a multi-target case with a single chaser drone has been hardly discussed in similar settings. Following more than one target is challenged by multiple visibility issues due to the inter-target occlusion and the limited field-of-view in addition to the possible occlusion and collision with obstacles. Also, reflecting multiple targets into planning objectives or constraints increases the computation load and numerical issues in the optimization compared to the single target case. To resolve the issues, we first develop a visibility score field for multiple targets incorporating the field-of-view limit and inter-occlusion between targets. Next, we develop a fast sweeping algorithm used to compute the field for the suitability of real-time applications. Last, we build an efficient hierarchical planning pipeline to output a chasing motion for multiple targets ensuring key objectives and constraints. For reliable chasing, we also present a prediction algorithm to forecast the movement of targets considering obstacles. The online performance of the proposed algorithm is extensively validated in challenging scenarios, including a large-scale simulation, and multiple real-world experiments in indoor and outdoor scenes. The full code implementation of the proposed method is released here: https://github.com/icsl-Jeon/dual_chaser.

INDEX TERMS Collision avoidance, cinematography, motion planning, trajectory optimization.

I. INTRODUCTION

Enhanced autonomy thanks to the advances in localization [1], [2], tracking [3] and control [4] has widened the utilization of drones equipped with a camera for vision-based tasks such as videography and surveillance. Achieving autonomy for those missions requires a reliable chasing planner reflecting obstacles. Regarding the motion generation, a group of recent works has suggested real-time planning algorithms [5]–[10]. They considered the key motion objectives such as the actuation efficiency of the chaser drone and the desired relative distance between the drone and the target. Safety of the chaser drone and visibility of the target were also handled against obstacles based on non-convex optimization [5], [6] or search-based methods [8], [9].

Still, there are various scenarios that can benefit from the capability of a *single* drone to follow *multiple* targets among obstacles and capture them in a single image frame (called *multi-shot* in cinematography [11]). For example, in a filming scene, a drone should capture multiple actors in a single camera view to express the relationship between actors. Also, sports events such as a marathon can deploy a camera drone to broadcast multiple players. Concerning the multi-target scenario among obstacles, the previous works [5]–[10] are not readily applicable due to the following reasons. First, in addition to the occlusion from obstacles, we should consider the visibility issues coming from inter-occlusion between targets and the field-of-view (FOV) limit when capturing all the targets. Also, the real-time computation becomes more

The associate editor coordinating the review of this manuscript and approving it for publication was Pedro Neto.

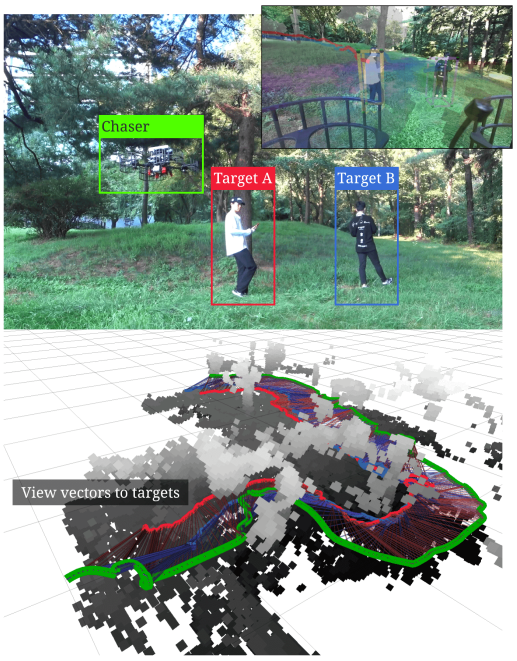

FIGURE 1. Outdoor experiment where a drone autonomously follows two targets in a forest environment. We implemented the fully onboard system, including target detection, prediction, obstacle mapping, planning and flight control. Top: Experiment scene and the onboard camera view of the drone. The camera view is overlaid with 3D detection boxes and a 2D slice of the score field that measures the visibility of the two targets. Bottom: Accumulated mapping from the onboard sensor and the history of the two targets (red and blue) and the drone (green). The thin red and blue arrows illustrate view vectors toward two targets from the chaser.

challenging due to the increased number of targets. The increased number of targets can also attribute to numerical issues in optimization, thus the increased conflicts between objectives (e.g. increasing the visibility of one target might degrade the others) if a single non-convex optimization is adopted as [5]–[7] did.

In both academic research and commercial products (e.g. DJI [12] or Skydio [13]), chasing multiple targets with a single drone has not been discussed as much as the single target case, especially in obstacle environments. The relevant works [14], [15] produced the motion of observing camera which minimizes the variance of projected target locations on the camera image. The authors of [16] considered the sports filming where a drone is directed to focus the areas of attention using context information. The work [17] included the principles of the cinematography (e.g. rule of the third) for a visually-pleasing video footage. Although the research [14]–[17] considered the multi-target scenario with a single camera setup, they did not address obstacle environments. Also, the practical considerations such as the inter-occlusion, movement efficiency of the chasing agent, and the desired relative distance to the targets were not discussed in an integrated manner.

A. TECHNICAL CHALLENGES

Here, we mention the major technical issues in the context of the motion planning, when building an autonomous system to capture multiple targets on a single camera frame without priors of the environment and future movements of the targets.

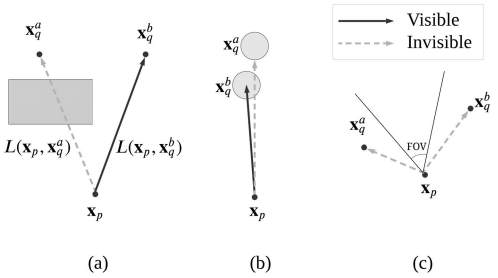

FIGURE 2. Infeasible cases when observing two targets with a single chaser. The targets' positions are written as $\mathbf{x}_q^a$ and $\mathbf{x}_q^b$ respectively. Chaser drone is denoted as $\mathbf{x}_p$. (a) Both targets should not be occluded from obstacles. (b) Occlusion of one target due to the volume of the other target should not occur. (c) The angle of the two bearing vectors should be smaller than the FOV of the camera of the chaser.

First, the system should be able to reliably forecast the targets' movement as it directly affects the quality of the chasing motion planning. The prediction should appropriately reflect the past motion of the target by minimizing the observation error. In addition, we should consider the presence of the non-traversal region for the applicability to obstacle scenarios. Using the fact that the targets do not intersect with obstacles can increase the accuracy of the prediction. If the targets are erroneously predicted to be inside the obstacles, the chasing planner might not reliably evaluate the visibility of the target.

Second, computing the chasing motion of the drone should handle multiple objectives jointly. The smoothness of the motion should be taken into account to reduce the input actuation and blur of the image of the drone. For a better shot-quality and the detection performance, the relative distance between the targets and the drone should be maintained at a desired level. Also, the collision of the drone with obstacles or the targets should be avoided. Notably, capturing two targets simultaneously should resolve the three visibility issues: occlusion from obstacles, inter-occlusion, and the FOV limit of the camera sensor observing the targets (see Fig. 2). Incorporating the considerations, moreover, the motion planning algorithm should be computed fast enough to respond to sudden motion changes in the multiple targets and the newly discovered obstacles.

B. CONTRIBUTIONS

This paper aims to build an efficient online system for a single drone to chase two targets and capture them in a single frame, which resolves the aforementioned issues in an integrated manner. The proposed system develops two modules: prediction and chasing.

The prediction module adopts an offline library containing candidate future trajectories of the targets. Among the candidates, the prediction is chosen as the motion having the minimum observation error and avoiding the intersection with non-traversable region (obstacles). We discuss the target forecasting algorithm and its performance in Section. III.

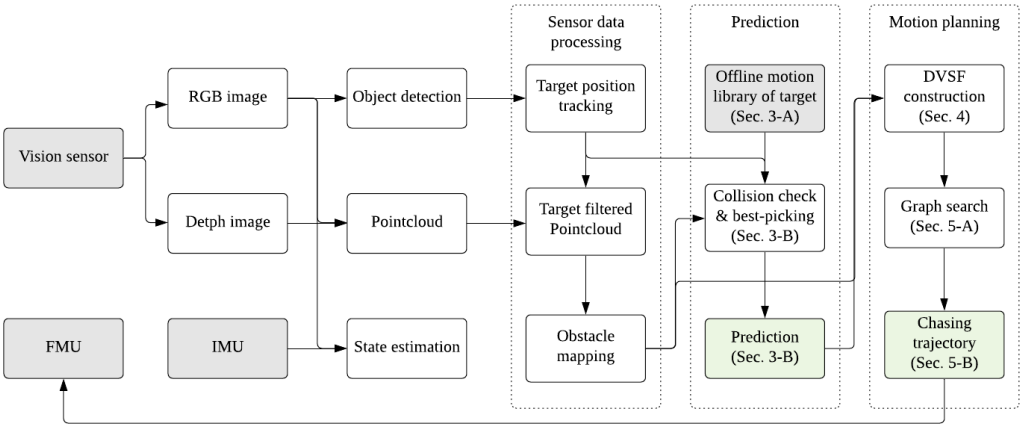

FIGURE 3. Presented online chasing system. The whole system including target estimation, mapping, planning, and control runs fully onboard. The code implementations of sensor data processing, prediction, and motion planning can be found https://github.com/icsl-Jeon/zed2_client, https://github.com/icsl-Jeon/chasing_utils, and https://github.com/icsl-Jeon/dual_chaser, respectively.

Regarding the chasing motion of the drone, the presented planning module is composed of three steps: 1) computation of the visibility score map reflecting two targets, 2) viewpoint construction, and 3) generation of chasing corridors and smooth trajectory. The first phase is explained in Section. IV where we propose a visibility score field for two targets incorporating the inter-occlusion and the FOV limit (see Fig. 2). Also, we introduce an efficient sweeping algorithm for the computation of the field, achieving ten times faster computation than our previous work [8]. Based on the score field, Section. V reveals the second and third phases. The second phase uses a search algorithm on a topologically sorted graph to compute a sequence of optimal viewpoints ensuring the key objectives and constraints. Using the viewpoints and a linearly-constrained quadratic programming (QP), the last phase outputs a smooth chasing motion in a numerically stable way. In Section. VI, we validate the online performance of the proposed pipeline with a high-fidelity simulation and multiple real-world experiments in four different scenes.

By proposing the mentioned online pipeline for the dual-target chasing among obstacles, our contributions are summarized as follows:

- We propose a fast prediction method for the targets with consideration of the observation error and obstacles.
- We propose a metric to encode the visibility for dual-target, taking into the FOV limit of the drone and inter-occlusion between the targets.
- We propose an efficient algorithm to compute the visibility score field using recursion and interpolation.
- We build and validate a hierarchical motion pipeline to chase two targets with a large-scale simulation and multiple real-world experiments including indoor and outdoor scenes, where the targets move in three dimensions.

To the authors' best knowledge, this work is the first to address the chasing of multiple targets using a single drone in dense obstacles. Although the proposed method can be extended to targets with more than two, this paper considers only two targets, as it is extremely difficult for a standard sensor setup to find a feasible solution which resolves all the technical issues for three or more targets.

C. ONBOARD SYSTEM OVERVIEW

Here, we briefly overview the proposed dual-target chasing system. The drone is equipped with an inertial measurement unit (IMU) and a vision sensor from which we can obtain RGB and depth images (e.g. a stereo camera). Using the IMU and the vision sensor output, visual odometry runs to estimate the state of the drone. We detect the targets from the RGB image and estimate their locations based on the depth image. We also use depth images to build the pointcloud filtering out the points on the targets. The remaining pointcloud is used for the obstacle mapping.

For the prediction of the targets, an offline motion library is built with the targets' candidate future trajectories. Also, the library associates each trajectory with the region the trajectory passes. Using observed targets' locations and the library, we obtain the candidate future movements. Then, based on the obstacle mapping, we test the feasibility of each candidate in terms of the collision with obstacles and select an optimal prediction concerning the observation error.

For a set of discrete points along the prediction, we now build a sequence of visibility score fields measuring how much a location can observe the targets in terms of the key objectives in Section. I-A. After spawning a set of candidate viewpoints on the fields, a graph search computes an optimal sequence of viewpoints satisfying essential constraints. Next, we obtain the corridors connecting the viewpoints, inside which the safety of the chaser is guaranteed. Last, a smooth chasing motion for the drone is planned inside the corridors, and the controller of the flight management unit (FMU) of the chaser drone executes the motion until the prediction is reliable. We replan the chasing trajectory when the planned

motion or the prediction becomes infeasible due to newly discovered obstacles.

II. PRELIMINARY

In this section, we overview the measures which quantify the safety of a location and its visibility toward a target against obstacles, based on our previous work [8]. Next, as a newly introduced concept in this paper, we define the visibility of a chaser position toward two targets considering the inter-occlusion and the FOV limit. Then, we state the assumptions and the problem to be solved in this paper.

A. SCORING FIELDS FOR SAFETY AND VISIBILITY

First, let us assume that the local map information can be represented with a 3D occupancy grid using a mapping framework such as Octomap [18]. Based on the grid, we quantify how much a position is safe against obstacles using Euclidean distance field (EDF) [19]. For a position $\mathbf{x} \in \mathbb{R}^3$, we measure the safety with the value of EDF $\phi(\mathbf{x}) \geq 0$ where $\phi(\mathbf{x}) = 0$ corresponds to the collision with obstacles and the positive value encodes the distance from the occupied cells. From this, $\mathbf{x}$ with a larger $\phi(\mathbf{x})$ is measured to be safer.

Next, we quantify how visible a target's position $\mathbf{x}_q \in \mathbb{R}^3$ is from a chaser position $\mathbf{x}_p \in \mathbb{R}^3$ based on the field $\phi(\mathbf{x})$. Letting $L(\mathbf{x}_p, \mathbf{x}_q) \subset \mathbb{R}^3$ denote the set of points on the line-of-sight (LOS) connecting $\mathbf{x}_p$ and $\mathbf{x}_q$, the visibility of $\mathbf{x}_p$ is defined as $\psi(\mathbf{x}_p; \mathbf{x}_q) \in \mathbb{R}$ based on the EDF value $\phi(\mathbf{x})$:

$$\psi(\mathbf{x}_p; \mathbf{x}_q) = \min_{\mathbf{x} \in L(\mathbf{x}_p, \mathbf{x}_q)} \phi(\mathbf{x}) \tag{1}$$

As discussed in our previous work [8], a larger value of $\psi(\mathbf{x}_p; \mathbf{x}_q)$ is interpreted to be more robust against the unknown movement of $\mathbf{x}_q$. Also, $\psi(\mathbf{x}_p; \mathbf{x}_q) = 0$ indicates the occlusion of $\mathbf{x}_q$. We call the grid field where $\psi(\mathbf{x}; \mathbf{x}_q)$ is evaluated for all the centers of the cells $\mathbf{x} \in \mathbb{R}^3$ as the visibility score field (VSF) of $\mathbf{x}_q$.

B. VISIBILITY OF TWO TARGETS

The previous section discussed the visibility score of a single target. Here, we state the necessary conditions to observe two targets $\mathbf{x}_q^a \in \mathbb{R}^3$ and $\mathbf{x}_q^b \in \mathbb{R}^3$ from a chaser position $\mathbf{x}_p$. The conditions introduced are the major difference from the single-target problem. For both targets to be visible simultaneously, the following three conditions should be first satisfied as illustrated in Fig. 2:

Condition 1: Necessary conditions for the visibility of dual-target

(a) None of $L(\mathbf{x}_p, \mathbf{x}_q^a)$ and $L(\mathbf{x}_p, \mathbf{x}_q^b)$ intersects with obstacles.
(b) The LOS toward one target does not intersect with the volume of the other target.
(c) The bearing angle formed by $L(\mathbf{x}_p, \mathbf{x}_q^a)$ and $L(\mathbf{x}_p, \mathbf{x}_q^b)$ is smaller than FOV limit.

C. PROBLEM SETUP

We set up the assumptions and the problem to be solved. In this work, the chaser is assumed to have a standard vision sensor with a FOV of less than 120° for observing the targets. The maximum acceleration of the chaser is assumed to be larger than the targets. The two targets are assumed to be visible when **Condition 1** is satisfied and the optical axis of the drone heads to the center of the targets $(\mathbf{x}_q^a + \mathbf{x}_q^b)/2$. For the targets of interest, they are allowed to move in three dimensions. To focus on the scope of this paper, let us assume that the targets of interest can be detected reasonably in the image, adopting algorithms such as [3], [20] when the targets are observed within a visible range. Also, we assume that the location of the targets can be estimated by combining the depth image and detection output from the RGB image. The future movements of them are unknown to the chasing drone, and we actively forecast the targets ensuring the prediction not to intersect with obstacles. Regarding the obstacles, we address 3D unstructured objects without presuming their shape or height. The obstacles are unknown initially and discovered with the depth sensor of the drone.

On top of the settings, we aim to build an online receding horizon planner (RHP) for a single drone to chase the dual-target, ensuring its own safety and the visibility conditions **Condition 1** along with key objectives in Section. I-A.

III. FORECASTING TARGET MOVEMENT

This section proposes a library-based forecasting algorithm, incorporating the obstacles and the past observations. After describing the motivations, we discuss the construction of the offline library and the online prediction process.

Most of the recent works on the chasing task [5]–[7] [9], [10] adopted a prediction algorithm which considers only minimization of the past observation error without reflecting the presence of obstacles. Although our previous work [8] introduced the prediction algorithm considering obstacles, via-points of the target and the obstacles should be known a priori, which has limited online applications. Also, the proposed prediction could not guarantee the non-intersection with obstacles because the collision avoidance was included as a cost of a non-convex optimization. In addition, the gradient computation involving a distance field was not efficient for a fine discretization level, and the iteration until a reliable solution was uncertain.

As an effort to extend the online applicability of our prediction algorithm, the recent works [21], [22] on the motion planning to reach a static goal are noteworthy. They consider feasible motion candidates of the drone in an offline library and pre-compute the region which each candidate passes through. These methods do not require the gradient computation of the non-convex objectives, and the online collision checking for a candidate takes only tens of microseconds. They achieved a high-speed flight of a drone in unknown environments represented with voxels. Inspired by the works, we predict the target motion with the following two steps: first, we generate an offline library which has a candidate primitives of the future motion of the targets. Also, the library encodes the relation of each candidate and a set of voxels it passes, given 3D discretization of the local space.

Second, based on the library, we perform online collision checking for the prediction candidates against the newly mapped obstacles. Last, the prediction is selected with a candidate having the lowest error on the past observations.

A. OFFLINE LIBRARY FOR TARGET MOTION

We explain the construction of the offline library to predict a single target $\mathbf{x}_q(t)$ for a considered horizon $0 < t \leq T$ (superscripts a and b are omitted here). The offline target library consists of motion primitives representing possible future trajectories and a set of voxels encompassing the entire primitives as shown in Fig. 4-(a). In the library, the target is assumed to move at a constant acceleration for the horizon. The initial position of the target is set to the origin of the reference frame and the direction of the initial velocity is aligned to $+x$ axis of the frame. For the horizon $(0, T]$, the below model is used to describe the motion of the target:

$$\ddot{\mathbf{x}}_q(t) = \mathbf{a},$$

subject to $\mathbf{x}_q(0) = \mathbf{0}$ and $\dot{\mathbf{x}}_q(0) = \mathbf{v} \in \mathbb{R}^3$ where the element of $\mathbf{v}$ is a non-negative value along the x axis and zeros for other two axes. The constant acceleration is denoted as $\mathbf{a} \in \mathbb{R}^3$. Integrating it leads to the below motion primitive:

$$\mathbf{x}_q(t) = \mathbf{a}\frac{t^2}{2} + \mathbf{v}t.$$

To generate a set of candidate primitives, we sample feasible $\mathbf{a}$ and $\mathbf{v}$ based on lattice discretization similar with [23]. Let us write the set of the primitives $\mathbf{x}_q(t)$ as $\mathcal{L}$. An example of $\mathcal{L}$ is shown in Fig. 4-(a) when the 20 values of $\|\mathbf{v}\|$ were sampled from uniform discretization between [0.1, 1.5] m/s while x and y elements of $\mathbf{a}$ chosen from [0.0, 0.2] m/s^2 (10 samples) and [−0.4, 0.4] m/s^2 (7 samples), respectively. Now, let us consider a set of voxels denoted as $\mathcal{V}$, which encompasses all the generated primitives with a defined resolution (see the transparent cubes in Fig. 4-(a)). In this work, the voxels are set to have the same resolution with the octomap. For a voxel $v \in \mathcal{V}$, we can identify the primitives in $\mathcal{L}$ that intersect with v, and the offline library encodes this information in the form of bitset in a similar manner with [22].

B. COMPUTING FEASIBLE PREDICTION

Here, we describe how to utilize the offline library to output a suitable prediction from online updates on obstacles and observations. Let us define a primitive not intersecting with obstacles to be a *feasible prediction*. The overview for online prediction is as follows: we first transform the motion primitives in $\mathcal{L}$ and voxels in $\mathcal{V}$ with respect to a new reference frame based on the latest observations. Then, we fast check whether each element of the transformed $\mathcal{L}$ intersects with the new obstacles, exploiting the traverse information encoded in the library.

To define a new reference frame of the library when prediction is requested, we assume that N_o observations of the target were gathered over past time steps $\{t_1, t_2, \ldots, t_{N_o}\} \subset \mathbb{R}^-$ and write them as $\{\bar{\mathbf{x}}_{q,1}, \bar{\mathbf{x}}_{q,2}, \ldots, \bar{\mathbf{x}}_{q,N_o}\}$ where $\bar{\mathbf{x}}_{q,n} \in \mathbb{R}^3$

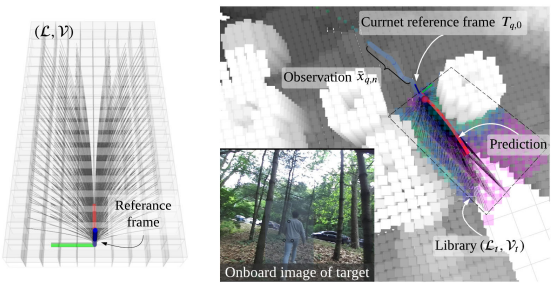

FIGURE 4. (a) An exemplary offline library for the target prediction. Assuming the direction of the velocity is aligned with the reference frame, a set of motion primitives $\mathcal{L}$ is sampled as the candidate future target trajectories. The voxels $\mathcal{V}$ passed by the primitives are also visualized in the transparent cubes. (b) An real-world result of the proposed prediction method. The target is detected, tracked, and predicted, considering the observation error and obstacles. The voxels in the gray scale denotes the inflated obstacles. The rainbow colormap of the cells in the library shows the distance value from the EDF and the set of black curves are the feasible prediction candidates.

$(1 \leq n \leq N_o)$ is the observation at the nth past step, and the larger value of the subscript corresponds to the older observation. For the current state estimation of the target $\bar{\mathbf{x}}_{q,0} \in \mathbb{R}^3$ at the time of prediction $t = 0$, the following pose $T_{q,0} \in SE(3)$ is defined as the new reference frame for transforming the offline library:

$$T_{q,0} = \begin{bmatrix} R_{q,0} & \bar{\mathbf{x}}_{q,0} \\ \mathbf{0} & 1 \end{bmatrix}, \tag{2}$$

where $R_{q,0} = \begin{bmatrix}\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\end{bmatrix} \in SO(3)$ is a rotational matrix with $\mathbf{e}_1 \in \mathbb{R}^3$ aligned with the currently estimated velocity of the target $\mathbf{e}_1 = \dot{\bar{\mathbf{x}}}_{q,0}/\|\dot{\bar{\mathbf{x}}}_{q,0}\|$. Assuming the major rotational motion of the target is yawing and pitching, we set $\mathbf{e}_2 = \begin{bmatrix}e_{21}, e_{22}, 0\end{bmatrix}^T$ subject to $\mathbf{e}_1 \cdot \mathbf{e}_2 = 0$ and $\|\mathbf{e}_2\| = 1$. The last component of $R_{q,0}$ is set with $\mathbf{e}_3 = \mathbf{e}_1 \times \mathbf{e}_2 \in \mathbb{R}^3$. To obtain possible subsequent motions from the current target observation $(\bar{\mathbf{x}}_{q,0}, \dot{\bar{\mathbf{x}}}_{q,0})$, we transform the set of primitives $\mathcal{L}$ and encompassing voxels $\mathcal{V}$ with respect to the frame $T_{q,0}$. We denote the transformed library as $(\mathcal{L}_t, \mathcal{V}_t)$. An example of past observations and corresponding $(\mathcal{L}_t, \mathcal{V}_t)$ is visualized in Fig. 4-(b).

Using the local EDF $\phi(\mathbf{x})$ built from online mapping, we can now check the collision states of all $v_t \in \mathcal{V}_t$. Based on the offline bitset association between $\mathcal{V}$ and $\mathcal{L}$, we prune out the trajectories in $\mathcal{L}_t$ which are associated with v_t inside the obstacle region, and collect feasible prediction primitives. The final prediction $\mathbf{x}_q(t)$ is chosen among them, which minimizes the following observation error:

$$E(\mathbf{x}_q(t)) = \frac{1}{N_o}\sum_{n=1}^{N_o} \|\mathbf{x}_q(t_n) - \bar{\mathbf{x}}_{q,n}\|^2 \tag{3}$$

Fig. 4-(b) illustrates the feasible candidates and the final prediction $\mathbf{x}_q(t)$ in a set of black curves and a red curve, respectively.

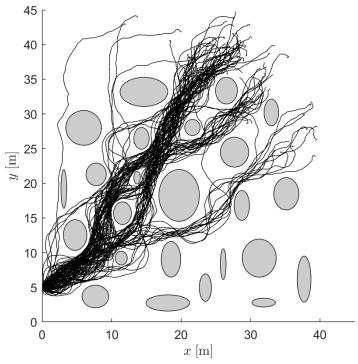

FIGURE 5. The set of target trajectories for testing prediction accuracy.

TABLE 1. Test result for prediction accuracy (mean/max) [m].

σ_n \ σ_{vw}	$(0.1, 10)$	$(0.5, 12)$	$(0.7, 13)$	$(0.9, 14)$
0.01	0.38/0.51	0.39/0.57	0.41/0.50	0.40/0.65
0.1	0.59/1.23	0.70/1.27	0.69/1.35	0.75/1.31
0.2	0.79/1.36	0.70/1.40	0.74/1.39	0.66/1.32
0.3	0.88/1.41	0.87/1.38	0.85/1.53	0.85/1.49

C. ANALYSIS

Before combining the proposed forecasting algorithm with the chaser motion planning module, we perform a trial test on a target with the unicycle model in an obstacle environment shown in Fig. 5. The linear velocity and angular velocity of the unicycle model are randomly actuated with Gaussian distributions whose standard deviations are $\sigma_v \in \mathbb{R}^+$ and $\sigma_w \in \mathbb{R}^+$ respectively. Let us write the pair as $\sigma_{vw} = (\sigma_v, \sigma_w)$. The prediction algorithm is provided with N_o past observations of the position with a Gaussian noise $\mathcal{N}(0, \sigma_n^2)$ for both x and y elements where $\sigma_n \in \mathbb{R}$ is the deviation of the noise.

As Fig. 5 shows, the initial location of the target as $[0, 5]$ m with the heading direction of $\pi/4$. Then, we consider four different levels for σ_{vw}: $\{(0.1, 10), (0.5, 12), (0.7, 13), (0.9, 14)\}$ (the units are m/s for σ_v and s^{-1} for σ_w) and sample 25 non-colliding target trajectories from each level of randomness, totalling 100 trajectories. The individual trajectory is integrated over 50 s, and we repeatedly predict the target motion over 2 s using $N_o = 10$ noisy observations whose standard deviations σ_n are chosen among $\{0.01, 0.1, 0.2, 0.3\}$ m. From this, we have 4×4 combinations of the actuation randomness σ_{vw} and the observation noise σ_n as shown in the Table. 1. For the offline library, we used the same setup in Section III-A, which produced the example in Fig. 4-(a).

To evaluate the local accuracy of the prediction over short horizon 2 s, we average the difference between the true position of the target and the predicted position. To evaluate the global accuracy over the entire duration 50 s, we take another average for the accumulated evaluations of the local accuracy. Each cell of Table. 1 summarizes the mean and max value of the global accuracy obtained from 25 test trajectories for a given actuation randomness and the observation. The average computation time for the setting was less than 20 ms in a standard i7 computer with 32 GB RAM.

IV. VISIBILITY SCORE FIELD FOR DUAL-TARGET

In the previous section, we discussed how to predict the motion of the targets. Now, we develop a metric to encode how stably the visibility toward two targets is maintained for an chaser position. The rationale in proposing the metric is the robust satisfaction of **Condition 1** against uncertain factors (e.g. inaccuracy in control and localization of the targets or the chaser), rather than a binary checking for the constraints. Also, by presenting a fast sweeping algorithm, we reduce the computation of the score (1) for a single target, which is used in the evaluation of visibility score for the dual-target. The algorithm is one of the major improvements from our previous work [8].

A. VISIBILITY SCORE FOR DUAL-TARGET

For the stable satisfaction of **Condition 1**-(a) and (b), it is preferable to have a larger distance between LOSs and obstacles (including the volume of the targets). Regarding **Condition 1**-(c), a smaller angle between two LOSs is advantageous. Reflecting the two, the below equation defines the visibility score $\psi_{ab}(\mathbf{x})$ of a chaser location $\mathbf{x}$ for the positions of two targets $\mathbf{x}_q^a$ and $\mathbf{x}_q^b$.

$$\psi_{ab}(\mathbf{x}) = \psi_a(\mathbf{x})\psi_b(\mathbf{x}) + \kappa\omega_{ab}(\mathbf{x}), \tag{4}$$

where $\psi_a(\mathbf{x}) = \psi(\mathbf{x}; \mathbf{x}_q^a)$, $\psi_b(\mathbf{x}) = \psi(\mathbf{x}; \mathbf{x}_q^b)$, and κ is a positive weight. In order to reflect the inter-occlusion, each target is considered as an obstacle with a volume when scoring another target (e.g. $\mathbf{x}_q^b$ is treated as an occluder with a volume when computing $\psi(\mathbf{x}; \mathbf{x}_q^a)$). In this work, we model the volume as a spheroid with the major axis along z direction. $\omega_{ab}(\mathbf{x})$ is defined as $\max\{\theta_{max} - \theta_{ab}(\mathbf{x}), 0\}$ where θ_{max} denotes the FOV limit of the drone, and $\theta_{ab}(\mathbf{x})$ is the angle of two lines $L(\mathbf{x}, \mathbf{x}_q^a)$ and $L(\mathbf{x}, \mathbf{x}_q^b)$. We call the scoring field derived from (4) *dual visibility score field* (DVSF).

An example DVSF and its construction are illustrated in Fig. 6-(b). In the first and second rows of the column, $\psi_a(\mathbf{x})$ and $\psi_b(\mathbf{x})$ treated $\mathbf{x}_q^b$ and $\mathbf{x}_q^a$ as part of obstacles (black circles) respectively. Due to this, for a position $\mathbf{x}$ s.t. $\psi_a(\mathbf{x}) \neq 0$ and $\psi_b(\mathbf{x}) \neq 0$, there exist LOSs toward the two targets which are not occluded either from obstacles or the other target. The third row shows the bearing score $\omega_{ab}(\mathbf{x})$ when FOV is 120°. The dark regions correspond to $\omega_{ab}(\mathbf{x}) = 0$ where the angle of two LOSs to the targets becomes larger than the FOV. In the contrary case where the **Conditions 1**-(c) is satisfied, $\omega_{ab}(\mathbf{x}) > 0$ holds.

From the Fig. 6-(b), we can observe the trade-off between the two score terms in (4), which is governed by κ. To increase the second term (reducing the bearing angle), it is advantageous for the chaser to observe both targets aligned in a straight line (see the white region of the third row).

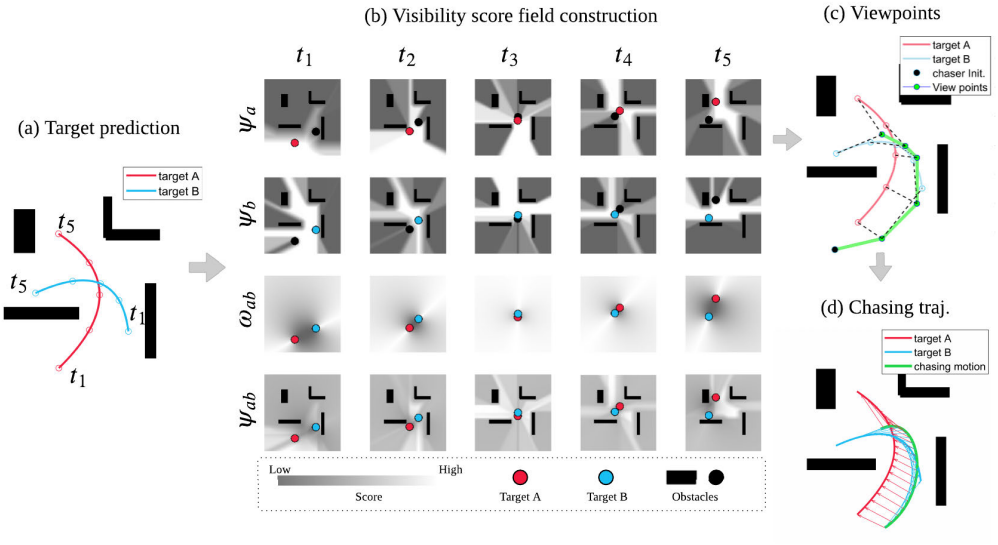

FIGURE 6. Proposed planning algorithm to chase two targets. Given the predictions of the targets and the local map information, we investigate the visibility field for dual-target (DVSF) by reflecting **Condition 1**. Based on the field, we search optimal viewpoints and output a smooth chasing motion based on the viewpoints.

However, the region with higher $\omega_{ab}(\mathbf{x})$ is more vulnerable to the inter-occlusion, resulting in a smaller value for $\psi_a(\mathbf{x})\psi_b(\mathbf{x})$. Thus, for the construction of DVSF, higher κ is preferable if the two targets are to be closely captured in the center of the image. On the contrary, setting a smaller value for κ is more advantageous when we give more importance to the inter-occlusion. Putting all the considerations into together, we construct DVSF $\psi_{ab}(\mathbf{x})$ as shown in the last row of Fig. 6-(b) so that the score (4) have a larger value for the location where the three terms in **Condition 1** are more stably satisfied.

B. FAST COMPUTATION FOR A SINGLE VISIBILITY FIELD

Now, we focus on the efficient computation of DVSF on a local gridmap where EDF $\phi(\mathbf{x})$ is computed. The scoring (4) involves evaluations of VSFs $\psi_a(\mathbf{x})$ and $\psi_b(\mathbf{x})$ for a single target. In our previous work [8], we evaluated the visibility score (1) with respect to a single target $\mathbf{x}_q$ and a chaser $\mathbf{x}_p$ by finding the minimum of $\phi(\mathbf{x})$ traversing all the cells included in $L(\mathbf{x}_p, \mathbf{x}_q)$. Although the method in our previous work can obtain the accurate VSF associated with the gridmap, the approach might not be fast enough for the online application when multiple VSFs should be computed for DVSF. For the multi-target setting, this section introduces an algorithm based on recursion and interpolation. We first derive the following relation for the visibility score (1) with EDF $\phi(\mathbf{x})$:

Lemma 1: For a target of interest $\mathbf{x}_q$, EDF $\phi(\mathbf{x})$ with a grid resolution Δx, and a chaser location $\mathbf{x}_p$ on the center of a grid cell,

$$\psi(\mathbf{x}_p; \mathbf{x}_q) = \min\{\phi(\mathbf{x}_p), \psi(\mathbf{x}_p{}'; \mathbf{x}_q)\} \tag{5}$$

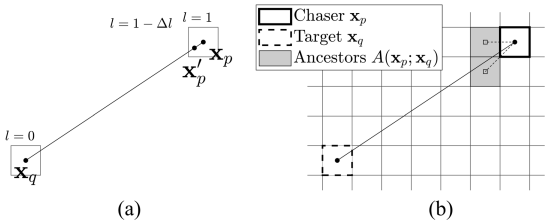

FIGURE 7. (a) The illustration for the recursion (5) in **Lemma 1**. (b) Interpolation of $\psi(\mathbf{x}_p{}'; \mathbf{x}_q)$ with ancestors $A(\mathbf{x}_p; \mathbf{x}_q)$.

holds for $\mathbf{x}_p{}' = \mathbf{x}_p + (\mathbf{x}_q - \mathbf{x}_p)\Delta l$ where Δl is a small perturbation s.t $0 < \Delta l < \Delta x \,/\, (2\|\mathbf{x}_q - \mathbf{x}_p\|)$.

Proof: Let us parameterize a point on $L(\mathbf{x}_p, \mathbf{x}_q)$ as $\mathbf{x}(l) = l\mathbf{x}_p + (1-l)\mathbf{x}_q \in \mathbb{R}^3$ where $l \in [0, 1]$. For a small positive number $0 < \Delta l < \Delta x \,/\, (2\|\mathbf{x}_q - \mathbf{x}_p\|)$, a point $\mathbf{x}_p{}' = \mathbf{x}(1 - \Delta l) \in \mathbb{R}^3$ exists on the cell whose center is $\mathbf{x}_p$ as $\|\mathbf{x}_p{}' - \mathbf{x}_p\| < \Delta x/2$ holds. Since EDF $\phi(\mathbf{x})$ has a homogeneous value inside a cell defined by the resolution Δx, $\phi(\mathbf{x}) = \phi(\mathbf{x}_p)$ holds for all the points $\mathbf{x} \in L(\mathbf{x}_p, \mathbf{x}_p{}')$, giving us $\phi(\mathbf{x}(l)) = \phi(\mathbf{x}_p)$ for $1 - \Delta l < l < 1$. Due to this,

$$\min_{l\in[0,1]} \phi(\mathbf{x}(l)) = \min\{\phi(\mathbf{x}(1)), \min_{l\in[0,1-\Delta l]} \phi(\mathbf{x}(l))\}$$

holds. Recalling the definition (1), we have $\min_{l\in[0,1]} \phi(\mathbf{x}(l)) = \psi(\mathbf{x}_p; \mathbf{x}_q)$ and $\min_{l\in[0,1-\Delta l]} \phi(\mathbf{x}(l)) = \psi(\mathbf{x}_p{}'; \mathbf{x}_q)$. Thus, $\psi(\mathbf{x}_p; \mathbf{x}_q) = \min\{\phi(\mathbf{x}_p), \psi(\mathbf{x}_p{}'; \mathbf{x}_q)\}$. □

An example of $\mathbf{x}_p{}'$ for the pair $\mathbf{x}_q$ and $\mathbf{x}_p$ is shown in Fig. 7-(a).

In order to use the recursion of **Lemma 1** in computing $\psi(\mathbf{x}_p; \mathbf{x}_q)$ where $\mathbf{x}_p$ exist on the center of the cells in

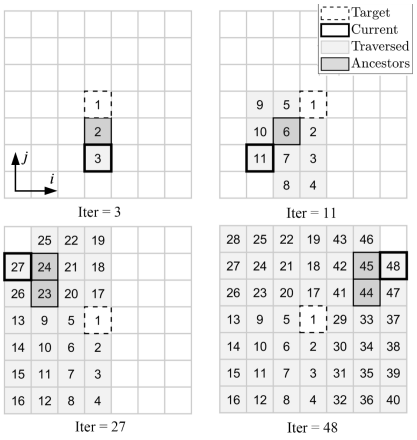

FIGURE 8. The graphical illustration of the proposed sweeping for a 2D gridmap. We compute the visibility field $\psi(\mathbf{x}; \mathbf{x}_q)$ while maintaining the order of evaluations for ancestors (dark gray cells) always to precede the currently evaluated cell (thick black-edged cell).

a gridmap, we need $\psi(\mathbf{x}_p{}'; \mathbf{x}_q)$ at $\mathbf{x}_p{}'$. As we operate on grid cells from the mapping, we interpolate $\psi(\mathbf{x}_p{}'; \mathbf{x}_q)$ from a set of neighboring cells of $\mathbf{x}_p$ which intersect with $L(\mathbf{x}_p, \mathbf{x}_q)$. We define those cells as the *ancestors* of $\mathbf{x}_p$ written as $A(\mathbf{x}_p; \mathbf{x}_q) \subset \mathbb{R}^3$. The example of ancestors is given in Fig. 7-(b). Assuming that the visibility scores are precomputed on the ancestors $A(\mathbf{x}_p; \mathbf{x}_q)$, we interpolate the score $\psi(\mathbf{x}_p{}'; \mathbf{x}_q)$ as below.

$$\psi(\mathbf{x}_p{}'; \mathbf{x}_q) = \sum_{\mathbf{x} \in A(\mathbf{x}_p; \mathbf{x}_q)} \tilde{w}(\mathbf{x}) \psi(\mathbf{x}; \mathbf{x}_q), \tag{6}$$

where $\tilde{w}(\mathbf{x}) = w(\mathbf{x}) \,/\, \sum_{\mathbf{x} \in A(\mathbf{x}_p; \mathbf{x}_q)} w(\mathbf{x})$ with $w(\mathbf{x}) = \dfrac{(\mathbf{x} - \mathbf{x}_p) \cdot (\mathbf{x}_q - \mathbf{x}_p)}{\|\mathbf{x} - \mathbf{x}_p\| \|\mathbf{x}_q - \mathbf{x}_p\|}$. The weight $\tilde{w}(\mathbf{x})$ has a larger value as the direction from the chaser to the ancestor is more aligned with $L(\mathbf{x}_p, \mathbf{x}_q)$.

If we compute the visibility field for a grid region from (5) and (6), we can observe that visibility evaluation of the ancestors $A(\mathbf{x}_p; \mathbf{x}_q)$ should precede the $\mathbf{x}_p$, which requires a sweeping method that maintains the order of evaluations. For this purpose, we adopt the sweeping method summarized in **Algorithm 1** and Fig. 8.

C. ANALYSIS

Here, we analyze the proposed visibility field sweeping method in **Algorithm 1** in terms of computation speed compared to previous work [8] and the accuracy. Concerning the accuracy, we assume the visibility map computation of [8] to be the ground truth in a given grid setting. For the sake of simpler discussion, we use a 40 m $\times$40 m 2D obstacle map depicted in Fig. 9 where the target is positioned at $\mathbf{x}_q = [20, 20]$ m. In Fig. 9-(a) and (b), we can find two maps derived from our previous work [8] and **Algorithm 1** when the resolution is 0.4 m. With respect to 9-(a) and (b), Fig. 9-(c) shows the error of the score computation $e(\mathbf{x})$ at each cell $\mathbf{x}$ versus distance from the target $\|\mathbf{x} - \mathbf{x}_q\|$ where

Algorithm 1 Visibility Sweeping

Input : *target cell index* (i_q, j_q, k_q)
bound $(i_{min}, j_{min}, k_{min}), (i_{max}, j_{max}, k_{max})$
Initial: $\psi_{i_q j_q k_q} = \phi_{i_q j_q k_q}$

```
for Δi = {−1, 1} do
    for Δj = {−1, 1} do
        for Δk = {−1, 1} do
            i = i_q, j = j_q k = k_q
            while (i, j, k) in bound do
                ψ_ijk = compute(φ_ijk, A_ijk)
                i ← i + Δi, j ← j + Δj, k ← k + Δk
            end
        end
    end
end
```

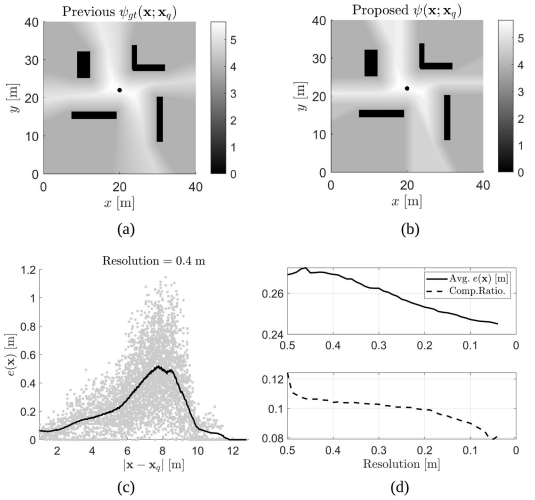

FIGURE 9. (a) and (b): exemplary VSFs for a single target (black dot) obtained from the previous work and the newly proposed method. (c): the score error $e(\mathbf{x})$ defined in (7) versus the distance from the target position. (d): the average error of the VSF obtained from the proposed method and the computation time ratio compared to the previous method, versus the grid resolution.

$e(\mathbf{x})$ is defined as below.

$$e(\mathbf{x}) = |\psi(\mathbf{x}; \mathbf{x}_q) - \psi_{gt}(\mathbf{x}; \mathbf{x}_q)| \tag{7}$$

where $\psi(\mathbf{x}; \mathbf{x}_q)$ is the score obtained from the proposed method and $\psi_{gt}(\mathbf{x}; \mathbf{x}_q)$ is from the previous work [8]. As can be seen, the error increases as a cell is farther from the target, while remaining below 1.2 m. The decrease in the error from $\|\mathbf{x}_q - \mathbf{x}\| = 8$ m corresponds to the occluded region where the score values were evaluated to zeros in both methods.

Fig. 9-(d) analyzes the accuracy and the computation speed from 50 cases of resolutions varying from 0.5 m to 0.02 m on the same settings with (a) and (b) of the figure. As can be observed, the average error of the proposed method decreases for the smaller grid resolution, while the computation time of the proposed algorithm becomes almost 10 times faster than

the previous work [8] from the resolution 0.3 m. The test was performed on a standard i7 laptop computer with a single-thread implementation.

V. CHASING TRAJECTORY GENERATION

In the previous sections, we introduced the DVSF $\psi_{ab}(\mathbf{x})$ to reflect the **Condition 1** and the efficient sweeping **Algorithm 1**. Based on these, this section presents a hierarchical planning structure to output the chasing motion of the drone, while optimizing the visibility with other objectives and constraints. As the increased number of the targets and the conflicts between objectives can entail numerical issues such as local-minima, we decompose the optimization into multiple stages as illustrated in Fig. 6, rather than relying on a single non-convex optimization. We first compute a sequence of viewpoints from a graph search algorithm, then we generate a smooth motion using QP. Although the basic concept of the structure is similar to our previous [8], we improve the graph search process based on our another work [24] and additionally consider the collision between the targets and the drone.

A. OPTIMAL VIEWPOINT SKELETON

We assume that the local map information and initial drone position $\mathbf{x}_{p,0} \in \mathbb{R}^3$ are given. Setting uniform time steps $t_n \in \mathbb{R}^+$ $(n = 0, \dots, N)$ on the horizon $[0, T]$, we focus on the optimal viewpoints of the chaser $\sigma = \{\mathbf{x}_{p,n}\}_{n=1}^{N}$ with respect to the known future target positions $\mathbf{x}_{q,n}^a = \mathbf{x}_q^a(t_n)$ and $\mathbf{x}_{q,n}^b = \mathbf{x}_q^b(t_n)$. To ensure the key objectives and essential conditions in section II-C, we compute σ with the following optimization:

$$\begin{aligned}
&\min_{\sigma} \sum_{n=1}^{N} c(\mathbf{x}_{p,n-1}, \mathbf{x}_{p,n}),\\
&\text{subject to } \mathit{Group}1\begin{cases}\max\{\|\mathbf{x}_{p,n}-\mathbf{x}_{q,n}^a\|,\|\mathbf{x}_{p,n}-\mathbf{x}_{q,n}^b\|\} < r_{max},\\ \psi_{a,n}(\mathbf{x}_{p,n}) > \epsilon_v,\\ \psi_{b,n}(\mathbf{x}_{p,n}) > \epsilon_v,\\ \omega_{ab,n}(\mathbf{x}_{p,n}) > 0,\end{cases}\\
&\qquad\qquad \mathit{Group}2\begin{cases}\|\mathbf{x}_{p,n-1} - \mathbf{x}_{p,n}\| < d_{max},\\ \phi(\mathbf{x}) \geq \epsilon_s \text{ for } \forall \mathbf{x} \in L(\mathbf{x}_{p,n-1}, \mathbf{x}_{p,n}),\\ L(\mathbf{x}_{p,n-1}, \mathbf{x}_{p,n}) \cap L(\mathbf{x}_{q,n-1}^a, \mathbf{x}_{q,n}^a) = \emptyset,\\ L(\mathbf{x}_{p,n-1}, \mathbf{x}_{p,n}) \cap L(\mathbf{x}_{q,n-1}^b, \mathbf{x}_{q,n}^b) = \emptyset,\end{cases}
\end{aligned} \tag{8}$$

where the cost between two viewpoints $\mathbf{x}_{p,n-1}, \mathbf{x}_{p,n}$ is defined as

$$\begin{aligned}
&c(\mathbf{x}_{p,n-1}, \mathbf{x}_{p,n})\\
&= \underbrace{\|\mathbf{x}_{p,n-1} - \mathbf{x}_{p,n}\|}_{\text{Travel distance}} + \lambda_v \underbrace{(\psi_{ab,n}(\mathbf{x}_{p,n}))^{-1}}_{\text{Visibility}}\\
&\quad + \lambda_d (\underbrace{|\|\mathbf{x}_{p,n} - \mathbf{x}_{q,n}^a\| - r_{des}|}_{\text{Relative distance with A}} + \underbrace{|\|\mathbf{x}_{p,n} - \mathbf{x}_{q,n}^b\| - r_{des}|}_{\text{Relative distance with B}}).
\end{aligned} \tag{9}$$

In the above equations (8) and (9), time step index n was attached to the terms $\psi_a(\mathbf{x})$, $\psi_b(\mathbf{x})$, $\psi_{ab,n}(\mathbf{x}_{p,n})$, and $\omega_{ab}(\mathbf{x})$, which are associated with $\mathbf{x}_{q,n}^a$ and $\mathbf{x}_{q,n}^b$. λ_v and λ_d are positive importance weights. The objective function (9) efforts to minimize the total distance of σ and maximize the visibility scores. Also, the desired relative distance r_{des} toward the targets is to be maintained.

In the constraints of (8), **Group 1** denotes the conditions imposed to the viewpoint $\mathbf{x}_{p,n}$ at a single time step, while **Group 2** corresponds to the conditions along two consecutive time steps. The first constraint in **Group 1** confines the relative distances toward both targets below r_{max}. The second and third prevent the occlusion of the targets at the corresponding step with a margin ϵ_v. The last imposes the FOV limit at each step. In **Group 2**, the first constraint bounds the maximum distance between two consecutive viewpoints below d_{max}. The second constraint ensures the safety along σ with a margin ϵ_s, while the last two prevent the collision between the targets and the chaser.

We obtain the optimal σ of (8) by choosing a set of discrete cells on the 3D grid where $\phi(\mathbf{x})$ is computed. For this purpose, we leverage the graph search on a directed-acyclic-graph (DAG) $G = (V, E)$. The set of nodes V is composed of subsets V_n along time steps t_n where V_n has only feasible cells around the targets with respect to the **Group 1** constraints (for t_0, V_0 has a sole element $\mathbf{x}_{p,0}$). The edges of E are created by connecting nodes from $\mathbf{x}_n \in V_n$ to $\mathbf{x}_{n+1} \in V_{n+1}$ once **Group 2** is satisfied. Then we assign a weight to each edge according to the cost function $c(\mathbf{x}_{p,n-1}, \mathbf{x}_{p,n})$. From the graph construction, every node and edge satisfies the constraints in (8), and the graph search on G from V_0 to every node in V_N is equivalent to finding the optimal σ. The DAG shares the same structure introduced in our another previous work [24] where we can immediately find the topological sorting on V without iteration. Thus, only edge-relaxation step [25] is required for the graph search, allowing us a fast computation of (8). For a more detailed discussion, we refer the reader to [24].

B. SMOOTH CHASING TRAJECTORY

In the previous section, we computed the high-quality viewpoints $\sigma = \{\mathbf{x}_{p,n}\}_{n=1}^{N}$ which satisfy the essential conditions and optimize the key objectives. Given the initial state of the chaser $\mathbf{x}_{p,0}$, $\dot{\mathbf{x}}_{p,0}$, $\ddot{\mathbf{x}}_{p,0}$, we now finalize the chasing motion of the drone enjoying the features of the viewpoints. We plan the translation and yaw $(\mathbf{x}_p(t), y(t)) \in \mathbb{R}^4$ based on the differential flatness of quadrotor dynamics [26]. The translation $\mathbf{x}_p(t)$ is parameterized with piece-wise polynomial curves written as the below:

$$\mathbf{x}_p(t) = \begin{cases} \sum_{k=0}^{K} \mathbf{p}_{1,k} t^k & (t_0 \leq t < t_1)\\ \sum_{k=0}^{K} \mathbf{p}_{2,k} t^k & (t_1 \leq t < t_2)\\ \dots & \\ \sum_{k=0}^{K} \mathbf{p}_{N,k} t^k & (t_{N-1} \leq t < t_N) \end{cases} \tag{10}$$

where $\mathbf{p}_{n,k} \in \mathbb{R}^3$ denotes the polynomial coefficient for order k defined over time interval $[t_n, t_{n+1})$ (the same time steps in the previous section are used). The piece-wise polynomial is computed from the following optimization:

$$\begin{aligned} \min \sum_{d=2}^{4} \int_{t_0}^{t_N} \rho_d \|\mathbf{x}_p^{(d)}(t)\|^2 dt \\ \text{subject to } \mathbf{x}_p^{(d)}(0) = \mathbf{x}_{p,0}^{(d)} \ \ (d = 0, 1, 2) \\ -\epsilon_w \preceq \mathbf{x}_p(t_n) - \mathbf{x}_{p,n} \preceq \epsilon_w \\ -\frac{\phi(\tilde{\mathbf{x}}_p(t))}{\sqrt{3}} \preceq \mathbf{x}_p(t) - \tilde{\mathbf{x}}_p(t) \preceq \frac{\phi(\tilde{\mathbf{x}}_p(t))}{\sqrt{3}} \end{aligned} \quad (11)$$

where $\tilde{\mathbf{x}}_p(t)$ is the linear interpolant of the points $(\mathbf{x}_{p,0}, \sigma)$ over time steps $(t_0, t_1, \ldots, t_N)$ evaluated at t. The symbol $\preceq$ denotes the element-wise inequality (less than or equal to). ρ_d is a positive weight corresponding to the derivative of the dth order.

The above optimization minimizes the high-order derivatives from the second to the fourth order. The first constraint imposes the initial condition of the chaser, and the second bounds the deviation from viewpoints obtained from (8) to take advantage of their optimized visibility with a tolerance ϵ_w. The last constraint ensures the collision avoidance of the entire trajectory $\mathbf{x}_p(t)$ by confining it inside the obstacle-free space along $\tilde{\mathbf{x}}_p(t)$. As an extension of our previous work [8], we modified the safe bound from the constant ϵ_s to $\phi(\tilde{\mathbf{x}}_p(t))$ so that minimization of the objectives can benefit from a wider feasible space. Note that, for the linear interpolant $\tilde{\mathbf{x}}_p(t)$, EDF value $\phi(\tilde{\mathbf{x}}_p(t))$ is ensured to be larger than ϵ_s due to the second constraint in (8).

The above optimization can be rearranged to a constrained QP concerning to $\mathbf{p}_{n,k}$ in a similar way with our previous work [8]. The formulation into QP provides the numerical stability in finding the global optimal, leveraging the algorithm such as [27]. For the yaw motion $y(t)$, we decide it in a myopic way so that the optical axis heads to the center of the targets $(\mathbf{x}_q^a(t) + \mathbf{x}_q^b(t))/2$.

C. PIPELINE

As a summary, we combine the proposed modules from the previous sections. First, we obtain the local (or global if available) voxel map from the sensor of the drone and compute EDF $\phi(\mathbf{x})$. Based on observations and the map, we predict the future movement of the targets $\mathbf{x}_q^a(t)$ and $\mathbf{x}_q^b(t)$ over $t \in (0, T]$. Then, we calculate the sequence of DVSFs $\psi_{ab}(\mathbf{x})$ for the predicted points at time steps $t_n \in (0, T]$. The DVSF at a time step is computed inside a 3D bounding volume that encloses the two targets with a enough margin larger than r_{max} in (8).

From the information, a graph search on the DAG obtains viewpoints σ from (8) ensuring the travel efficiency, safety, visibility, and relative distance. On top of the optimal viewpoints toward the dual-target, we finalize the efficient motion of the chaser with QP (11), which strictly ensures the safety and uses the advantages of the skeleton σ. We repeat the described process to update the chasing motion when either the chasing motion becomes unreliable due to the newly discovered obstacles or the prediction error grows more than a defined level.

TABLE 2. Simulation result.

Key objectives					
Travel Dist. Ratio (chaser/target)		Avg. Accel Ratio (chaser/target)		Rel. Dist. Ratio (relDist/r_{des})	
A	**B**	**A**	**B**	**A**	**B**
1.12	1.07	0.78	0.79	0.84	1.22

Essential conditions					
Safety [m] (min/avg)	Occu. Obst.[m] (min/avg)		Occu. Inter. [m] (min/avg)		Bearing (avg/max)
	A	**B**	**A**	**B**	
0.8 / 2.6	**0.7** / 2.0	**0.8** / 2.4	**0.4** / 5.7	**0.5** / 3.1	0.9 / **1.4**

VI. VALIDATIONS

In this section, we extensively validate the online performance of the proposed algorithm from a high-fidelity simulation and multiple real-world experiments including indoor and outdoor scenarios.

A. HIGH-FIDELITY SIMULATION

A realistic simulation is performed using Unreal engine and AirSim [28] where a drone with a camera operates in a large-scale warehouse environment (120 m $\times$160 m) to follow two targets autonomously. The targets move for 357 m and 373 m respectively at 2 to 3 m/s. The simulation includes the 3D scenarios where the targets move along stairs and one of them jumps down to the ground from the second floor.

In the simulation, the chasing drone is assumed to have the information on the entire map and the future movement of the targets over the planning horizon $T = 8$. The settings are intended to examine the proposed planning algorithm when enough information is given, and this will be relaxed in the outdoor experiments with a real drone. The effective FOV of the drone was assumed to 100◦. The maximum pitching-down of the camera gimbal was limited to 45◦, which disables the vertical downward view from the drone. The resolution of the field was set to 0.4 m, and the desired relative distance was set to $r_{des} = 4.5$ m for both targets. The resolution for the octomap and DVSF were set to 0.2 m and 0.4 m respectively. The simulation was performed on an Intel i7 desktop with 32GB RAM.

The planning result is shown in Fig. 10 and Table. 2. Fig. 10 shows the position history of the two targets and the planning output of the chaser and the LOSs toward the targets. The dashed box (a) highlights when the targets entered a dense indoor, and (b) shows when the targets are on the stairs and the male target jumps down (target B in the blue curve). When the targets ascended and then descended along stairs, the drone adjusted its altitude and preceded the targets to prevent the

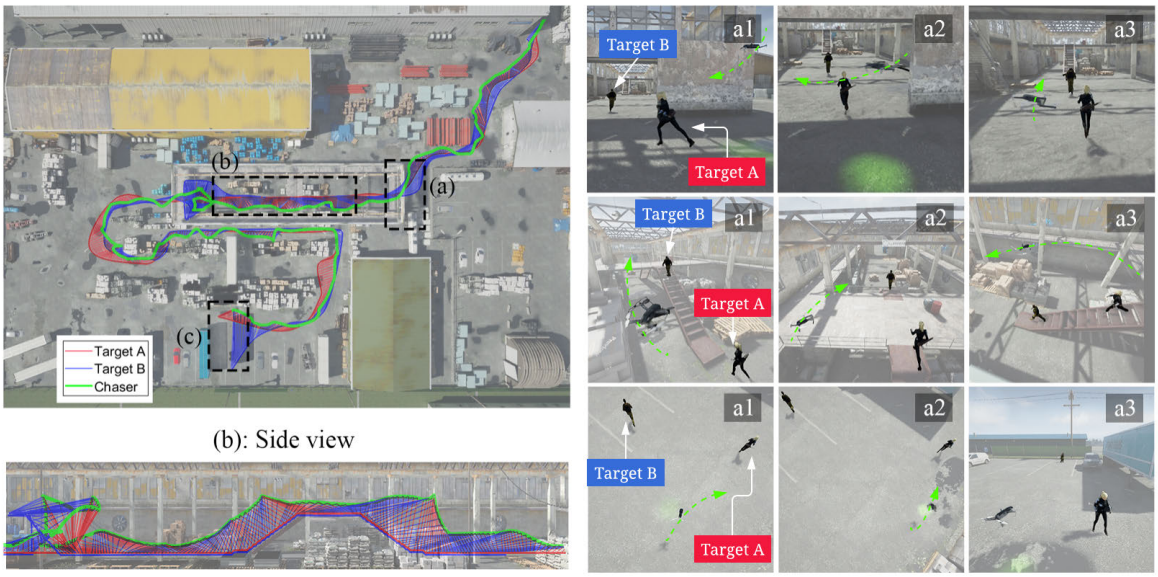

FIGURE 10. Simulation results in a warehouse environment (video can be found at https://www.youtube.com/watch?v=27f4tEpAIpw). The position history of the chaser (green) and two targets (red and blue) and bearings toward two targets from the chaser drone are plotted. The two targets and the chaser drone start from the north-east. The small figures a1-a3,b1-b3, and c1-c3 are the snapshots taken in dashed-box regions (a), (b), and (c) in the top-down view.

inter-occlusion from the height difference of the targets. The dashed box (c) corresponds to the scene where two targets start to diverge from each other. In order to maintain the bearing angle below the FOV limit, the drone moved near target A (red curve) to observe target B in the rear as well as target A.

Table. 2 analyzes the performance quantitatively in terms of the objectives and conditions, where A and B represent the values corresponding to the two targets, respectively. In the row corresponding to the key objectives, we can observe the travel distance was close to 1 and the ratio of average accelerations of the chaser and the targets was less than 1, showing the translation efficiency of the chaser. Also, the desired relative distance toward two targets was achieved in 20 percent error on average. The next row of the table shows the result of essential conditions. As seen in the first and the second column (Occu. Obst.), the safety measure $\phi(\mathbf{x})$ and visibility $\psi(\mathbf{x}; \mathbf{x}_q^a)$, $\psi(\mathbf{x}; \mathbf{x}_q^b)$ of the two targets remained positive with a margin (the radius of the collision volume of the drone was set to 30 cm, and the volumes of the targets were modeled as ellipsoids with the additional 20 cm margin). The third column (Occu. Inter.) denotes the closest distance of the LOS to one target from the ellipsoid volume of the other target, showing that the inter-target occlusion was prevented during the simulation. As seen in the last column (Bearing.), the maximum bearing angle of the proposed planning was 1.4 rad, which is less than the FOV limit. The average computation was faster than 10 Hz even while the Unreal engine ran with the high CPU and RAM usage. From the simulation, we confirmed that the aimed performance was achieved in a long-range mission among dense 3D obstacles.

B. INDOOR EXPERIMENT

Here, we present the key results of a real-world experiment where a chaser drone autonomously follows other two drones in an indoor environment without an external motion capture system. The targets traverse space with 5 m $\times$ 15 m scale at a speed 0.4 m/s. As shown in Fig. 11-(d), the chaser drone has a firmly attached vision sensor whose FOV is less than 100°. Intel NUC7i7BNH was mounted on the chaser for the onboard computation. All the drones utilized Intel realsense T265 camera for the visual odometry (VO). For the experiment, the grid resolution was set to 0.2 m, and the future step size $N = 3$ was set for the skeleton generation. The quintic polynomials were used to generate the smooth trajectory. The desired relative distance to targets was set to $r_{des} = 3.5$ m. The obstacles were mapped before the mission using realsense D435i. The two target drones flew autonomously along the predetermined trajectories which are shown in red and blue curves in the right columns of Fig. 11-(a) and (b), which are not known to the chaser drone. The chaser drone receives only the partial information over the horizon of 5 s at each planning trigger, as depicted in the thick segments of the targets' trajectories in the figure.

The experiment includes the following scenarios: 1) cornering of the targets at an obstacle and 2) passing through a narrow space between two obstacles as shown in (a) and (b) of Fig. 11 respectively. When the two targets were cornering right ($-y$ direction in the reference coordinate shown in the dashed circle in the Fig. 11-(a)), the chaser drone detoured a longer distance to capture both targets against the obstacle. The increased velocity along x and y elements is shown in Fig. 11-(c). When the two targets passed the gap

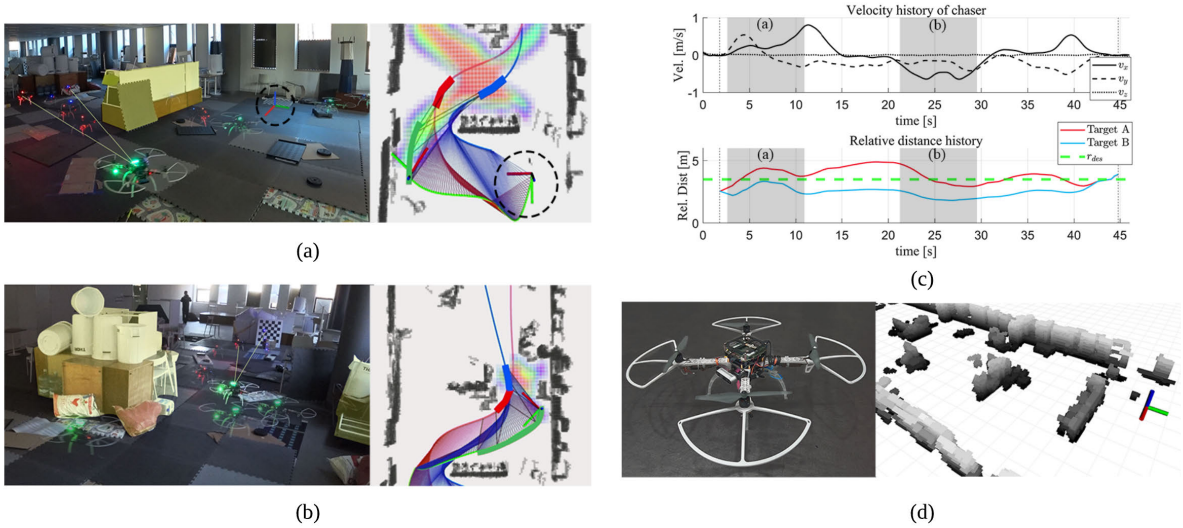

FIGURE 11. (a) and (b) The left column describes the trails of the chaser drone (green LED) and two target drones (red and blue LED). We highlighted the obstacles as yellow, which affect the planning outcome the most. In the right column, the algorithm views are shown. The black-dashed circle in (a) denotes the reference coordinate. DVSF of the last step of the future horizon is drawn in jet-colormap where the red scale corresponds to the high visibility zone. The thick curve segments of the targets (red and blue) denote the partial future trajectories known to the chaser. The thick green curve shows the planning result of the chaser over the considered horizon. The LOSs toward the targets are also stamped as red and blue arrows. (a) depicts the cornering of two targets at an obstacle while (b) corresponds to the narrowing of the targets between two clusters of obstacles. (c) The history of the velocity of the chaser drone (top) and the distance between the targets and the chaser (bottom). The grayed regions are matched with the scenes in (a) and (b). (d) The hardware setup and the EDF of the experiment scene.

TABLE 3. Onboard computation time.

Field eval.	Viewpoint		Smooth Traj.
0.44 ms	Graph building	Search	12.25 ms
	12.53 ms	0.05 ms	

between two close-by objects, the obstacles could occlude the targets or inter-occlusion could occur due to the small distance between the targets. The drone moved further to the right ($-x$ direction) to prevent the possible occlusions, rather than staying directly behind the targets. Throughout the experiment, the two targets were visible to the camera of the chaser drone, and the chaser avoids the collision with obstacles. The algorithm ran at nearly 40 Hz and onboard computation times are summarized in Table. 3, running in total. The averaged speed ratio of the chaser to each target was 1.56 and 1.39 respectively.

C. OUTDOOR EXPERIMENT

Now, we present the result of the outdoor experiment, implementing the whole onboard system in Fig. 3 where the sensor data processing and the prediction are fully included. The hardware setup for the chaser drone is illustrated in Fig. 12. We used two onboard computers: 1) NUC8i7BEH for the motion planning and the prediction, and 2) Jetson Xavier NX to run the sensor data processing. Jetson Xavier Nx is connected to a ZED2 stereo camera to obtain the visual odometry of the drone, the RGB image, and the depth image. For the flight controller, Pixhawk4 was used to obtain IMU data and run as an FMU. Another ZED camera is used to map the obstacles on the opposite side of the frontal ZED2 camera. ZED SDK [29] is used to detect the bounding box for human objects in the images and 3D coordinate. We also obtain 3D positions and velocities of the objects from the SDK. Then, we compute the dominant HSV colors for the detected objects, using the RGB pixels of 2D bounding boxes. At the start of the mission, the targets were set as the two closest objects from the chaser drone. To update the observation on the targets, the newly detected objects were matched with the previously tracked targets based on the similarity of the position, the velocity, and the dominant HSV color. The observed targets' positions are collected in the prediction module to compute the reference frame of (2) and the observation error of (3).

FIGURE 12. The hardware setup for the outdoor experiment.

Regarding online obstacle mapping, the pointcloud built from the original depth images can include the points from the targets and the bleeding artifacts [30] around the border

between the targets and the background. As illustrated in the top row of Fig. 13, mapping these points as obstacles can degrade the performance of the motion planning for the chaser, as prediction becomes infeasible and visibility evaluation (1) becomes incorrect. To prevent this, we first crop the target region of the depth image using the detection box in the RGB image (see the second row of Fig. 13), and use only the uncropped region for the generation of the pointcloud. When a target is not updated in the RGB image, we crop the 3D bounding box around the last-tracked position. Next, the radius outlier removal algorithm [31] is used to remove the bleeding from the intuition that the bleeding region generally forms a thin string of noisy points with fewer neighbors than the cluttering obstacle region. The example of the removed bleeding region is visualized as red points in the bottom of Fig. 13. The processed pointcloud is used for building octomap and its EDF. Based on the sensor data processing for targets' positions and the obstacle mapping, we thoroughly demonstrate the online performance of the chasing system in three outdoor scenes: a rooftop, a stair, and a forest. As will be discovered, each of them contains a distinct challenge in terms of the target movements and the environments. For the experiments, the drone was not given any prior information of the target movements and the operation space (e.g. slope, the height range of the obstacles). They are predicted and mapped on-the-fly, respectively.

For all the cases, the same parameter setting was used, showing that our algorithm does not require a sophisticated parameter tuning. For the octomap and the distance field, the resolution was set to 0.2 m and the distance field is computed until 2.0 m. We set the resolution of the DVSF as 0.4 m and the maximum connection as $d_{max} = 4.5$ m. The horizon for the prediction and the planning was set to $T = 2.0$ s and the step number $N = 3$ was used for the sequence of DVSFs. For the constraints of the preplanner, the maximum bearing angle was chosen as $\theta_{max} = 1.25$ considering the lens specification of the ZED2 camera. The desired relative distance toward the targets was $r_{des} = 3.5$ m where the performance of the object detection was observed to be stable. The maximum raycast range for building octomap was set to 8 m form the camera position. Our system runs faster than 30Hz on the onboard computer during the mission.

1) SCENE 1: ROOFTOP

This experiment site includes walls and multiple pillars, where the two targets sequentially undergo the following phases of relative movements: 1) walking side by side and 2) following each other in a circular path. This scene is challenging as the chaser should simultaneously consider the collision with pillars and inter-occlusion when the targets are circling. For the scenario, we performed two experiments with different chaser setups where one fully considers the inter-occlusion while the other does not (setting the occlusion volumes of the targets to zero). For the ease of discussion, let us call the former *proposed chaser* while the latter *baseline chaser*. For the baseline chaser setup, we increased the

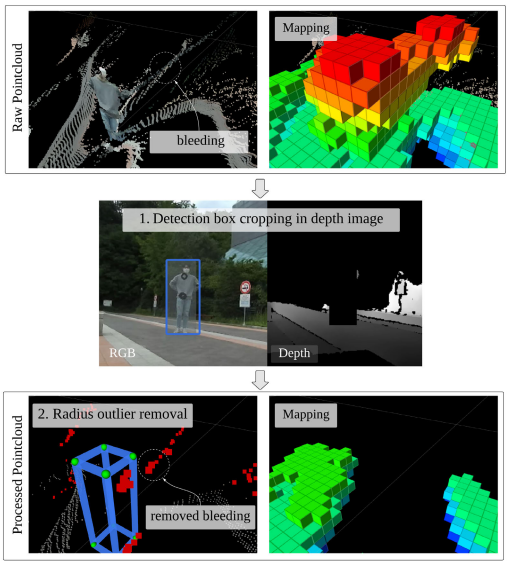

FIGURE 13. Processing pointcloud for obstacle mapping. To remove the pointcloud of the targets, we first crop the target region from depth image and remove the points around the targets' positions. Then, radius outlier removal is used to filter out the bleeding points on the border between the targets and the background (shown in red points in the bottom row).

importance of the distance minimization with smaller λ_v and λ_d in (9). All the other parameters were set as mentioned previously.

The comparative result is shown in Fig. 14 where the key snapshots and the planning history are included. Both proposed and the baseline setup yielded a similar motion when the targets were initially walking side-by-side. When the targets started to circle, however, the two setups showed different behaviors as highlighted in the yellow boxes of Fig. 14-(b). In the case of baseline, the drone remained in the yellow boxed region depicted in the top of Fig. 14-(b) and kept almost identical relative distance to targets without much yawing or translation. This motion led to repeated inter-occlusions whenever the two targets and the chaser drone are aligned in a line (see three snapshots in Fig. 14-(a)). In contrast, the proposed setup first makes the drone stride into the region where the targets are circling and then the drone arcs around to the direction of two targets as shown in the bottom row of Fig. 14-(b). Due to this motion, the drone was able to safely keep the sight into both targets without inter-occlusions as seen in the last row of Fig. 14.

Both strategies produced a safe motion for the drone without collision with obstacles or the targets. For two cycles of the circular motion of the targets, inter-occlusion occurred four times for the baseline while the proposed setup had no instance. During the mission, the averages of the magnitude of accelerations of the drone was 1.5 m/s^2 and 1.92 m/s^2 for the baseline and the proposed respectively. The average speeds was 0.42 m/s and 0.7 m/s while the maximums was 2.23 m/s and 4.32 m/s for the baseline and the proposed. From the experiments, the proposed construction of DVSF

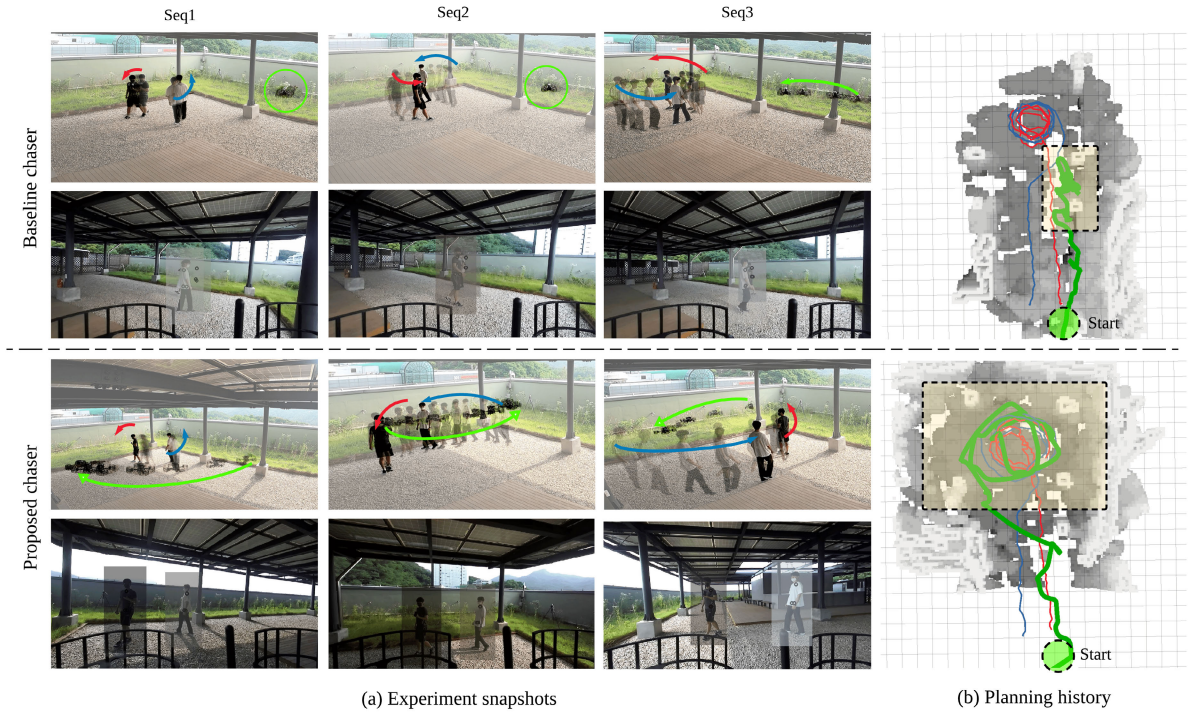

FIGURE 14. Rooftop scenario. (a) The three snapshots taken from the experiments when the targets are circling. Red and blue colored curves denote the movement of the targets while the drone's movement is colored in the green. The upper two rows show the result of the chasing motion when inter-occlusion was not considered. The lower two rows correspond to the setup where inter-occlusion was fully considered. (b) The green curve shows the history of the planning motion. The red and the blue curves are the history of the tracked positions of the targets. The voxels represent the local distance field truncated at 0.3 m when the mission ends. The gray scale corresponds to the height along z axis where white color corresponds to the higher value of z. The thin black grid on the plane is 1 m × 1m.

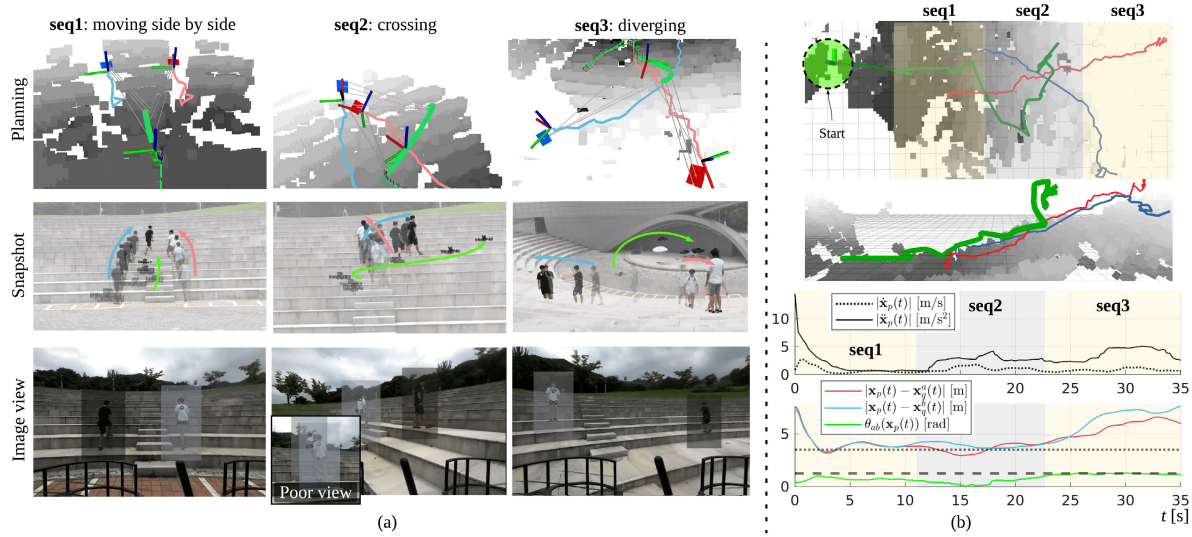

FIGURE 15. Stair scenario. (a) Experiment result of the three sequences. (b) Top: the positional history of the targets (red and blue) and the chaser drone (green) from a top view toward $x-y$ plane. Here the shading of three sequence is associated with the target positions. Middle: side view of the position histories. The gray color scale is associated with the height along z axis. Bottom (the last two columns): the key objectives along the mission time.

was found to effectively handle the inter-occlusion despite of the imperfect information on the targets' motions and the obstacles.

2) SCENE 2: STAIR

Here, the proposed system is tested in a scenario where the two targets climb up a wide stairway. As mentioned, no prior

information of the slope is given and the environment (stair) is mapped online. As illustrated in **seq1**, **seq2**, and **seq3** of Fig. 15, the relative movement between the two targets involves three phases: 1) walking side by side, 2) crossing each other, and 3) diverging. When the targets suddenly change their relative motions from the first to the second phase, inter-occlusion can interrupt the chasing of the drone if it keeps observing the targets from the backside. The sub-window of the image view of the **seq2** in Fig. 15-(a) shows an example of such poor view. The transition from the second to the third phase involves diverging motion between the targets. As the vision sensor observing the targets has a limited field of view, the targets can leave the image view of the drone. In this experiment, we observe the resultant motions in response to the two transitions.

As can be seen from **seq1** of the bottom two sub-figures in Fig. 15-(b), the planned chasing motion first increased the velocity to keep the desired relative distance $r_{des} = 3.5$ m as the drone started around 6 m behind the targets. In **seq2**, the chaser moves to the right side of the targets' heading, reducing the occlusion of the target with the black T-shirt from the target with the white T-shirt. In **seq2**, the chaser drone increased only its height and reduced its $x - y$ translation as shown in the second row of Fig. 15-(b). From this, the chaser was able to capture both targets in the limited FOV, sacrificing the desired relative distance toward the targets. This can be seen quantitatively from the red and blue curves in the bottom of Fig. 15-(b). In this experiment where the targets maneuver in three-dimensions, we observed that the proposed chasing system was able to overcome the cases (b) and (c) of Fig. 2 while mapping the terrain on-the-fly.

3) SCENE 3: FOREST

Now, we validate the proposed system in a forest environment where multiple unstructured obstacles can occlude the targets and the chaser drone should handle the collision with the obstacles. During 2.5 minutes, the two targets move in a three-dimension following the slope of the terrain. In this setting, we predict the target while planning an online chasing motion that jointly handles safety and visibility of the targets considering **Condition 1**. The top-down view and the perspective view of four key sequences (**seq1** to **seq4**) are shown in Fig. 16-(a) and Fig. 1, respectively. Their snapshots can be found in Fig. 17.

The first sequence **Seq1** in Fig. 17 shows the two targets passing through a tree whose trunk and foliage can hamper the safety of the drone and the visibility of the targets (see two yellow regions in the snapshot view). To handle the foliage, the proposed algorithm outputs a descending motion for the drone to maintain the safety and the visibility. After passing the tree, the second sequence (**seq2**) highlights how the drone prevents the possible inter-occlusion of the targets. As can be seen in Fig. 17, the drone takes a circling motion, showing a similar result with the proposed chaser of the rooftop scenario

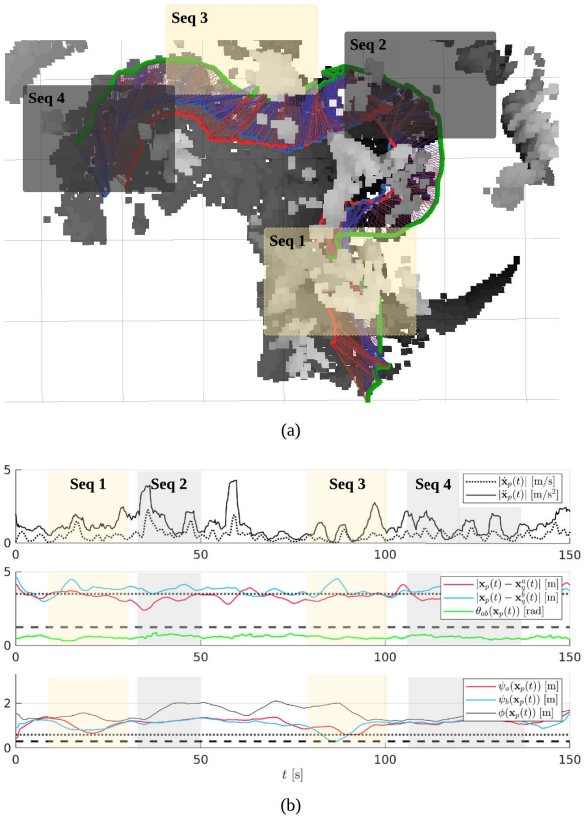

FIGURE 16. (a) Forest scenario is illustrated in a top down view of the planning history and the octomap at the end of the mission. The green curve denotes the history of the chaser drone while the red and blue curves show the targets' histories. The grid size is 5 m. (b) The history of key objectives and constraints is shown in the matching color with (a). In the middle row, the dotted line is the desired relative distance $r_{des} = 3.5$ m, and the dashed line is the maximum bearing angle $\theta_{max} = 1.25$. In the bottom row, the dotted line is the safety margin $\epsilon_s = 0.8$ m, and the dashed line is visibility margin $\epsilon_v = 0.3$ m.

shown in the bottom of Fig. 14. From the motion, the chaser was able to stay in the region with high value of $\psi_{ab}(\mathbf{x})$ as illustrated in the red-colored area of the planning view. The low-scored region due to the occlusion between the targets is also shown in the black-dashed box. As shown in the camera view, the drone keeps observing the target without the inter-occlusion in contrast to the poor view obtained from the same baseline parameter setting used in the rooftop scenario.

In the third sequence (**seq3**), targets climb up a slope. In response, the chaser is ascending to keep the visibility of the targets, while avoiding the tree highlighted as yellow. As can be seen in the planning view, the chaser circumvents the tree (the black dashed box) to the right side rather than the left side (poor view in the green circle) to avoid the occlusion of targets (see poor view in the camera view). The chasing mission ends at the last sequence (**seq4**) where the chaser could avoid the tree thanks to the safe corridor (cyan boxes in the planning view) that maintains the safety margin. During the experiment, as shown in Fig. 16-(b), the chaser drone

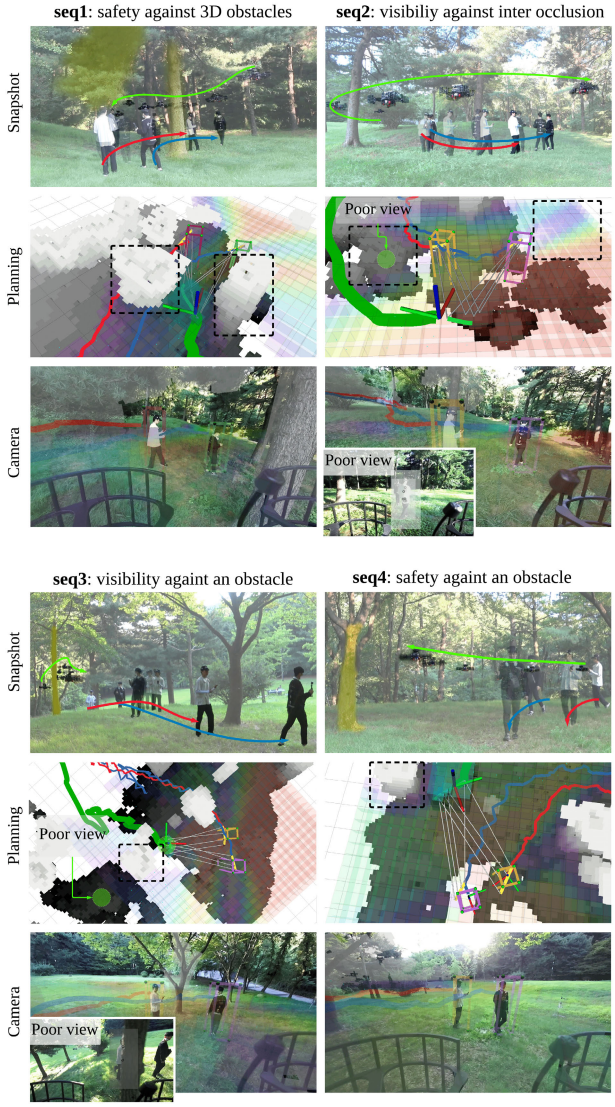

FIGURE 17. **Forest scenario. For each sequence, the first row shows the experiment snapshot where the yellow regions are the obstacles in the corresponding local planning. The second row is the planning view. It includes the 2D-slice of DVSF $\psi_{ab}(\mathbf{x})$ (jet-colored field) at the first time step t_1 and the planning result (thin green curve). The chasing corridors are drawn in cyan boxes. The traces of the chaser (thick green curve) and the two targets (red and blue curves) are also represented. The truncated EDF $\phi(\mathbf{x})$ around the chaser is drawn in the gray scale cubes, and the black-dashed boxes correspond to the yellow regions of the first row. In seq2 and seq3, the example poor views are marked as green circles. The third row of each sequence shows the camera view of the drone. EDF $\phi(\mathbf{x})$ and DVSF $\psi_{ab}(\mathbf{x})$ are overlaid on the camera image. The poor views are obtained from the green-circle-location in the planning views.**

was able to maintain its safety and the visibility toward both targets, satisfying the **Condition 1** without prior information of the environment and the targets' movement.

VII. CONCLUSION

In this paper, we proposed online chasing system which forecasts the motions of the targets and outputs the chasing motion in obstacle environments considering key objectives such as safety, visibility, motion efficiency, and the desired relative distances to the targets. The forecasting algorithm was able to predict the future trajectories of the targets not to intersect with the obstacle region and reflect the past observations. For the planning, the visibility metric was introduced to reflect **Condition 1** which is essential to capture the two targets with a single camera. The hierarchical planning structure was presented to optimize the key objectives in a numerically stable way. We also validated the online performance of the algorithm in challenging scenarios where the targets move in the three-dimensions among unknown obstacles. Especially, we implemented the fully autonomous system with a real drone, showing the wide applicability in various scenes. For future work, we plan to incorporate the artistic objectives for filming multiple actors where we consider the relative location between the targets and the background.

REFERENCES

[1] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source slam system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[2] T. Qin, P. Li, and S. Shen, "VINS-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018.

[3] J. Li, Y. Wang, C. Wang, Y. Tai, J. Qian, J. Yang, C. Wang, J. Li, and F. Huang, "DSFD: Dual shot face detector," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5060–5069.

[4] A. Castillo, R. Sanz, P. Garcia, W. Qiu, H. Wang, and C. Xu, "Disturbance observer-based quadrotor attitude tracking control for aggressive maneuvers," *Control Eng. Pract.*, vol. 82, pp. 14–23, Jan. 2019.

[5] R. Bonatti, W. Wang, C. Ho, A. Ahuja, M. Gschwindt, E. Camci, E. Kayacan, S. Choudhury, and S. Scherer, "Autonomous aerial cinematography in unstructured environments with learned artistic decision-making," *J. Field Robot.*, vol. 37, no. 4, pp. 606–641, 2020.

[6] 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.

[7] 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. Automat. Lett.*, vol. 2, no. 3, pp. 1696–1703, Jul. 2017.

[8] 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.

[9] Z. Han, R. Zhang, N. Pan, C. Xu, and F. Gao, "Fast-tracker: A robust aerial system for tracking agile target in cluttered environments," 2020, *arXiv:2011.03968*. [Online]. Available: http://arxiv.org/abs/2011.03968

[10] 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.

[11] B. Brown, *Cinematography: Theory and Practice: Image Making for Cinematographers and Directors*. New York, NY, USA: Taylor & Francis, 2016.

[12] *DIJ Air 2S*. Accessed: Apr. 12, 2021. [Online]. Available: https://dl.djicdn.com/downloads/DJI_Air_2S/DJI_Air_2S_User_Manual_v1.0_en1.pdf

[13] Skydio. (2019). *Skydio Autonomy*. [Online]. Available: https://www.skydio.com/skydio-autonomy

[14] N. R. Gans, G. Q. 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.

[15] 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.

[16] F. Patrona, I. Mademlis, A. Tefas, and I. Pitas, "Computational UAV cinematography for intelligent shooting based on semantic visual analysis," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 4155–4159.

[17] N. Joubert, L. E. Jane, D. B Goldman, F. Berthouzoz, M. Roberts, J. A. Landay, and P. Hanrahan, "Towards a drone cinematographer: Guiding quadrotor cameras using visual composition principles," 2016, *arXiv:1610.01691*. [Online]. Available: http://arxiv.org/abs/1610.01691
[18] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Auton. Robot.*, vol. 34, no. 3, pp. 189–206, 2013.
[19] B. Lau, C. Sprunk, and W. Burgard, "Efficient grid-based spatial representations for robot navigation in dynamic environments," *Robot. Auton. Syst.*, vol. 61, no. 10, pp. 1116–1130, 2013. [Online]. Available: http://octomap.sf.net/
[20] Q.-C. Mao, H.-M. Sun, Y. B. Liu, and R.-S. Jia, "Mini-YOLOv3: Real-time object detector for embedded applications," *IEEE Access*, vol. 7, pp. 133529–133538, 2019.
[21] J. Zhang, C. Hu, R. G. Chadha, and S. Singh, "Maximum likelihood path planning for fast aerial maneuvers and collision avoidance," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 2805–2812.
[22] V. K. Viswanathan, E. Dexheimer, G. Li, G. Loianno, M. Kaess, and S. Scherer, "Efficient trajectory library filtering for quadrotor flight in unknown environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 2510–2517.
[23] S. Liu, N. Atanasov, K. Mohta, and V. Kumar, "Search-based motion planning for quadrotors using linear quadratic minimum time control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 2872–2879.
[24] B. F. Jeon, D. Shim, and H. Jin Kim, "Detection-aware trajectory generation for a drone cinematographer," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 1450–1457.
[25] D. Jungnickel and D. Jungnickel, *Graphs, Networks and Algorithms*. Berlin, Germany: Springer, 2005.
[26] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 2520–2525.
[27] S. Mehrotra, "On the implementation of a primal-dual interior point method," *SIAM J. Optim.*, vol. 2, no. 4, pp. 575–601, 1992.
[28] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "AirSim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics*. Berlin, Germany: Springer, 2018, pp. 621–635.
[29] Stereolabs. (2021). *ZED SDK*. [Online]. Available: https://www.stereolabs.com/docs/object-detection
[30] A. Geiger, M. Roser, and R. Urtasun, "Efficient large-scale stereo matching," in *Proc. Asian Conf. Comput. Vis.* Berlin, Germany: Springer, 2010, pp. 25–38.
[31] X.-F. Han, J. S. Jin, M.-J. Wang, W. Jiang, L. Gao, and L. Xiao, "A review of algorithms for filtering the 3D point cloud," *Signal Process., Image Commun.*, vol. 57, pp. 103–112, Sep. 2017.

BOSEONG FELIPE JEON received the B.S. degree in mechanical and aerospace engineering from Seoul National University, in 2017, where he is currently pursuing the Ph.D. degree in aerospace engineering. His research interests include autonomous cinematography and aerial systems. He received the guaranteed full scholarship funding from Samsung Research of Samsung's Consumer Electronics of South Korea.

YUNWOO LEE received the B.S. degree in electrical and computer engineering from Seoul National University, Seoul, South Korea, in 2019, where he is currently pursuing the integrated M.S./Ph.D. degree in mechanical and aerospace engineering. His current research interests include planning and control of unmanned vehicle systems.

JEONGJUN CHOI (Graduate Student Member, IEEE) received the B.S. degree in mechanical and aerospace engineering from Seoul National University, Seoul, South Korea, in 2021, where he is currently pursuing the integrated M.S./Ph.D. degree in aerospace engineering. His current research interests include mapping and exploration with unmanned aerial vehicles.

JUNGWON PARK (Graduate Student Member, IEEE) received the B.S. degree in electrical and computer engineering and the M.S. degree in mechanical and aerospace engineering from Seoul National University, Seoul, South Korea, in 2018 and 2020, respectively, where he is currently pursuing the Ph.D. degree in aerospace engineering as a member of the Lab for Autonomous Robotics Research under the supervision of H. Jin Kim. His current research interests include multi-agent path planning for UAVs in unknown environments. His work was a finalist of the Best Paper Award on Multi-Robot Systems in ICRA 2020.

H. JIN KIM (Member, IEEE) received the B.S. degree from Korea Advanced Institute of Technology (KAIST), in 1995, and the M.S. and Ph.D. degrees in mechanical engineering from the University of California at Berkeley (UC Berkeley), in 1999 and 2001, respectively. From 2002 to 2004, she was a Postdoctoral Researcher and a Lecturer in electrical engineering and computer science (EECS) at UC Berkeley. From 2004 to 2009, she was an Assistant Professor with the School of Mechanical and Aerospace Engineering, Seoul National University (SNU), Seoul, South Korea, where she is currently an Associate Professor. Her research interests include applications of nonlinear control theory and artificial intelligence for robotics, and motion planning algorithm.