2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-7281-6212-6/20/$31.00 ©2020 IEEE | DOI: 10.1109/IROS45743.2020.9341368

Detection-Aware Trajectory Generation for a Drone Cinematographer

Boseong Felipe Jeon, Dongsuk Shim and H. Jin Kim

***Abstract*— This work investigates an efficient trajectory generation for chasing a dynamic target, which incorporates the detectability objective. The proposed method actively guides the motion of a cinematographer drone so that the color of a target is well-distinguished against the colors of the background in the view of the drone. For the objective, we define a measure of color detectability given a chasing path. After computing a discrete path optimized for the metric, we generate a dynamically feasible trajectory. The whole pipeline can be updated on-the-fly to respond to the motion of the target. For the efficient discrete path generation, we construct a directed acyclic graph (DAG) for which a topological sorting can be determined analytically without the depth-first search. The smooth path is obtained in quadratic programming (QP) framework. We validate the enhanced performance of state-of-the-art object detection and tracking algorithms when the camera drone executes the trajectory obtained from the proposed method.**

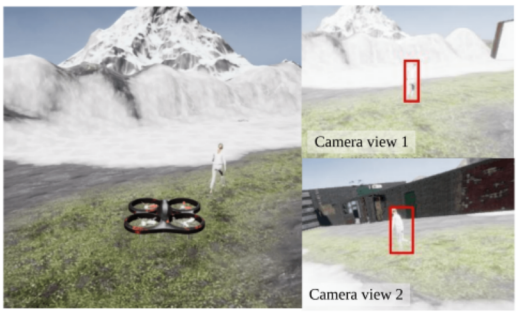

Fig. 1. **Left** : An illustration for autonomous chasing by cinematographer drone. Here, the drone is following the an actor with white clothes in a snow environment. **Rtight** : Image views from different observation bearing. The position of the actor is same for the two cases.

I. INTRODUCTION

Aerial cinematography where a flying agent is employed to autonomously follow a dynamic target is an important application of a drone equipped with vision sensors. From video filming for personal usage to industrial inspection, micro aerial vehicles (MAVs) have been successfully employed, which has been spurred by developments in control [1], planning [2], localization [3] and object tracking [4] and detection [5]. In such tasks, a camera drone should detect the target (or actor) of interest and localize its state, in order to determine the motion for chasing [6]. Thus, the localization of target is one of the most crucial modules. However, dynamic object detection and tracking by MAVs is challenging due to 1) occlusion from obstacles, 2) motion blur from the movement of both object and the drone 3) color ambiguity with background. Addressing 1)-3) becomes more important in the cases where a drone films an actor at some distance apart due to safety.

One option to circumvent the issues 1)-3) is to actively utilize the motion of drone rather than entirely relying on the image-processing technique such as target detection [5] or tracking [4], [7]. Inspired by the idea, a group of recent works [8]–[11] seeks to enhance the performance of target identification assisted by the motion strategy. The previous works [8], [9] of the authors proposed a chasing trajectory to improve the target perception against occlusion from obstacles. Also, the proposed motion strategy includes the travel efficiency and safety jointly. On the other side, the works [10], [11] also aim to enhance the perceptibility of target against the flight motion of camera drone. [10] leveraged model predictive control (MPC) to account for the motion efficiency and the perception objective so that the projected motion of a target on the image of a drone is minimized. In the case of [11], they analyzed the motion blur effect from the camera movement to alleviate blurring of the observed edges of a target of interest.

A. Target detectability in videography tasks

By guiding the movement of the camera drone, the mentioned works tried to improve the visual tracking performance focusing on occlusion and motion blur. In addition to them, color distinguishability of an actor from the background is also an important factor in localizing the object in the view of a drone. Technically, distinct separability in a color space is advantageous for the detection and tracking methods where the core objective is to build a classifier for an object of interest from backgrounds [7], [12], [13]. At the same time, detectability is also important for viewers and aesthetic of cinematography (the terms detectability and distinguishability will be interchangably used to refer the color separability of the target and the background). For example, it might be undesirable to constantly keep an actor wearing a white clothes on a white background (compare the two camera views in the right column of Fig. 1).

B. Contributions

In the field of image processing, there have been rich studies to analyze and circumvent the color ambiguities [14]–[16]. However, they deal with predetermined video footages, not an active motion strategy to acquire a better-detectable

*This material is based upon work supported by the Ministry of Trade, Industry & Energy(MOTIE, Korea) under Industrial Technology Innovation Program. No.10067206, 'Development of Disaster Response Robot System for Lifesaving and Supporting Fire Fighters at Complex Disaster Environment'

Department of mechanical and aerospace engineering, Seoul national university of South Korea {a4tiv,tlaehdtjr01,hjinkim}@snu.ac.kr

image sequence. In this work, we aim to bridge the gap between the research of motion planning of MAVs and resolving color ambiguity in image processing by presenting *detectability aware receding horizon planner* (DA-RHP). The proposed method generates a chasing path along which a target can be easily distinguishable from environments in terms of color. The contributions of our work can be summarized as the followings:

- We propose a detectability score metric for the motion planner. The metric can quantify how clearly the target can be distinguished from background given a RGB-channeled image.
- We build an efficient preplanning process leveraging a graph search method. From construction of a directed acyclic graph to which topological sorting can be determined analytically, we can reduce the computational load in finding an optimal solution.
- We test our algorithm validating the enhanced performance of well-known visual detection and tracking algorithms.

To the best knowledge of the authors, there is few studies in motion planning which efforts to generate a trajectory for chasing a dynamic object to improve the detectability using color information of a target and an environment.

II. Problem Statement

In this section, we put forward the assumptions and the aimed capabilities of DA-RHP. Regarding a camera drone, we assume that the allowable velocity is larger than a dynamic object and the drone has a vision sensor which can compute the state of the actor [6], [17]. For the ease of discussion, we assume the process is reliable enough once the object is successfully identified in the image of the drone. We will denote the pose observation of the target as $T_{a,i} \in SE(3)$ at the ith time step. In this work, the future movement of target over a time window is to be predicted from observations. We write them as $\{\hat{T}_{a,i}\}_{i=n+1}^{n+N}$ where N is the number of discretization over a horizon. Letting a RGB point cloud $\mathcal{P} = \{(\mathbf{x}, \mathbf{c}) \mid \mathbf{x} \in \mathbb{R}^3, \mathbf{c} \in \mathbb{C}\}$ where $\mathbb{C}$ is a set of RGB-triplets (r, g, b), we assume that the appearance models of the actor and the background are available as $\mathcal{P}_a$ and $\mathcal{P}_b$ respectively.

On top of the assumptions, we focus on an autonomous chasing framework as visualized in Fig. 2. On the drone side, it localizes and gathers the target observation from the visual information. Based on the observations, we predict the target motion and utilize it as an input of a receding horizon planner (RHP) for a local horizon. Then, RHP outputs a chasing trajectory as a control input for the MAV. The loop is continued until the end of the mission. In the pipeline, we focus on an online chasing strategy which aims at achieving 1) enhanced performance of object detectors and trackers along with human perception, 2) travel efficiency with dynamic feasibility for the drone, and 3) an efficient computation to be used as RHP to rapidly respond the behavior of target.

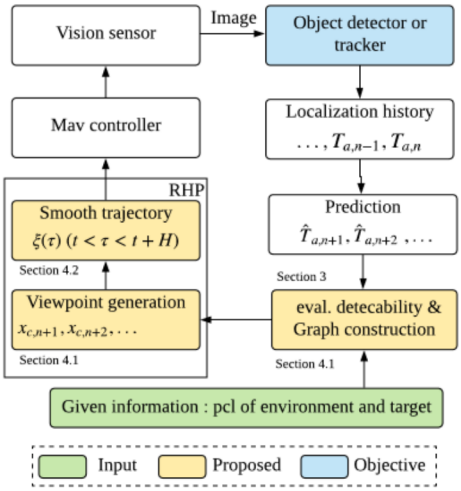

Fig. 2. The overall system description for the receding horizon planner. In the pipeline of the general autonomous target following framework, this work tackles the receding horizon planner (RHP) considering detectability given the RGB point cloud of background and target in order to enhance the performance of color-based detection and tracking algorithms.

A. Outline

To achieve the objectives, our planning strategy runs the three modules: detectability-evaluated graph generation, viewpoints optimization and continuous trajectory generation. We first consider a set of candidate viewpoints for a prediction $\hat{T}_{a,i}$ and evaluate the target detectability with respect to the viewpoints based on a metric. We introduce the metric in the upcoming section. Then, section IV-A discusses how to compute a sequence of viewpoints $\{\mathbf{x}_{c,i}\}_{i=n+1}^{n+N}$ given the prediction set $\{\hat{T}_{a,i}\}_{i=n+1}^{n+N}$, which optimizes the translational distance and the color detectability metric based on a directed acyclic graph search. We will also explore how to reduce the complexity exploiting the structure of the graph. As the last step, a dynamically feasible trajectory is obtained using $\mathbf{x}_{c,i}$ as a skeleton, which is explained in section IV-B.

III. Evaluation of Target Detectability

In this section, we describe the evaluation of detectability given a target prediction $\hat{T}_a$ captured in the image of the camera drone. For the purpose, we first quantify the color separability between the foreground and background, given an image which has an object of interest. In the field of image processing, the work [14] addressed this point in order to adaptively select a 1D feature space which best distinguishes between object and background. Based on the log-likelihood ratio of the two distributions on the foreground and background pixels, they measured the *variance ratio* as an extension of Linear Discriminant Analysis (LDA) to incorporate the multi-modality of the 1D features of an image.

While [14] proposed a well-suited metric for color detectability of a target, the original work appears to have two shortcomings which necessitate modifications for our scenario. First, recent object trackers and detectors such as

[4], [5], [18], [19] exploit multi-channeled feature space to improve accuracy, processing the multi-dimensional feature in real-time supported by the enhanced computing power. For those detection and tracking modules to be applicable to the camera drone, we will evaluate the color separability in the RGB space rather than limiting it into 1D feature space as [14] did.

Second, the original method [14] does not take spatial information of pixels into account when computing detectability, as only the color value of pixel is considered. As a result, the formulation in [14] might associate high detectability with an image which includes a small part of background area which has similar colors with target even though the pixels are very close to the target. As concrete examples, [14] gives a higher score to Fig. 4-(d) more than Fig. 4-(b) in terms of detectability although the overlapped region with background of the similar color is larger than Fig. 4-(b). In general, trackers such as [4], [7] perform template matching starting from the tracking result of the previous step. Thus, encoding the spatial proximity along with the color similarity is justifiable. We will consider not only the color but also the location of pixels in developing the detectability metric.

A. Color detectability computation

In order to define the detectability of the target projected in the image of the drone, we first consider a transformed pointcloud of the actor $\mathcal{P}_a$ with $\hat{T}_a \in SE(3)$, which can be written as

$$\mathcal{P}_a(\hat{T}_a) = \{(\hat{T}_a\mathbf{x}, \mathbf{c}) \mid (\mathbf{x}, \mathbf{c}) \in P_a\}. \tag{1}$$

Now, we can synthesize a projection image $I_s : \mathbf{u} \in \mathbb{R}^2 \rightarrow \mathbf{c} \in \mathbb{C}$ from the union of $\mathcal{P}_b$ and $\mathcal{P}_a(\hat{T}_a)$ given a camera pose and intrinsic parameters in a similar manner with [10]. In this work, I_s is assumed as an image which has pixels representing both target and background. Examples of rendered images in this way are depicted in (a) and (b) of Fig. 3. Naturally, we can identify whether an image coordinate $\mathbf{u}$ was projected from $\mathcal{P}_a(\hat{T}_a)$ or $\mathcal{P}_b$. We denote the set of $\mathbf{u}$ from $\mathcal{P}_a(\hat{T}_a)$ and $\mathcal{P}_b$ as U_a, $U_b \subset \mathbb{R}^2$ respectively. Now we are now ready to compute the detectability score of a rendered image I_s extending the metric proposed in [14].

We now proceed to create a RGB bin for the pixels $\mathbf{u}$ in a set $U \in \mathbb{R}^2$ of the I_s by assigning a discretized bin to each pixel $\mathbf{u} \in U$ based on the RGB-color value $\mathbf{c} = I_s(\mathbf{u})$. The resultant RGB-binning of U defines a map $\phi : \mathbb{C} \rightarrow [0, 1]$ which is written as

$$\phi(\mathbf{c} \mid U) = \frac{|Q(\mathbf{c} \mid U)|}{|U|} \tag{2}$$

where $|\cdot|$ is the cardinality of a set, and $Q(\mathbf{c} \mid U)$ is a set of pixels in U which are grouped into the same bin with a query color $\mathbf{c}$. The map (2) gives the density of the color value $\mathbf{c} \in \mathbb{C}$ in the RGB bin created from pixels U. Now a mono-scaled image $I_l : \mathbb{R}^2 \rightarrow \mathbb{R}$ is defined as

$$I_l(\mathbf{u}) = \log \frac{\max(\phi(\mathbf{c} \mid U_a), \epsilon)}{\max(\phi(\mathbf{c} \mid U_b), \epsilon)}, \tag{3}$$

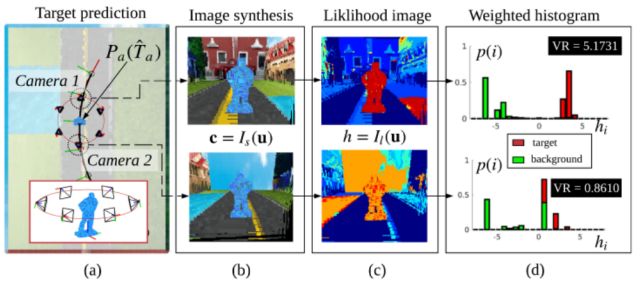

Fig. 3. Detectability evaluation process for a prediction pose of a target $\hat{T}_a$. (a) A sky-blue coloured target $\mathcal{P}_a$ with two different cameras observing it. (b) Rendered RGB images I_s of $\mathcal{P}_a(\hat{T}_a)$ and $\mathcal{P}_b$ from the two cameras. (c) log-likelihood image I_l for the two images. The mono-scaled image is shown in jet-colormap scale where red denotes high values and blue for low values. (d) Histograms of value in target pixels (red) and background pixels (green) for each I_l. It is noticed that the separability between the two distribution p_a and p_b is more outstanding for the images from *camera 1* than *camera 2*. The values of variance ratio of the two images are stamped in the black boxes.

where $\mathbf{c} = I_s(\mathbf{u})$ and ϵ is a small positive number to avoid numerical issues. Let us call I_l as *log-likelihood ratio image* or *likelihood image* for the brevity. Basically, $h = I_l(\mathbf{u})$ converts RGB triplets $\mathbf{c} = I_s(\mathbf{u})$ into a scalar h which encodes a likelihood that a given color $\mathbf{c}$ belongs to the foreground pixels U_a than background pixels U_b. Examples of I_l are illustrated in Fig. 3-(c) for synthesized images in Fig. 3-(b).

We now move on to creation of two histograms (i.e. 1D bin) for U_a and U_b in a likelihood image I_l. We write a normalized histogram value $p(i) \in [0, 1]$ for a log likelihood value $h_i \in \mathbb{R}$ of the ith bin, while representing the histograms of U_a and U_b as $p_a(i)$ and $p_b(i)$ respectively. Regarding data counting during histogram creation, an unweighted histogram adds an equal amount to the corresponding bin while a *weighted histogram* gives a weight when counting a data. Unlike [14] where all the histograms $p_a(i)$, $p_b(i)$ were an unweighted ones, we build an unweighted histogram for U_a and a weighted histogram for U_b. In building a weighted histogram for background, we assign a weight $w(\mathbf{u})$ to a likelihood value $h = I_l(\mathbf{u})$ of $\mathbf{u} \in U_b$, given by

$$w(\mathbf{u}) = \begin{cases} w_{max}(1 - \dfrac{d}{d_c}) + \dfrac{d}{d_c}, & d \leq d_c \\ 1. & d > d_c \end{cases} \tag{4}$$

where $d = \|\mathbf{u} - \overline{\mathbf{u}}_a\|$ and $\overline{\mathbf{u}}_a$ is the centroid of the target pixels U_a. That is, $w(\mathbf{u})$ linearly increases in proportion to the proximity with the target pixels if $\mathbf{u}$ is within a boundary d_c. Fig. 3-(d) visualizes the histograms $p_a(i)$ (red) and $p_b(i)$ (green) from likelihood images. As the last phase to define the metric for detectability, we quantify the separability between the two distributions p_a and p_b by computing a variance ratio $R(I_s)$ of *between-class variance* to the sum of *within-class variances* given by

$$R(I_s) = \frac{\mathrm{Var}(h; (p_a + p_b)/2)}{\mathrm{Var}(h; p_a) + \mathrm{Var}(h; p_b)} \tag{5}$$

Fig. 4. The evaluations of detectability with $R(I_s)$ for several footages of the scenario Fig. 3-(a). We linearly scaled the value of $R(I_s)$ to $[1, 10]$ for 4 cases (a)-(d) so that (a) has the highest score 10. The green labels with numbers show the score using weighted histogram proposed in this work while the blue labels shows scores acquired from [14].

where $\mathrm{Var}(h; p)$ is a variance of likelihood h with respect to the probability distribution from a histogram p. The metric (5) captures how tightly the likelihood values $h = I_l(\mathbf{u})$ of the target and background pixels are clustered within each group (denominator), and how much the two are spread apart (numerator). As (5) quantifies the separability of the target-likelihood distributions of target pixels and background pixels, we utilize (5) as a score for detectability. The raw values $R(I_s)$ of the two images from *camera 1* and *camera 2* are stamped in Fig. 3-(d) in the black-labelled boxes. We can notice that the high-scored camera view in Fig. 3 (camera 1) has more distinct background color against the target, which is also reflected as the large separation between the histograms of target and background.

B. Discussion of the proposed metric

In developing the metric for target detectability based on the distributions of pixels of the background and the foreground, there were two modifications from [14]. First, we have utilized the full-color information of pixels by creating a three-dimensional bin when computing the log-likelihood ratio. From this, we were able to define separability in RGB color space which is frequently utilized as the raw input in the state-of-the-art algorithms for detection and tracking [4], [13], [20]. Second, we were able to exploit the spatial information of pixels in (4) by building a weighted histogram for the background pixels. Examples of evaluations of $R(I_s)$ of ours versus [14] are compared in Fig. 4. Both methods give the highest score for Fig. 4-(a). However, [14] gives a high score for Fig. 4-(d) as the image has only small amount of dubious colors despite of their close proximity to the target pixels. In contrast, the weighted histogram proposed in this work gives the lower scores to Fig. 4-(c) and (d) considering the locations of the pixels of similar colors.

IV. Detection-Aware Trajectory Generation

A. Computation of optimal viewpoints

In the previous section, we defined a detectability metric as the variance ratio (5) starting from a synthesized image I_s given a camera viewpoint. Making use of the scoring metric $R(I_s)$ and the initial position of chaser $\mathbf{x}_{c,0} = \mathbf{x}_c(t_0) \in \mathbb{R}^3$, we explain how to plan discrete view points $\sigma = \{\mathbf{x}_{c,i}\}_{i=1}^N$ $(\mathbf{x}_{c,i} \in \mathbb{R}^3)$ for a prediction sequence $\{\hat{T}_{a,i}\}_{i=1}^N$ along which the accumulative detectability is maximized

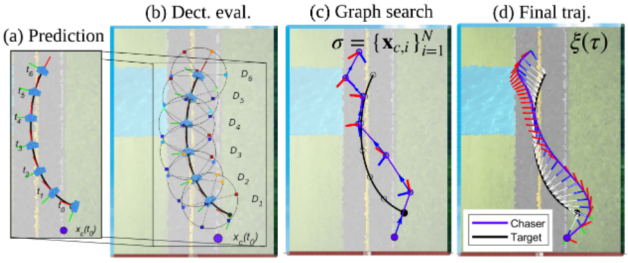

Fig. 5. Trajectory generation process. (a) A six-step prediction sequence for the blue-coloured target with the initial location of chaser $\mathbf{x}_{c,0}$ (purple circle) (b) For each prediction, the view sphere D_i is composed of six candidate viewpoints. The color of each viewpoint shows the scale of detectability score with jet-colormap. We build a DAG on top of the set of viewpoints having $x_c(t_0)$ as a root. (c) The graph search result for discrete viewpoints at each time step. (d) The final trajectory. The optical axis (z) and x-axis of drone-body coordinate are stamped with bearing vector (white arrow).

while the total travel is reduced. From this, we will use $\mathbf{x}_{c,i}$ as a planned position for the chaser at time t_i with the optical axis toward a target prediction $\hat{T}_{a,i} = \hat{T}_a(t_i)$.

As a requirement of σ, we constrain the Euclidean distance of the predicted target and the chaser with $r_d \in \mathbb{R}^+$. Also, we bound the inter-distance of points $\mathbf{x}_{c,i}, \mathbf{x}_{c,i+1}$ at steps t_i and $t_i + \Delta t$ below r_{max}, assuming that the velocity larger than $r_d/\Delta t$ is undesirable. Coupling the two constraints with the detectability objective, we formulate an optimization as below.

$$\begin{aligned} \min_{\sigma} \quad & \sum_{i=0}^{N-1} \underbrace{\|\mathbf{x}_{c,i} - \mathbf{x}_{c,i+1}\|}_{\text{distance}} + \lambda \sum_{i=1}^{N} \underbrace{L(\mathbf{x}_{c,i}|\hat{T}_{a,i})}_{\text{detection}} \\ \text{subject to} \quad & \|\mathbf{x}_{c,i} - \mathbf{x}_{a,i}\| = r_d \\ & \|\mathbf{x}_{c,i} - \mathbf{x}_{c,i+1}\| \le r_{max}, \end{aligned} \tag{6}$$

where $L(\mathbf{x}_{c,i}|\hat{T}_{a,i})$ is a positive-valued cost function which penalizes a low value of $R(I_s)$ where I_s is the rendered image of $\mathcal{P}_a(\hat{T}_{a,i})$ by the camera with center $\mathbf{x}_{c,i}$ and optical axis $\mathbf{x}_{a,i} - \mathbf{x}_{c,i}$ (we represented $\mathbf{x}_{a,i} \in \mathbb{R}^3$ as a translation of $\hat{T}_{a,i}$). λ is the importance weight for detectability. Instead of searching $\mathbf{x}_{c,i}$ on the entire half-sphere with a center $\mathbf{x}_{a,i}$ and radius r_d to solve (6), we compute $\mathbf{x}_{c,i}$ in a finite set D_i which is a set of discretized points satisfying $\|\mathbf{x} - \mathbf{x}_{a,i}\| = r_d$ (see the surrounding cameras in Fig. 3-(a) as an example).

Now we consider a directed acylic graph (DAG) $G_d = (V, E)$ to find an optimal $\sigma = \{\mathbf{x}_{c,i}\}_{i=1}^N$ on a sequence of the finite set $\mathbf{x}_{c,i} \in D_i$. Setting $V_0 = D_0 = \{\mathbf{x}_{c,0}\}$, we define the set of vertices V as

$$V = \bigcup_{i=0}^{N} V_i \ , \ V_i \subset D_i \tag{7}$$

where V_i is a subset of D_i. The graph G_d is constructed as follows. First, we initialize $V_i = D_i$. Then, we wire two nodes $\mathbf{x}_{c,i} \in D_i$ and $\mathbf{x}_{c,i-1} \in D_{i-1}$ as a directed edge $e = (\mathbf{x}_{c,i-1}, \mathbf{x}_{c,i})$ if the condition $\|\mathbf{x}_{c,i-1} - \mathbf{x}_{c,i}\| \le r_{max}$ is met. Then, we collect the edge to E. After all the admissible

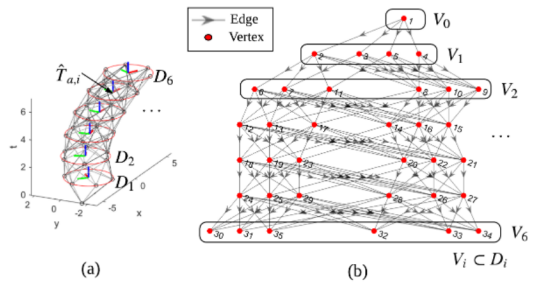

Fig. 6. (a) An illustration of graph construction from view spheres of Fig. 5-(b). Due to the connection limit r_{max}, we have two unreachable points in D_1 from the root $\mathbf{x}_{c,0}$ resulting in only 4 vertices in V_1. (b) The structure of DAG made from (a). We perform a graph search to find an optimal sequence from the root to all the possible destinations in V_6.

edges are found, V_i is reset to have only reachable elements $\mathbf{x}_{c,i} \in D_i$ from the root $\mathbf{x}_{c,0}$. A structure of G_d for the case Fig. 5-(a) is visualized in Fig. 6. We now define an weight L_i for $e = (\mathbf{x}_{c,i-1}, \mathbf{x}_{c,i})$ as

$$L_i = \|\mathbf{x}_{c,i-1} - \mathbf{x}_{c,i}\| + \lambda L(\mathbf{x}_{c,i}|\hat{T}_{a,i}). \tag{8}$$

Thus, applying a graph-search method for G becomes equivalent to solving (6). Performing a graph-search on a general DAG is composed of two steps: 1) *topological sorting* and 2) *edge relaxation* [21]. Here, we introduce the following definitions before discussing the topological sorting for the graph G_d.

Definition 1. *A sorting S for a set A is defined as a bijective function $S : A \to \{n|1 \le n \le |A|\}$.*

Definition 2. *A topological sorting for a directed acyclic graph $G = (V, E)$ is a sorting $S_T : V \to \{n|1 \le n \le |V|\}$ satisfying $S_T(u) < S_T(v)$ given any edge $e = (u, v) \in E$ where $u, v \in V$.*

The time complexity required for topological sorting in a general DAG is $O(|V| + |E|)$ when using depth-first search DFS with an extra stack [22]. In the case of G_d, however, we can directly determine a topological sorting S_T. For the enumeration purpose, we represent V_i as $\{\mathbf{x}^j_{c,i}\}_{j=1}^{n_i}$ where n_i is cardinality of V_i. We determine the sorted index for a vertex $\mathbf{x}^j_{c,i}$ as below:

$$S_T(\mathbf{x}^j_{c,i}) = \begin{cases} \sum_{k=0}^{i-1} n_k + j & i > 0, \\ 1 & i = 0. \end{cases} \tag{9}$$

Now we show (9) is a topological sorting by **Proposition 1**.

Proposition 1. *The sorting of* (9) *is a topological sorting for G_d.*

Proof. Following the wiring process mentioned above, every edge $e \in E$ can be written as $e = (\mathbf{x}^{j_1}_{c,i}, \mathbf{x}^{j_2}_{c,i+1})$ where $\mathbf{x}^{j_1}_{c,i} \in V_i$, $\mathbf{x}^{j_2}_{c,i+1} \in V_{i+1}$ and $1 \le j_1 \le n_i,\ 1 \le j_2 \le n_{i+1}$.

From this, $\sum_{k=0}^{i-1} n_k < S_T(\mathbf{x}^{j_1}_{c,i}) \le \sum_{k=0}^{i} n_k$ and $\sum_{k=0}^{i} n_k < S_T(\mathbf{x}^{j_2}_{c,i}) \le \sum_{k=0}^{i+1} n_k$ hold. As $S_T(\mathbf{x}^{j_1}_{c,i}) < S_T(\mathbf{x}^{j_2}_{c,i})$ is satisfied for every edge, S_T is a topological sorting for G_d. ■

This decreases the computation time by sparing us the need for an extra algorithm for topological sorting. Based on the vertex enumeration from (9), a dynamic programming [22] can compute the shortest path to every node of G_d from the root node using only edge relaxation step whose complexity is linear to the number of edges. Thus, a discrete path $\sigma = \{\mathbf{x}_{c,i}\}_{i=1}^{N}$ which optimizes (6) can be found by identifying the minimum cost among all the pairs $(\mathbf{x}_{c,0}, \mathbf{x}^j_{c,N})$ from the root to vertices in V_N. An example of G_d is visualized in Fig. 5-(c), which was generated from a prediction $\{\hat{T}_{a,i}\}_{i=1}^{N}$ and corresponding view sphere $\{D_i\}_{i=1}^{N}$ of Fig. 5-(a) and (b). To examine how the deterministic sorting (9) can reduce the computation time for the overall graph search process for G_d, we compared our implementation with a Matlab built-in function which performs the topological sorting equipped with additional search algorithm. The result is as shown in Fig. 7.

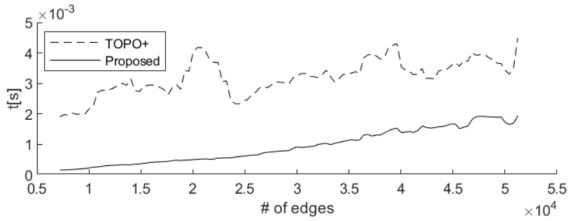

Fig. 7. Computation time to solve *the single-source-shortest-path (SSSP)* for the graphs of the same structure with G_d by increasing the number of edges. The dashed line denotes the result of applying MATLAB `shortestpathtree` function to our the graphs, which performs topological sorting using a search-based method. The thick black line shows the result when the sorting process was replaced with (9).

B. Dynamically feasible trajectory for drones

The formulation (6) and the graph-search have planned a detectability enhanced viewpoint $\mathbf{x}_{c,i}$ at every time step t_i, assuming that target's pose will be $\hat{T}_{a,i}$. Based on the initial sketch $\sigma = \{\mathbf{x}_{c,i}\}_{i=1}^{N}$ and its initial state $\mathbf{x}_{c,0}$, $\dot{\mathbf{x}}_{c,0}$ and $\ddot{\mathbf{x}}_{c,0}$, we will finalize a continuous trajectory consisting of position and yaw $\xi(\tau) = [x(\tau)\ y(\tau)\ z(\tau)\ \psi(\tau)]^T \in \mathbb{R}^4$ of the camera drone for $t_0 \le \tau \le t_N$. For the position $\mathbf{x}_c(\tau) = [x(\tau)\ y(\tau)\ z(\tau)]^T \in \mathbb{R}^3$, we represent the trajectory with a polynomial spline curve

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

where $\mathbf{p}_{i,k} \in \mathbb{R}^3$ denotes the polynomial coefficient for order k defined over time segment $[t_i, t_{i+1})$. We want to calculate the polynomials which effort to pass through the viewpoints

$\mathbf{x}_{c,i}$ at time t_i while decreasing the magnitude of the high-order derivative for smooth transition. For the purposes, an optimization can be formulated as

$$\begin{aligned} \min \quad & \int_{t_0}^{t_N} \left\| \mathbf{x}_c{}^{(3)}(\tau) \right\|^2 d\tau \; + \; \rho \sum_{i=1}^{N} \left\| \mathbf{x}_c(t_i) - \mathbf{x}_{c,i} \right\|^2 \\ \text{subject to} \quad & \mathbf{x}_c(t_0) = \mathbf{x}_{c,0} \\ & \dot{\mathbf{x}}_c(t_0) = \dot{\mathbf{x}}_{c,0} \\ & \ddot{\mathbf{x}}_c(t_0) = \ddot{\mathbf{x}}_{c,0} \end{aligned} \tag{11}$$

This can be converted to a quadratic programming with respect to $\mathbf{p}_{i,k}$ in a similar way with [23], which can be solved efficiently using the algorithms such as interior point [24]. Fig. 5-(d) demonstrates a smooth curve obtained from the viewpoints in Fig. 5-(c). Regarding the trajectory of $\psi(\tau) \in \mathbb{R}$, we decide it by heading the optical axis of drone to the actual center of the actor in a myopic way. Based on the dynamics and the differential-flatness of a quadrotor model, the trajectory $\xi(\tau) = [x(\tau)\ y(\tau)\ z(\tau)\ \psi(\tau)]^T$ can be executable with dynamic-feasibility for MAVs within the actuation limits [25]. $\xi(\tau)$ is fed into the controller of the drone until a prediction is reliable or the defined period expires (see the final input to MAV controller in Fig. 2).

Up to now, we have explained the two steps to generate a detection-aware chasing trajectory: 1) detectability-optimized viewpoints generation and 2) smooth trajectory generation. We now put them together to build DA-RHP as summarized in **Algorithm 1**.

Algorithm 1: DA-RHP

Input : Pointclouds $\mathcal{P}_a$, $\mathcal{P}_b$. Mission time $[t_s, t_f]$
Initialize : Accumulated prediction error; `accumErr` $= 0$

for $t = t_s$ **to** t_f **do**
  if `accumErr` $> \epsilon$ **then**
    Set window $\tau \in [t, t+H]$, discretization $\{t_n, t_{n+1}, ..., t_{n+N}\}$ and $\mathbf{x}_{c,n} = \mathbf{x}_c(t)$.
    $\{\hat{T}_{a,i}\}_{i=n+1}^{n+N}$ = `Predict` $(\{t_i\}_{i=n+1}^{n+N})$
    Eval. detect. for `ViewSphere` $(\hat{T}_{a,i})$
    $\{\mathbf{x}_{c,n+1}, \mathbf{x}_{c,n+2}, ..., \mathbf{x}_{c,n+N}\}$=`DAGSearch` ()
    $\xi(\tau)$ = `smoothTraj` $(\{\mathbf{x}_{c,i}\}_{i=n}^{n+N})$
    `accumErr` $= 0$
  end
  update `accumErr`
  exec. $\xi(t)$
end

V. Validations

In this section, we validate the proposed planner DA-RHP by comparing it with a plain chasing strategy in a simulated environment. Several key results are presented including computation time, travel distance and the performance of visual identification of a target when using correlation filter-based trackers [4], [7] and learning-based object detection methods [13], [20] for the output video footage taken from the drone.

A. Experimental setup

We performed high-fidelity simulations in *Unreal engine* with Airsim plugin [26] where a drone with vision sensors is supposed to autonomously follow a walking actor with white clothes. In the environment, there are multiple piles of snow which can interrupt the visual tracking of the actor as shown in Fig. 1. The actor is set to walk with the maximum speed 1.3 m/s along the path shown as ruby curve in the first column of Fig. 8. We use constant velocity model for prediction over a short prediction horizon. The drone was mounted with a firmly attached camera having 120° field-of-view. The whole algorithm in **Algorithm 1** was written in C++, and ROS was used to operate the simulated drone via px4 SITL. Additionally, we applied QPOASES to compute (11). All the computations including running the simulator were performed in a computer with CPU Intel i7-6700K CPU @ 4.00GHz and RAM 16GB.

Based on them, we perform two simulations. One is a plain chasing planner without detectability consideration and the other is DA-RHP. The only difference is that the former sets the detection importance λ to zero while the latter to 20 in (6). Other than that, the same parameters were applied: the time horizon $H = 4$ and time discretization $N = 4$ were used, while the number of elements in the view-sphere was set to 8. For the smooth trajectory, the order of polynomial spline was set to $K = 5$. The observation distance was chosen as $r_d = 5$ m. Around $200,000$ points were included in $\mathcal{P}_b$ for the environment in Fig. 8, and the total duration of the mission is 60 s. From the two simulations, we compare the translation histories and the sequences of images from the drone (see Fig. 8). In the simulations, we fed the current pose information of the target to the follower drone rather than calculating the target's pose from the images. This was to observe how the receding horizon planners operate for the entire duration, not to be disturbed the tracking failure. In fact, without such setup, the planner without detectability consideration will malfunction due to tracking failures.

B. Results

First, we assess the performance of target detectability of DA-RHP based on the two object detection (DSFD [13], YOLOv3 [20]) and two tracking algorithms (CSR-DCF [4], KCF [7]). To measure the detection accuracy for the four algorithms, the history of IOU (intersection over union) is analyzed as shown in Fig. 9 and Table I. Additionally, the average precision (AP) of the neural networks is presented in the case of detection algorithms as analyzed in the literature such as [13], [20].

For both trackers, the tracking was improved when the target was filmed from DA-RHP, resulting in longer duration with reliable IOU (≥ 0.4) than the plain RHP. In the case of the plain RHP, KCF tracker and CSR-DCF started to lose the accuracy from t_2 and t_3 respectively as shown in Fig. 8 when they encountered the snow background. In contrast,

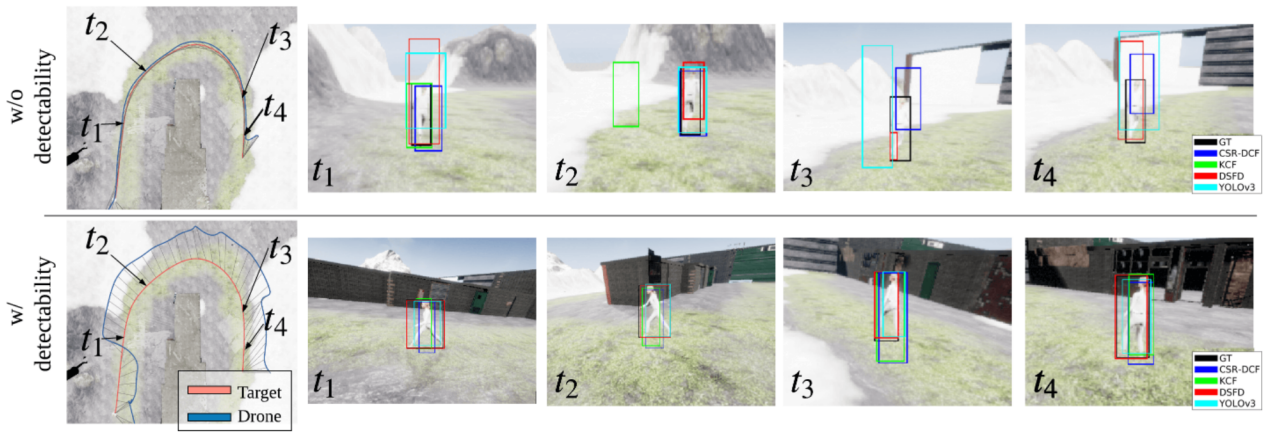

Fig. 8. The positional histories (left) and video footage ($t_1, .., t_4$) of the two motion strategies. In the image sequence, the detection boxes for the actor are visualized for 4 different algorithm plus the ground truth. **Top**: A plain-chasing strategy. **Bottom**: The proposed DA-RHP.

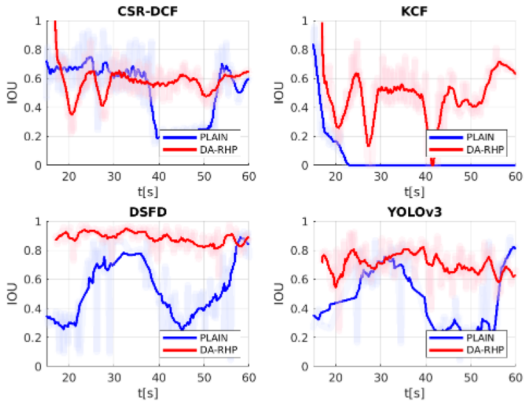

Fig. 9. The IOU (intersection over union) results from the simulation with two tracking algorithms (CSR-DCF, KCF) and two detection algorithms (DSFD, YOLOv3). For each algorithm, the IOU is computed for the footage taken along the trajectory generated from either the proposed (red) or plain receding horizon planner (blue).

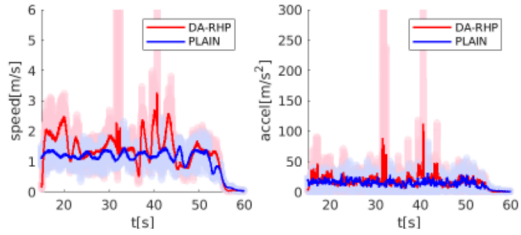

Fig. 10. (a) The histories of speed of the drone during the simulations when the drone executes DA-RHP (red) and plain-RHP (blue). (b) Norm of acceleration results in the simulation.

DA-RHP took a detour to maintain the actor in front of the brick walls avoiding snow backgrounds. The averages of IOU are summarized in Table I.

To validate our algorithm with detection algorithms YOLOv3 and DSFD, we applied the deep neural networks originally designed for multi-object classification to a single-class setting for our target detection scenario. With supervised learning, each network was trained with 500 RGB images taken from the drone observing the actor at various locations and bearing direction. This was intended to to mimic a human perception test where a human subject is shown many figures of the actor and requested to segment the actor from the two different video footage from the cinematographer drone. The training set also included highly-ambiguous footages. Then, we tested the footages from the drone during the simulations. To record IOU, we included only the images throttled by the network output confidence 0.5. The results show the averaged IOU of DA-RHP was higher than the plain strategy as noted in Table I.

Regarding the length of the chasing path, DA-RHP traveled 18 m longer than the plain chasing planner while giving more bearing views advantageous for detectability performance. The average computation time was 220 ms, 1 ms and 3 ms for detectability evaluation, graph-search and quadratic programming respectively. The total pipeline ran at 4-5 Hz showing the real-time performance to be used as a receding horizon planner.

C. Discussion

As mentioned above, most of the computation time was spent on the image synthesis process with respect to all the camera poses in view-sphere. In our implementation, image projection of the point cloud $\mathcal{P}_b$ handled self-occlusion of the objects (e.g. a brick house in the center Fig. 8) in the environment to deal with more general situations. In a simpler scenario without such objects, we can reduce the computation time by omitting the occlusion-culling. It is also noteworthy that DA-RHP showed a slightly degraded detection and tracking performance around 20 s and 40 s as can be seen in Fig. 9. These are associated with an increased velocity and acceleration as visualized in Fig. 10. We found that the motion guidance to obtain a high-detectability had caused a perceptible change of the location and the scale of the target in the image view along with motion blur, resulting in low accuracy during a short period. In this case, increasing the number of discretization of view spheres or reducing

r_{max} can mitigate abrupt motions of the drone. On the hardware side, utilizing a sensor with wider FOV or gimbal stabilization can be an option. Also, we found that the nature of DA-RHP to observe more distinguishable background often changes the relative bearing (see the bottom of Fig. 8), which might make tracking of the target challenging. For example, the case with the CSR-DCF tracker (top left of Fig. 9) indicates the temporarily poorer IOU performance of DA-RHP than the plain method, up to around 40 sec (t_3 in Fig. 8) when the snow misled the tracker. However, it is because the relative view angle of the plain method remained almost same in the top-left figures of Fig. 8 during that interval.

	Tracking (IOU)		Detection (IOU/AP)		dist. [m]
	KCF [7]	CSR-DCF [4]	DSFD [13]	YOLO [20]	
plain RHP	0.0522	0.5175	0.5332/ 0.0.3197	0.5089 / 0.1879	73
DA-RHP	0.4663	0.5725	0.8876/ 0.7313	0.7042 / 0.3277	91

TABLE I

PERFORMANCE OF DETECTABILITY

VI. CONCLUSION AND FUTURE WORKS

In this work, we presented a detectability metric and a chasing motion strategy named DA-RHP which jointly embraces the color distinguishability and total travel distance. In almost-real simulations, we validated the enhanced performance of the object identifiers by comparing DA-RHP with a plain chasing planner without detectability consideration. We also confirmed the reduced complexity in the graph search method from analytical topological sorting, validating the online performance. As future works, we will extend the proposed algorithm into real-world scenarios, validating the color detectability metric in various dataset. Especially, the collision safety will be included by applying the approach proposed in the previous works of the authors [9], [27]. Also, we will consider cases where the camera drone receives the point cloud information on-the-fly, enabling the capability to handle dynamic environments.

REFERENCES

[1] M. Neunert, C. de Crousaz, F. Furrer, M. Kamel, F. Farshidian, R. Siegwart, and J. Buchli, "Fast nonlinear model predictive control for unified trajectory optimization and tracking," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1398–1404, May 2016.

[2] M. Ryll, J. Ware, J. Carter, and N. Roy, "Efficient trajectory planning for high speed flight in unknown environments," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 732–738, IEEE, 2019.

[3] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[4] A. Lukezic, T. Vojir, L. Cehovin Zajc, J. Matas, and M. Kristan, "Discriminative correlation filter with channel and spatial reliability," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6309–6318, 2017.

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[6] R. Bonatti, C. Ho, W. Wang, S. Choudhury, and S. Scherer, "Towards a robust aerial cinematography platform: Localizing and tracking moving targets in unstructured environments," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 229–236, Nov 2019.

[7] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 3, pp. 583–596, 2014.

[8] B. F. Jeon and H. J. Kim, "Online trajectory generation of a mav for chasing a moving target in 3d dense environments," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1115–1121, Nov 2019.

[9] B. F. Jeon, Y. Lee, and H. Kim, "Integrated motion planner for real-time aerial videography with a drone in a dense environment," in *International Conference on Robotics and Automation (ICRA). To appear*, IEEE, 2020.

[10] D. Falanga, P. Foehn, P. Lu, and D. Scaramuzza, "Pampc: Perception-aware model predictive control for quadrotors," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–8, IEEE, 2018.

[11] R. Katoch and J. Ueda, "Edge-preserving camera trajectories for improved optical character recognition on static scenes with text," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4467–4474, 2019.

[12] S. Avidan, "Support vector tracking," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 8, pp. 1064–1072, 2004.

[13] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5060–5069, 2019.

[14] R. T. Collins, Y. Liu, and M. Leordeanu, "Online selection of discriminative tracking features," *IEEE transactions on pattern analysis and machine intelligence*, vol. 27, no. 10, pp. 1631–1643, 2005.

[15] M. Kristan, J. Perš, A. Leonardis, and S. Kovačič, *Probabilistic tracking using optical flow to resolve color ambiguities*. na, 2007.

[16] Y. Ueda, T. Koga, N. Suetake, and E. Uchino, "Color quantization method based on principal component analysis and linear discriminant analysis for palette-based image generation," *Optical Review*, vol. 24, no. 6, pp. 741–756, 2017.

[17] H. Song, W. Choi, S. Lim, and H. Kim, "Target localization using rgb-d camera and lidar sensor fusion for relative navigation," in *2014 CACS International Automatic Control Conference (CACS 2014)*, pp. 144–149, Nov 2014.

[18] M. Luber, L. Spinello, and K. O. Arras, "People tracking in rgb-d data with on-line boosted target models," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3844–3849, Sep. 2011.

[19] X. Lan, M. Ye, S. Zhang, H. Zhou, and P. C. Yuen, "Modality-correlation-aware sparse representation for rgb-infrared object tracking," *Pattern Recognition Letters*, 2018.

[20] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[21] D. Jungnickel and D. Jungnickel, *Graphs, networks and algorithms*. Springer, 2005.

[22] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2009.

[23] J. Chen, T. Liu, and S. Shen, "Tracking a moving target in cluttered environments using a quadrotor," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 446–453, IEEE, 2016.

[24] S. Mehrotra, "On the implementation of a primal-dual interior point method," *SIAM Journal on optimization*, vol. 2, no. 4, pp. 575–601, 1992.

[25] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE International Conference on Robotics and Automation*, pp. 2520–2525, IEEE, 2011.

[26] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and service robotics*, pp. 621–635, Springer, 2018.

[27] B. F. Jeon and H. J. Kim, "Online trajectory generation of a mav for chasing a moving target in 3d dense environments," *arXiv preprint arXiv:1904.03421*, 2019.