Received November 4, 2020, accepted November 28, 2020, date of publication December 3, 2020, date of current version December 15, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3042258

Reactive Collision Avoidance Algorithm for UAV Using Bounding Tube Against Multiple Moving Obstacles

JONGHO PARK[1], NAMHOON CHO[2], (Member, IEEE), AND SEOKWON LEE[3]
[1]Department of Military Digital Convergence, Ajou University, Suwon 16499, Republic of Korea
[2]The 1st Research and Development Institute, Agency for Defense Development, Daejeon 34186, Republic of Korea
[3]School of Aerospace, Transport and Manufacturing, Cranfield University, Cranfield MK43 0AL, U.K.

Corresponding author: Seokwon Lee (blueswl12@gmail.com)

This work was supported by the new faculty research fund of Ajou University.

ABSTRACT A reactive collision avoidance algorithm is proposed to enable safe autonomous flight in the presence of multiple dynamic obstacles. A position-controlled hexacopter equipped with a visual sensor obtaining obstacle information is considered as an Unmanned Aerial Vehicle (UAV) platform. The proposed method centers on the concept of bounding tube which intrinsically extends the static bounding box to incorporate forthcoming movement of the obstacles into the collision avoidance framework. The processing pipeline consists of separate components for each of the sequential tasks in obstacle sensing and tracking. Computation of a spherical bounding box for each obstacle is followed by discrete-time Kalman filtering for prediction of obstacle trajectory to detect potential collision. If the current course of UAV turns out highly likely to end in collision with any of the obstacles, the vehicle steers to an aiming point chosen from among the bundle of candidates produced by constructing a bounding tube that takes account of predicted obstacle motion. The bounding-tube-based aiming point generation extends seamlessly to the case with multiple moving obstacles through running in series with multi-obstacle track management that combines hierarchical clustering of sensory data points for obstacle identification and a simple geometric method for data association. Numerical simulations are conducted to verify the performance of the proposed collision avoidance algorithm.

INDEX TERMS Reactive collision avoidance, guidance algorithm, multiple obstacles, dynamic environment, unmanned aerial vehicle.

I. INTRODUCTION

Recent development of the autonomy and aerospace technologies has brought increased opportunities for Unmanned Aerial Vehicles (UAVs) to replace manned operations in the civil and military applications. In particular, small rotary-wing UAVs have great potential for a wide range of areas as the multi-rotor configuration provides Vertical Take-Off and Landing (VTOL) and hovering capabilities with a high thrust-to-weight ratio. The multi-rotor UAVs have shown promises in various fields such as surveillance, logistics, and cooperative operation. For instance, the logistics industry has demonstrated autonomous delivery services using the UAVs. However, a rapid increase in the number of aircraft causes dense air traffic and may increase the risk of collision. The necessity of an effective and efficient collision avoidance strategy thus poses a major challenge to the integration of UAVs into the controlled airspace. In particular, the current regulatory framework requires UAVs to adopt a sense-and-avoid technology so that the UAVs detect and avoid the risk of collision as in the way manned aircrafts resolve conflicts [1]–[3].

Reactive collision avoidance for the UAV operations has been widely investigated for decades [4]. In reactive collision avoidance, the UAV decides and executes the collision avoidance maneuver online based on the information of local surroundings obtained by onboard sensors. It allows for rapid response to sudden changes in the environment and requires little computational efforts and prior knowledge of the configuration space. From these features,

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

the reactive approach has many benefits in the dynamic environment compared to the proactive planning approach where the UAV incorporates map information to avoid collision with known obstacles in the path planning level, e.g., sampling-based path planning [5], [6] and optimization-based methods [7]–[9]. Vector field approach has been studied as one of the ways of reactive collision avoidance where it utilizes potential functions that consist of repulsive or attractive force fields repelling a UAV from an obstacle or attracting it towards a predefined goal point [10]–[16]. Santos *et al.* [10] proposed a trajectory tracking controller based on a time-variant artificial potential field, where the obstacle motion was modeled as the variations of the potential function. Sun *et al.* [11] proposed an optimization process into the artificial potential field algorithm to improve collision avoidance performance in dynamic environment, and Du *et al.* [12] proposed a vector-field-based method that is appropriate for real-time application. The vector field approach has strengths in practice as it is intuitive and easy to implement. However, it is well known that the necessity to treat the local minima adds complications to the implementation of the approach.

Geometry-based approaches have also been extensively investigated in the context of reactive collision avoidance [17]. In the geometric approach, the collision detection based on predicted trajectories of the vehicle and obstacles precedes maneuver planning. Various guidance schemes including intercept angle control [18], [19] and proportional navigation [20], [21] have been proposed to produce an avoidance maneuver to escape from the danger zone. Specifically, the collision-cone-based approach has attracted interests of many researchers. This approach determines a collision condition by the relation between the obstacle position and the UAV position, and the direction of the UAV velocity. Chakravarthy and Ghose [22] applied the collision cone approach which is appropriate for dynamic obstacle environment and also generalized the collision cone concept to three-dimensional environment [23]. Many other studies extended the collision cone concept to the vision-based collision avoidance [24]–[29], collision avoidance in dynamic environment [30], [31], and formation flight [32]. Despite the significant benefits, most of the studies on the collision cone approach consider only a single obstacle. Therefore, the collision cone based methods may not provide satisfactory solution in the presence of multiple moving obstacles. On the other hand, aiming point approach [33] can offer an effective solution to cope with moving obstacles. In the aiming point approach, possible points of avoidance are collected, and an avoidance maneuver is performed by following the aiming point selected among the candidates. Mujumdar and Padhi [34] presented a nonlinear geometric guidance method that incorporates aiming point to avoid moving obstacles. A sphere-shaped boundary from the obstacle was formed to impose additional safety, and the point of closest approach [35] was adopted to choose the aiming point among the boundary. In a more dynamic environment, however, the sphere-tracking algorithm should be enhanced by considering the movement of multiple obstacles efficiently and stably.

This study presents a reactive collision avoidance guidance algorithm for multi-rotor UAVs to prevent close encounter with multiple moving obstacles with lateral acceleration in three-dimensional space. The proposed method constructs a bounding tube around each obstacle to find aiming point candidates for the avoidance maneuver whenever the UAV is determined to be on a collision course with the obstacle. Inappropriate candidate points are eliminated during a collision-checking step, and the selected final aiming point is provided as the command to the position controller. Meanwhile, the proposed framework performs hierarchical clustering and cluster-track association to handle multiple obstacles, because both obstacle state estimation and bounding tube formation should be conducted for each obstacle individually. A new cluster is assigned or an existing cluster is removed, depending on the circumstances. Numerical simulations considering maneuvering obstacles show that the proposed algorithm based on the bounding tube successfully avoids collision by producing aiming point candidates that are not invaded by the movement of the obstacle, whereas the previous algorithm developed in [24] based only on the instantaneous bounding box results in failure. Simulation studies also show that the algorithm appropriately handles multiple dynamic obstacles with the proposed cluster tracking method.

The contributions of this study are clarified as follows. First, the proposed guidance law improves the aiming point approach by adopting the bounding tube concept. The concept of bounding tube extends the static bounding box to incorporate the forthcoming movement of the obstacles into collision avoidance strategy. Second, this study proposes an integrated framework of the reactive collision avoidance algorithm for UAV platforms. The processing pipeline that consists of obstacle detection, clustering, filtering, guidance, and controller provides an integrated solution to avoid moving obstacles in real time. Third, the proposed reactive avoidance algorithm provides conflict resolution to multiple moving obstacles.

The remainder of this article is organized as follows. Section II briefly describes the architecture of UAV system. In Section III, a collision avoidance method based on the concept of bounding tube is proposed to deal with single dynamic obstacle. Then, the method is extended to the case of multiple obstacles in Section IV. The simulation results are presented in Section V, and the conclusion is given in Section VI.

II. VISUAL SENSING AND CONTROL SYSTEM FOR AUTONOMOUS MULTI-ROTOR UAV

This section briefly explains the hierarchical control system that enables autonomous flight of multi-rotor UAV equipped with a visual sensor. The flight control system performs several low-level, short time-scale tasks related to navigation and

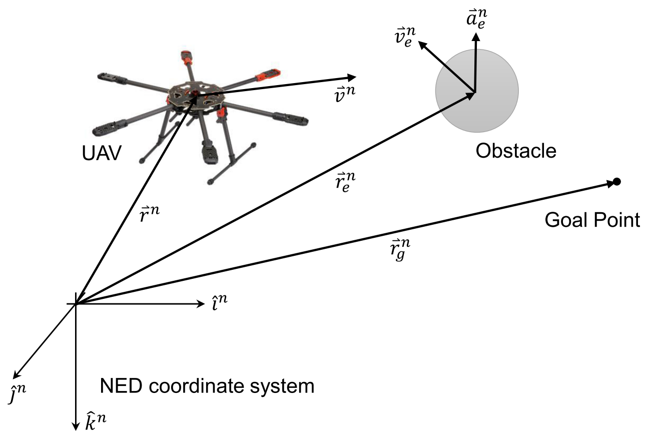
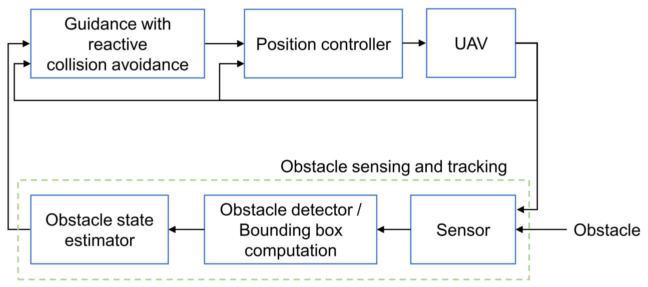


FIGURE 1. **Definition of variables and coordinate system.**



Guidance with reactive collision avoidance
Position controller
UAV
Obstacle sensing and tracking
Obstacle state estimator
Obstacle detector / Bounding box computation
Sensor
Obstacle

FIGURE 2. **Block diagram of sensing and control system.**

vehicle stabilization, which are the essential building blocks of airborne vehicle autonomy. The visual sensor provides distance to the objects in its sensing range in the form of point cloud that is further processed to obtain the estimates for the object movement and the bounding box. An overview of the UAV system is provided in this section. Details of each subsystem comprising the entire architecture are described in our previous work [24].

A. MODELING

Fig. 1 shows the quantities that are required for modeling of the dynamic system and development of the proposed collision avoidance algorithm. The North-East-Down (NED) coordinate system is used to represent vector quantities that are defined with respect to the inertial frame of reference. In Fig. 1, $\vec{r}^n$ and $\vec{v}^n$ represent the position and the velocity of the UAV, respectively. Likewise, $\vec{r}_e^n$, $\vec{v}_e^n$, and $\vec{a}_e^n$ are the estimated position, velocity, and acceleration of the obstacle, respectively. Lastly, $\vec{r}_g^n \equiv [x_g^n \ y_g^n \ z_g^n]^T$ denotes the position vector of the goal point for which the UAV is heading.

B. OVERVIEW OF UAV SYSTEM

Fig. 2 shows the UAV sensing and control pipeline represented in block diagram. The UAV system consists of two main parts: i) obstacle sensor and tracking filter, and ii) position tracking controller.

1) OBSTACLE SENSING AND TRACKING

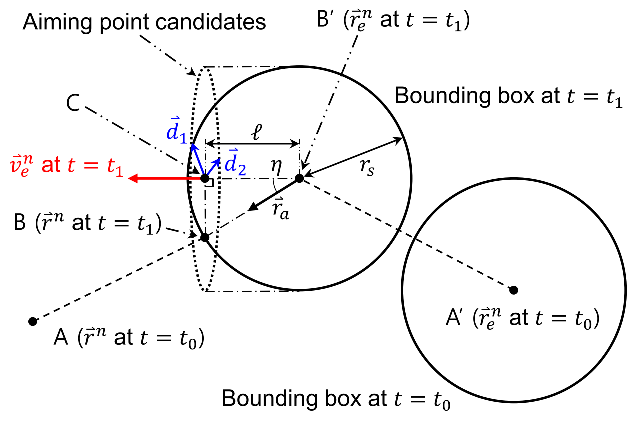
Obstacles are initially unknown to the UAV. They are detected by a sensor such as LiDAR mounted on the platform whenever the obstacle comes within the Field-Of-Regard (FOR) during the flight. In this study, the FOR is modeled as a sphere whose radius is d_a and center is $\vec{r}^n$. Once detected, the sensor measurements can be mapped into the data points in three-dimensional space that together form a cloud around the surface of the obstacle. Then, a spherical bounding box of radius r_s enclosing the arbitrarily-shaped obstacle while including a safety margin is obtained from the point cloud at each obstacle data acquisition time step as an abstract representation of the volume occupied by the obstacle. Note that the bounding box usually grows in its size as more data points are collected during a short period after detecting the obstacle. The velocity and the acceleration of the obstacle that are needed to perform collision detection are estimated by using a simple discrete-time Kalman filter with the center position of the spherical bounding box as the only measurement.

2) POSITION TRACKING CONTROL

A model for the UAV dynamics can be obtained with a rigid-body assumption. The dynamics of usual multi-rotor UAV is inherently under-actuated as it does not have an independent actuator assigned per each translational and rotational degree-of-freedom. The thrust provided by each rotor/motor assembly produces moment with respect to the center of mass of the UAV. The force produced by the propeller blades attached to the motors tilts as the UAV rotates due to the moment generated. Therefore, the UAV can gain or lose acceleration by adjusting its attitude. This allows the UAV to move to the desired position via attitude control based on time-scale separation between the translational and the rotational motion. The position loop developed in [24] produces the moment command based on a Proportional-Derivative (PD) control of the position tracking error. The guidance algorithm that will be developed in the present study to perform reactive collision avoidance produces the desired position for feeding into the position controller.

III. COLLISION AVOIDANCE STRATEGY CONSIDERING DYNAMIC OBSTACLE

The traditional approach to obstacle avoidance known as the collision cone approach takes a point of reference on the safety sphere to find the flying direction that requires minimal course correction to circumvent the threat. However, this strategy may expose vehicle to the risk of unexpected crash when the obstacle is moving. The collision cone approach makes the UAV head for the aiming point chosen on the stationary bounding box obtained at each instance without consideration to the obstacle motion. As a result, the UAV abruptly avoids the obstacle as they get close to each other, which increases the risk of the collision. To overcome this shortcoming, this study presents a collision avoidance



Aiming point candidates
B′ ($\vec{r}_e^n$ at $t = t_1$)
Bounding box at $t = t_1$
C
$\vec{d}_1$
ℓ
$\vec{v}_e^n$ at $t = t_1$
$\vec{d}_2$
η
r_s
$\vec{r}_a$
B ($\vec{r}^n$ at $t = t_1$)
A′ ($\vec{r}_e^n$ at $t = t_0$)
A ($\vec{r}^n$ at $t = t_0$)
Bounding box at $t = t_0$

FIGURE 3. Calculation of aiming point candidates.

strategy that explicitly takes account of obstacle movement predicted for a predefined period.

A. AIMING POINT CANDIDATES

Fig. 3 shows how to obtain aiming point candidates considering the predicted motion of obstacle in the situation where the UAV is currently located at point A and heads for the goal point while the obstacle is located at point A′ at the same time ($t = t_0$). From the point where the onboard sensor detects the obstacle, the obstacle state estimator employing the standard Kalman filter algorithm provides information about the position, velocity, and acceleration of the obstacle. In the meantime, the UAV calculates the future trajectory of the obstacle by propagating the filter output as follows [36], [37]:

$$\hat{x}(\Delta t_d + t_0) = \Phi_d(\Delta t_d)\hat{x}(t_0) \tag{1}$$

where $\hat{x}(t_0)$ is the estimated obstacle state at the current time t_0, and $\Phi_d(t)$ represents the discrete-time transition matrix of the process model. The future position of the obstacle can be expressed by assuming constant acceleration as

$$\vec{r}_e^n(\Delta t_d, t_0) = \frac{1}{2}\Delta t_d{}^2 \vec{a}_e^n + \Delta t_d \vec{v}_e^n + \vec{r}_e^n \tag{2}$$

where the first argument of $\vec{r}_e^n()$, Δt_d, is the prediction horizon, and the second argument is the current time. The UAV also predicts its own future trajectory in a similar manner as follows:

$$\vec{r}^n(\Delta t_d, t_0) = \Delta t_d \vec{v}^n + \vec{r}^n \tag{3}$$

Note that (3) does not incorporate vehicle acceleration, as the usual practice is to predict collision by determining whether the UAV will encounter an obstacle when moving in the current velocity direction.

The UAV determines that it is on a collision course with the obstacle, if there exists a $\Delta t_d \in [0, T_d]$ such that the predicted trajectories of the UAV and the obstacle satisfy

$$\| \vec{r}^n(\Delta t_d, t_0) - \vec{r}_e^n(\Delta t_d, t_0) \|_2 \le r_s \tag{4}$$

Let us suppose that the earliest moment after which (4) holds is denoted by $t = t_1$, and the UAV and the obstacle are at points B and B′, respectively, as shown in Fig. 3.

Calculation of the aiming point for vehicle guidance considering the obstacle movement is central to cope with the weakness of the standard collision cone approach that obtains the aiming point for the obstacle regarded as stationary at each instance. A practical approach for the collision avoidance strategy to manage the cases with dynamic obstacles through searching the aiming point from the candidates defined on a plane leading the obstacle is proposed in this study.

The aiming point should be found to compute the avoidance maneuver as the UAV detects the potential hazard of collision with the obstacle at $t = t_1$. First, a unit vector $\vec{r}_a$ in the direction viewing point A from point B′ as shown in Fig. 3 can be obtained as follows:

$$\vec{r}_a = \frac{\vec{r}^n - \vec{r}_e^n(t_1 - t_0, t_0)}{\| \vec{r}^n - \vec{r}_e^n(t_1 - t_0, t_0) \|_2} \tag{5}$$

Then, the angular separation η in Fig. 3 can be calculated as

$$\eta = \cos^{-1}\left(\frac{\vec{r}_a \cdot \vec{v}_e^n(t_1 - t_0)}{\| \vec{v}_e^n(t_1 - t_0) \|_2}\right) \tag{6}$$

where $\vec{v}_e^n(t_1 - t_0)$ is the obstacle velocity for $t = t_1$ predicted at $t = t_0$ given by

$$\vec{v}_e^n(t_1 - t_0) = (t_1 - t_0)\vec{a}_e^n + \vec{v}_e^n \tag{7}$$

The proposed guidance method generates a single waypoint at each instance for use in the overall control architecture to define the desired flying direction. The proposed method takes an approach separating the process of aiming point generation into two distinguishable steps: i) generation of a cloud of candidates, and ii) selection of the best one ensuring safety of the vehicle.

The group of aiming point candidates are generated by deterministic sampling from a virtual circle on the tangent plane to the predicted bounding box that is also perpendicular to the predicted obstacle velocity $\vec{v}_e^n(t_1 - t_0)$. Let us define point C as the center of aiming point candidates. The distance between point B′ and point C, which is denoted by ℓ in Fig. 3, can be computed as follows:

$$\ell = r_s \cos \eta \tag{8}$$

The position vector of point C represented in the NED coordinate system can be expressed as follows:

$$\vec{r}_c^n = \vec{r}_e^n(t_1 - t_0, t_0) + \ell \frac{\vec{v}_e^n(t_1 - t_0)}{\| \vec{v}_e^n(t_1 - t_0) \|_2} \tag{9}$$

Now, suppose that the vectors $\vec{d}_1$ and $\vec{d}_2$ in Fig. 3 along with $\vec{v}_e^n(t_1 - t_0)$ form an orthogonal basis of three-dimensional Euclidean space. These vectors can be obtained by computing the bases of the null space of $\vec{v}_e^n(t_1 - t_0)^T$. Then, their magnitudes are adjusted to r_s. The aiming point candidates represented in the NED coordinate system, $\vec{r}_p^n$, can be expressed as

$$\vec{r}_p^n = \vec{d}_1 \cos\varphi + \vec{d}_2 \sin\varphi + \vec{r}_c^n \tag{10}$$

for φ values in a uniform grid spanning over $[0, 2\pi)$. The density of the φ grid determines the number of aiming point candidates.

This process results in aiming point candidates on a circle whose center, radius, and normal vector are $\vec{r}_c^n$ (point C), r_s, and $\vec{v}_e^n(t_1 - t_0)$, respectively. It is worth noting that the aiming point candidates are not invaded by the movement of the obstacle predicted for the period of $[t_0, t_1]$.

B. BOUNDING TUBE

The collision avoidance strategy needs a certain degree of inherent conservatism against uncertain factors that may be present in reality but are neglected in the simplified prediction models. Without appropriate management of unexpected risks, the mismatch between the predicted trajectories and the actual trajectories due to model uncertainties can bring about catastrophic harm, i.e., failure in collision avoidance.

Once the UAV detects the possibility of collision with the obstacle at the future time $t = t_1$, the UAV changes the velocity direction to the aiming point selected among the candidates given as (10). The avoidance maneuver induces curvature of the trajectory by the UAV, resulting in trajectory prediction errors due to the discrepancy between the constant velocity model assumed for prediction in (3) and the actual trajectory. As a consequence, the UAV arrives at the aiming point later or earlier than the estimated time $t = t_1$. Also, the UAV describing a curved trajectory may approach closer to the obstacle while performing collision avoidance maneuver than the straight path repetitively predicted with the simple model, leading to the violation of the bounding box.

To prevent complete failure and to enhance safety during execution of avoidance maneuver, it is desirable to employ a robustifying modification to the process of generating the aiming point candidates, even though a more conservative planning can be achieved only at the cost of sacrifice in efficiency in terms of less steering. The overall direction of modification is to place more candidate points around the nominal set of aiming point candidates. One possible way of extending the set of aiming point candidates is to generate points on the external surface of a bounding tube as shown in Fig. 4. Here, the bounding tube is defined as the volume swept by the obstacle moving in the time interval of $[t_1 - \Delta t_b, t_1 + \Delta t_b]$ where Δt_b is a design parameter that determines the length of the tube.

To obtain an extended set of candidate points, let us consider a sweeping parameter Δt_p that attains value in the range of $[-\Delta t_b, \Delta t_b]$. As shown in Fig. 4, the center of the bounding box at $t = t_1 + \Delta t_p$ can be computed as $\vec{r}_e^n(\Delta t_p, t_1)$ according to (2). The velocity of the obstacle at $t = t_1 + \Delta t_p$, $\vec{v}_e^n(\Delta t_p)$, can be predicted by using (7). Then, the aiming point candidates corresponding to the bounding box at $t = t_1 + \Delta t_p$ (dotted circle in Fig. 4) can be obtained by following the relation of (10) with $\vec{d}_1$ and $\vec{d}_2$ defined to be perpendicular to $\vec{v}_e^n(\Delta t_p)$ and $\vec{r}_c^n$ replaced by $\vec{r}_e^n(\Delta t_p, t_1)$. Finally, all the

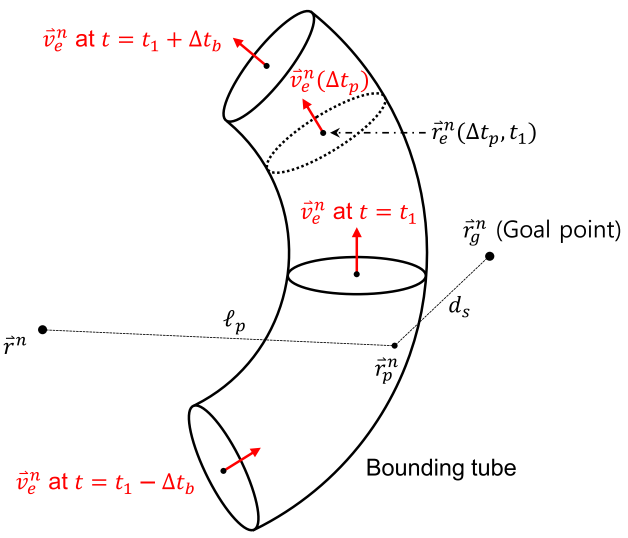

$\vec{v}_e^n$ at $t = t_1 + \Delta t_b$
$\vec{v}_e^n(\Delta t_p)$
$\vec{r}_e^n(\Delta t_p, t_1)$
$\vec{v}_e^n$ at $t = t_1$
$\vec{r}_g^n$ (Goal point)
d_s
ℓ_p
$\vec{r}^n$
$\vec{r}_p^n$
Bounding tube
$\vec{v}_e^n$ at $t = t_1 - \Delta t_b$

FIGURE 4. Bounding tube: Generation of aiming point candidates for an interval $[t_1 - \Delta t_b, t_1 + \Delta t_b]$.

candidate points can be obtained along the bounding tube by varying Δt_p from $-\Delta t_b$ to Δt_b. Note that Δt_p is taken from a uniform grid over $(0, \Delta t_b]$, and the lower end of the time interval for which the bounding tube is obtained cannot be less than the initial time t_0, i.e., the interval defining the bounding tube is actually $[\max(t_1 - \Delta t_b, t_0), t_1 + \Delta t_b]$.

C. SELECTION OF AIMING POINT

An appropriate aiming point should be selected among the candidates generated previously. In this study, the distance between a candidate point and the goal point, d_s in Fig. 4, is selected as the measure for evaluation of the candidate points. If $\vec{r}_p^n$ is one of the candidate points, d_s can be calculated as follows:

$$d_s = \| \vec{r}_p^n - \vec{r}_g^n \|_2 \tag{11}$$

Then, the entire candidate points can be sorted by the value of d_s for each $\vec{r}_p^n$ in ascending order.

Let us denote $\vec{r}_{p,min}^n$ as the candidate point with minimum d_s, and t_{min} as the time instance that corresponds to $\vec{r}_{p,min}^n$ on the bounding tube. If $\vec{r}_{p,min}^n$ is chosen as the final aiming point, it means that the UAV performs a collision avoidance maneuver with the minimum possible deviation from the current line-of-sight towards the goal point. However, it should be guaranteed that the UAV does not collide with the obstacle while heading for $\vec{r}_{p,min}^n$.

An additional collision-checking step is necessary to ensure minimum clearance of r_s between the vehicle trajectory predicted with the velocity in the direction towards a chosen aiming point, instead of the current velocity, and the predicted obstacle trajectory. The distance between $\vec{r}^n$ and $\vec{r}_{p,min}^n$, ℓ_p in Fig. 4, can be computed as follows:

$$\ell_p = \| \vec{r}^n - \vec{r}_{p,min}^n \|_2 \tag{12}$$

TABLE 1. Pseudocode of aiming point calculation utilizing bounding tube.

```
design parameter: $t_d$; $T_d$; $t_p$; $\Delta t_b$;
input: $\vec{r}^n$; $\vec{v}^n$; $\vec{r}_e^n$; $\vec{v}_e^n$; $\vec{a}_e^n$;
output: final aiming point;
begin
  for $\Delta t_d = 0 : t_d : T_d$
    compute $\vec{r}_e^n(\Delta t_d, t_0)$ and $\vec{r}^n(\Delta t_d, t_0)$ using (2) and (3);
    if (4) is true
      break;
    end
  end
  bound_tube = [ ];
  for $\Delta t_p = -\Delta t_b : t_p : \Delta t_b$
    if $\Delta t_d + \Delta t_p \geq 0$
      compute $\vec{r}_e^n(\Delta t_p, t_1)$ and $\vec{v}_e^n(\Delta t_p)$ using (2) and (7);
      obtain $\vec{r}_p^n$ using (10);
      calculate $d_s$ using (11);
      add $\vec{r}_p^n$ and $d_s$ to bound_tube;
    end
  end
  sort bound_tube by $d_s$ in ascending order;
  while
    $\vec{r}_{p,min}^n \leftarrow \vec{r}_p^n$ with minimum $d_s$;
    if collision occurs for $[t_0, t_{go}]$
      remove current $\vec{r}_{p,min}^n$ from bound_tube;
    else
      final aiming point $\leftarrow \vec{r}_{p,min}^n$;
      break;
    end
  end
end
```

If v_{avg} is the average speed of the UAV, the time-to-go of the UAV to $\vec{r}_{p,min}^n$ can be calculated as follows:

$$t_{go} = \frac{\ell_p}{v_{avg}} \tag{13}$$

Then, t_{go} is the length of time-window considered for the collision-checking. The distance between the predicted obstacle position, $\vec{r}_e^n(\Delta t_d, t_0)$ of (2), and the predicted UAV position, $\vec{r}^n(\Delta t_d, t_0)$ of (3), is computed for the time interval of $[t_0, t_{go}]$. Note that the UAV trajectory here should be the one predicted with the velocity of

$$\vec{v}_p^n = v_{avg} \frac{\vec{r}_{p,min}^n - \vec{r}^n}{\| \vec{r}_{p,min}^n - \vec{r}^n \|_2} \tag{14}$$

instead of $\vec{v}^n$ in (3). If the distance decreases below the radius of bounding sphere, r_s, at any point on the predicted trajectory, the aiming point $\vec{r}_{p,min}^n$ tested at the current iteration is discarded from the set of candidates as it cannot guarantee safety of the vehicle. Then, the point of minimum d_s among the remaining candidate points is selected as a new aiming point $\vec{r}_{p,min}^n$, and the collision-checking step is performed again in the same fashion. The pruning process is repeated over the sorted list until finding the final aiming point $\vec{r}_{p,min}^n$ such that the distance between the predicted trajectories of UAV and obstacle is greater than r_s for the entire time interval of $[t_0, t_{go}]$.

Table 1 summarizes the proposed algorithm in the form of a pseudocode to facilitate implementation.

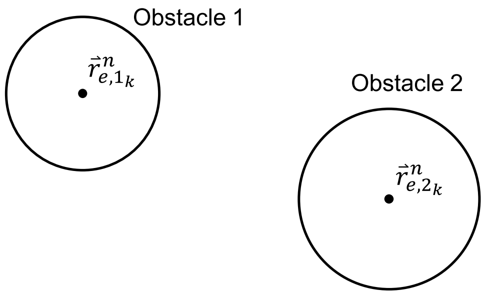


FIGURE 5. Two obstacles detected at the k-th step.

IV. EXTENSION TO MULTIPLE OBSTACLES

The methodology of generation of aiming point candidates by using the bounding tube followed by pruning of candidates can be applied in a similar fashion to the case where multiple obstacles exist in the environment. However, straightforward extension to the multi-obstacle case is possible only if the UAV is capable of multi-target tracking. That is, the UAV should be able to identify each obstacle from the clouds of acquired obstacle data points and associate different clusters with respective correct track in order to correctly estimate states of the obstacles and to construct bounding tubes. This section presents a cluster-to-track assignment method for the extension of the proposed algorithm to multiple obstacles.

In the clustering step, sensed data points are grouped into clusters at each time step. This study adopts distance connectivity based hierarchical clustering [27] for cluster-to-track assignment of multiple obstacles. The Euclidean distance between the closest pair is considered as a cutoff threshold, d_{ct}. The UAV perceives each cluster as an individual obstacle and the clusters move in space according to movement of the corresponding obstacles.

In the next step, track assignment is conducted to handle multiple clusters. Each measurement cluster should be associated with the correct one from among the evolving tracks that exist at each instant. This track assignment step is necessary because wrong pairing between a measurement cluster and a track would result in large and abrupt Kalman filter transients, which may lead to catastrophic consequences such as collision with obstacles or loss of vehicle stability.

Now let us explain how to determine the association between newly observed clusters and obstacle tracks with an illustrative example. Fig. 5 shows two obstacles detected by the sensor mounted on the UAV at the k-th step. It is assumed that the distance between the obstacles is greater than d_{ct}. Thus, two clusters are generated by the hierarchical clustering method. It is also assumed that obstacle 1 in Fig. 5 was detected a while ago, and its state estimate has converged to a steady-state value. Obstacle 2 is assumed to be detected at the k-th step for the first time, and its state estimator is just initiated. Also, $\vec{r}_{e,1_k}^n$ and $\vec{r}_{e,2_k}^n$ are the position estimates of obstacle 1 and obstacle 2 at the k-th step, respectively, which can be directly obtained from the sensor.

Fig. 6 shows the obstacles detected at the $(k+1)$-th step, i.e., the two clusters are formed. Each of the newly obtained

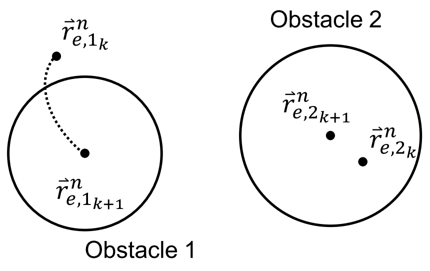


FIGURE 6. Two obstacles detected at the $(k+1)$-th step.

clusters should be associated with the right obstacle track that has been maintained until the previous step for smooth state estimation. After the transient response in the state estimate of a cluster has died out, it can reliably be utilized for tracking of the cluster. The trajectory of obstacle 1 between the k-th and the $(k+1)$-th steps can be estimated using (2), which is depicted as dotted line in Fig. 6. If the data points comprising one of the clusters at the current step and the points sampled along the trajectory estimated from the track of obstacle 1 between neighboring time steps belong to the same cluster after performing the hierarchical clustering once again, then it is reasonable to say that the cluster at the $(k+1)$-th step belongs to the track of obstacle 1 maintained up to the k-th step. Note that multiple clusters are generated as a result of the hierarchical clustering performed over the trajectory estimated with the information of the track of obstacle 1 and the cluster around obstacle 2 in Fig. 6, which is obviously a mismatch.

The state estimate of a cluster cannot be used reliably if the transient behavior in the state estimate of a cluster has still not died out yet. In this case, minimum distance between the position estimates is utilized to determine the track that corresponds to the cluster. Thus, the association of obstacle 2 at the $(k+1)$-th step in Fig. 6 is conducted as follows:

$$\underset{i}{\operatorname{argmin}} \left\{ \| \vec{r}^{\,n}_{e,2_{k+1}} - \vec{r}^{\,n}_{e,i_k} \|_2 \mid i \in \{1,2\} \right\} \tag{15}$$

Let us define n_k and n_{k+1} as the numbers of clusters generated at the k-th and the $(k+1)$-th steps, respectively. Depending on how the number of clusters changes between two consecutive steps, there are six possible cases as listed below:

- $n_{k+1} > n_k$ and $n_k = 0$: CL_1 in Table 2
- $n_{k+1} > n_k$ and $n_k \neq 0$: CL_2 in Table 2
- $n_{k+1} < n_k$ and $n_{k+1} = 0$: CL_3 in Table 2
- $n_{k+1} < n_k$ and $n_{k+1} \neq 0$: CL_4 in Table 2
- $n_{k+1} = n_k \neq 0$: CL_5 in Table 2
- $n_{k+1} = n_k = 0$: CL_6 in Table 2

The clusters should be managed appropriately according to CL_k at each time step, where $k \in \{1, \cdots, 6\}$. The proposed cluster-track association as well as the overall summary of the proposed collision avoidance algorithm is described in Table 2.

TABLE 2. Pseudocode of collision avoidance against multiple obstacles.

```
design parameter: $d_{ct}$;
input: obstacle data points at current step;
output: guidance command to UAV;
begin
  $Avoid \leftarrow 0$;
  obtain cluster using hierarchical clustering at current step;
  determine $CL_k$ by comparison with previous step;
  if $CL_3$ or $CL_6$
    $CA \leftarrow 0$;
    remove every cluster;
  else
    $CA \leftarrow 1$;
  end
  if $CA$ is 1
    obtain $\vec{r}^{\,n}_e$ and $r_s$ for each cluster;
    if $CL_1$
      assign new cluster;
    end
    if $CL_2$ or $CL_4$ or $CL_5$
      track clusters by proposed cluster-to-track assignment;
      if $CL_2$
        assign the remainder as new cluster;
      end
      if $CL_4$
        remove the lost cluster;
      end
    end
    for each cluster
      update Kalman filter using $\vec{r}^{\,n}_e$;
      update $r_s$;
      examine collision using pseudocode in Table 1;
      if (4) is true
        $Avoid \leftarrow 1$;
        construct bounding tube using pseudocode in Table 1;
      end
    end
    gather all the aiming point candidates;
    compute final aiming point using pseudocode in Table 1;
  end
  if $Avoid$ is 1
    guide UAV to final aiming point;
  else
    guide UAV to goal point;
  end
end
```

V. NUMERICAL SIMULATION AND DISCUSSION

Numerical simulations are performed to demonstrate the performance of the proposed reactive collision avoidance algorithm. The time step of numerical integration and control update is 0.005 s, and the sampling time of obstacle data acquisition from the sensor is 0.05 s. Also, the sensor range, d_a, is set to 20 m. To avoid excessive maneuver, the roll and pitch angle commands are limited not to exceed $\pm 20°$. The UAV utilizes the obstacle state estimate 1 s after the first detection of an obstacle in order to allow a period for the Kalman filter estimate to converge. Design parameters of the proposed collision avoidance algorithm are set as follows:

$$\begin{aligned} t_d &= 0.05 \text{ s} \quad T_d = 20 \text{ s} \\ t_p &= 0.025 \text{ s} \quad \Delta t_b = 5 \text{ s} \\ d_{ct} &= 0.4 \text{ m} \end{aligned} \tag{16}$$

where t_d and t_p are time step parameters introduced in Table 1. The rest of the parameters of UAV, Kalman filter, and

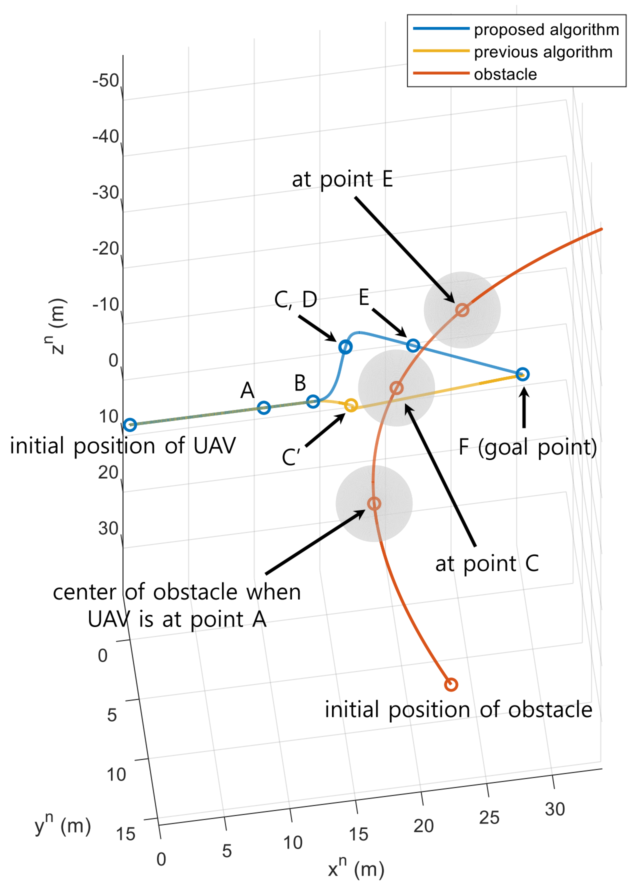


FIGURE 7. Trajectories of the UAV and the obstacle (Simulation I).

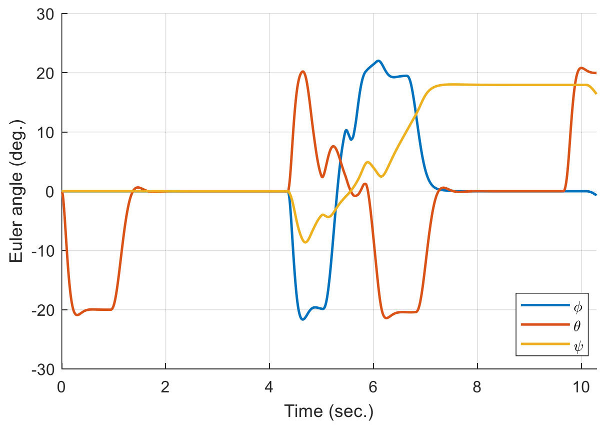


FIGURE 8. Roll, pitch, and yaw angles (Simulation I).

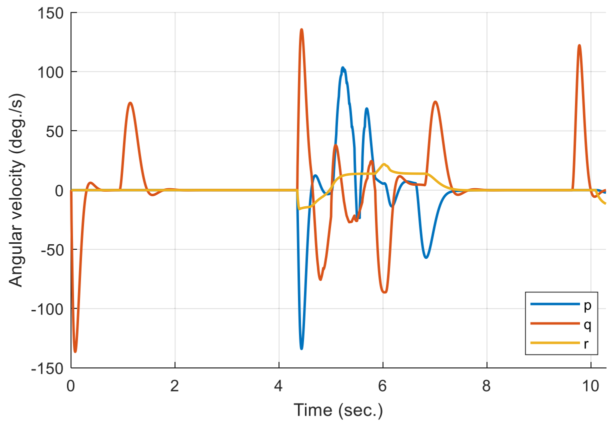


FIGURE 9. Body rate (Simulation I).

collision avoidance algorithm used in simulations are identical to those used in [24].

A. SIMULATION I: SINGLE OBSTACLE

In Simulation I, the proposed collision avoidance algorithm is compared to the previous algorithm [24] in the presence of a single obstacle. The initial position of the UAV and the goal point are set to $[0\ 0\ 0]^T$ and $[30\ 0\ 0]^T$ in the NED coordinate system, respectively. A spherical obstacle with a radius of 2.9 m is initially located at $[23.5\ 7.1\ 38.4]^T$. Its initial velocity and acceleration vectors are set to $[-2.6\ \ -3.8\ \ -3.2]^T$ and $[0.7\ 0.9\ \ -1.2]^T$, respectively. There exists a lateral acceleration since the angle between the velocity vector and the acceleration vector of the obstacle is 98.7°. Without a collision avoidance maneuver, the UAV and the obstacle are on a collision course.

Fig. 7 shows the trajectories of the UAV with the proposed algorithm and the previous algorithm, and the obstacle in the NED coordinate system. Figs. 8-12 show the time responses of the attitude, angular velocity, velocity, and speed of the UAV, and the minimum distance between the UAV and the obstacle. Also, Fig. 13 shows the time responses of the state estimate of the Kalman filter, where $[x^n_{e,k}\ y^n_{e,k}\ z^n_{e,k}]^T$, $[\dot{x}^n_{e,k}\ \dot{y}^n_{e,k}\ \dot{z}^n_{e,k}]^T$, and $[\ddot{x}^n_{e,k}\ \ddot{y}^n_{e,k}\ \ddot{z}^n_{e,k}]^T$ are the components of the position, velocity and acceleration estimates of the obstacle, respectively, in the NED coordinate system.

At first, the UAV heads for the goal point because the obstacle is not sensed until point A ($t = 3.35$ s) in Fig. 7. At point A, the obstacle enters the sensor range as shown in Fig. 12, and the Kalman filter initiates estimation of the obstacle state. After the transient phase, the obstacle state estimates converge to the true states at point B ($t = 4.35$ s) as shown in Fig. 13. From this point, the UAV uses the obstacle information provided by the Kalman filter to compute the aiming point for collision avoidance. Note that the trajectories of the UAV resulting from the proposed and previous algorithms remain identical until point B, as shown in Fig. 7.

At point B, the future positions of the obstacle and the UAV are calculated by using (2) and (3), respectively. The UAV detects that it is on a collision course, as (4) is true for one of the predicted bounding boxes. Then, the bounding tube is formed as shown in Fig. 14 to obtain the aiming point for avoidance maneuver. The candidate points on the bounding tube are sorted, and some of them are removed from the set by observing an additional possible collision with the proposed pruning process. At last, the final aiming point, $[17.7\ \ -3.8\ \ -2.4]^T$, is chosen as shown in Fig. 14.

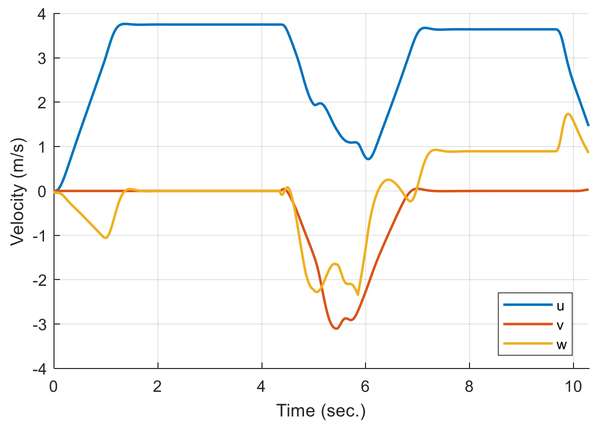


FIGURE 10. Velocity (body-fixed coordinate system, Simulation I).

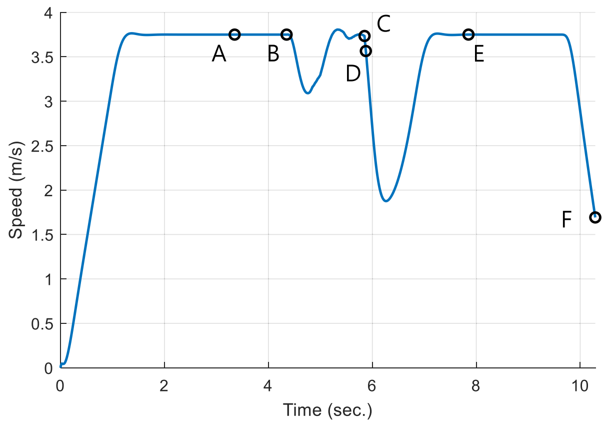


FIGURE 11. Speed (Simulation I).

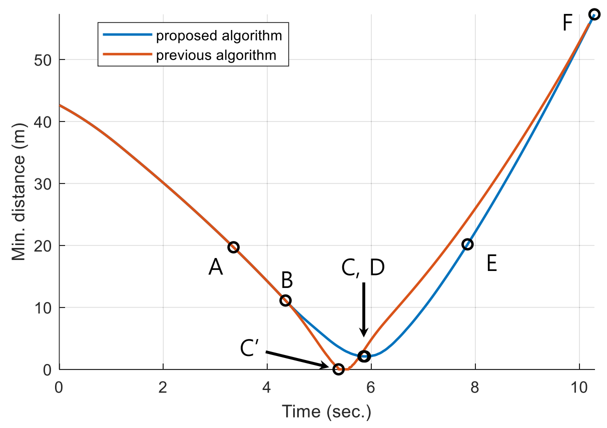


FIGURE 12. Minimum distance between the UAV and the obstacle (Simulation I).

TABLE 3. Parameters of the obstacles (Simulation II).

Shape	sphere	sphere	sphere
Radius (m)	1.85	1.03	1.95
Initial position (m)	$\begin{bmatrix} 40.7 \\ 2.6 \\ -13.8 \end{bmatrix}$	$\begin{bmatrix} 29.3 \\ 9.7 \\ -0.6 \end{bmatrix}$	$\begin{bmatrix} 36.2 \\ -1.7 \\ -4.1 \end{bmatrix}$
Initial velocity (m/s)	$\begin{bmatrix} -0.2 \\ 0.7 \\ 0.5 \end{bmatrix}$	$\begin{bmatrix} 0.2 \\ -0.2 \\ 0.7 \end{bmatrix}$	$\begin{bmatrix} -0.3 \\ -0.7 \\ -0.4 \end{bmatrix}$
Initial acceleration (m/s^2)	$\begin{bmatrix} -0.9 \\ -0.4 \\ 0.6 \end{bmatrix}$	$\begin{bmatrix} -0.8 \\ -0.6 \\ -0.2 \end{bmatrix}$	$\begin{bmatrix} -0.9 \\ 0.4 \\ 0.4 \end{bmatrix}$

Meanwhile, the previous algorithm finds the candidate points using a conventional collision cone approach, and removes some of them by considering only a short-term future movement of the obstacle, as shown in Fig. 15. Then, the point closest in distance to the velocity vector of the UAV, $[15.5 \ \ -3.0 \ 3.6]^T$, is chosen as the the final aiming point as shown in Fig. 15. Though the previous algorithm selects the aiming point on the opposite side of the obstacle's predicted trajectory, it does not consider whether the UAV is on a collision course while heading for the aiming point. As a result, the UAV collides with the obstacle at point C′ ($t = 5.37$ s) as shown in Fig. 7 and Fig. 12.

On the contrary, the UAV with the proposed algorithm successfully avoids the obstacle with the minimum separation of 2.08 m from the obstacle at point C ($t = 5.88$ s), as shown in Fig. 7 and Fig. 12. Note that a safety margin of the bounding box is set to 2 m. At point D ($t = 5.9$ s), the UAV determines that the obstacle is no longer a threat and heads again for the goal point. Then, at point E ($t = 7.85$ s), the obstacle moves outside the sensing range as shown in Fig. 12. Note that the performance of the Kalman filter begins to deteriorate after point E as shown in Fig. 13. Finally, the UAV reaches the goal point at $t = 10.29$ s as shown in Fig. 7.

B. SIMULATION II: MULTIPLE OBSTACLES

In Simulation II, the effectiveness of the proposed algorithm in case of multiple moving obstacles is demonstrated. The initial position of the UAV and the goal point are set to $[0\ 0\ 0]^T$ and $[30\ 0\ 0]^T$ in the NED coordinate system, respectively. The parameters of the obstacles are listed in Table 3. The UAV is about to collide with all three obstacles if it flies directly to the goal point. In this study, collision between obstacles is not considered.

Fig. 16 shows the trajectories of the UAV and the obstacles in the NED coordinate system. Figs. 17-22 show the time responses of the attitude, angular velocity, velocity, speed, and cluster management of the UAV, and the minimum distance between the UAV and the obstacles.

At first, the UAV heads for the goal point as shown in Fig. 16, and no cluster is obtained (CL_6) as shown in Fig. 21. Then, the obstacle is detected by the UAV at $t = 2.60$ s as shown in Fig. 22. At $t = 2.60$ s, CL_1 is obtained and the corresponding cluster is assigned as obstacle 1. The Kalman filter begins to estimate the state of obstacle 1. In Fig. 21, the number of cluster does not increase at $t = 2.60$ s because it represents the cluster which is on a collision course. However, collision cannot be detected because the state estimate of obstacle 1 is not available yet.

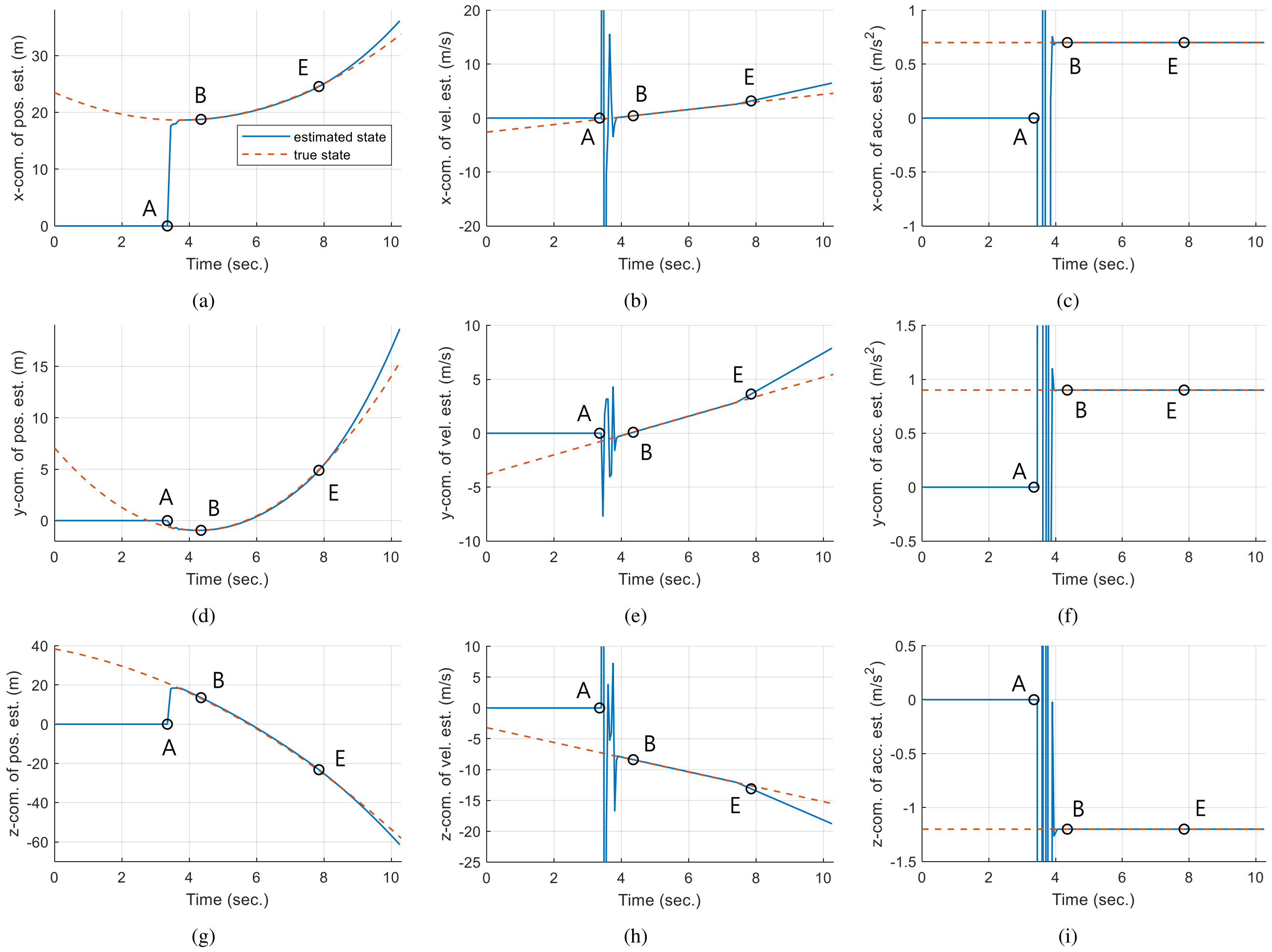


FIGURE 13. Time histories of the state estimate (Simulation I): (a) $x^n_{e,k}$, (b) $\dot{x}^n_{e,k}$, (c) $\ddot{x}^n_{e,k}$, (d) $y^n_{e,k}$, (e) $\dot{y}^n_{e,k}$, (f) $\ddot{y}^n_{e,k}$, (g) $z^n_{e,k}$, (h) $\dot{z}^n_{e,k}$, and (i) $\ddot{z}^n_{e,k}$.

After $t = 2.60$ s, CL_5 is obtained and obstacle 1 is tracked by the proposed track assignment algorithm. At $t = 3.15$, another obstacle is detected by the UAV, and as a result, two clusters are formed by the hierarchical clustering. At this moment, CL_2 is obtained, and obstacle 1 is tracked first. Then, the remaining cluster is assigned as obstacle 2 for which a separate Kalman filter starts running. Now, the UAV has to track two obstacles identified by the cluster-to-track assignment strategy (CL_5). The state estimate of obstacle 1 converges at point A ($t = 3.60$ s), and the proposed collision avoidance algorithm starts to incorporate its information. By calculating the future positions of the UAV and obstacle 1, the UAV determines that obstacle 1 is on a collision course as shown in Fig. 21. Then, the bounding tube is formed as shown in Fig. 23 to obtain the aiming point for collision avoidance maneuver. Though there exist two clusters, only one bounding tube is formed in Fig. 23 because the state estimate of obstacle 2 is in its transient phase. The candidate points on one bounding tube are sorted and post-processed by the pruning process. The final aiming point for avoidance maneuver is chosen from obstacle 1 as shown in Fig. 21 and Fig. 23. Note from Fig. 16 that the UAV starts to change its flight path due to the proposed collision avoidance algorithm.

At $t = 3.95$ s, the last obstacle is detected and CL_2 is obtained. Two clusters are associated to the tracks of obstacle 1 and obstacle 2, and the remaining cluster is assigned as obstacle 3. The state estimate of obstacle 2 becomes available to use after point B ($t = 4.15$ s), and the UAV detects potential collision with two obstacles as shown in Fig. 21. Now, two bounding tubes are formed and all the candidate points from the tubes are sorted by d_s. After performing the pruning process, the final aiming point is selected from the bounding tube around obstacle 1 as shown in Fig. 21. That is, though the bounding tube from obstacle 2 is introduced, the UAV sticks to avoid obstacle 1.

At point C ($t = 4.95$ s), the state estimate of obstacle 3 becomes available and three bounding tubes are formed as shown in Fig. 24. However, the final aiming point is still obtained from the bounding tube enclosing obstacle 1 as shown in Fig. 21 and Fig. 23. At point D ($t = 5.54$ s), the minimum distance between the UAV and obstacle 1 is 2.12 m, as shown in Fig. 22. Note from Fig. 16 that the

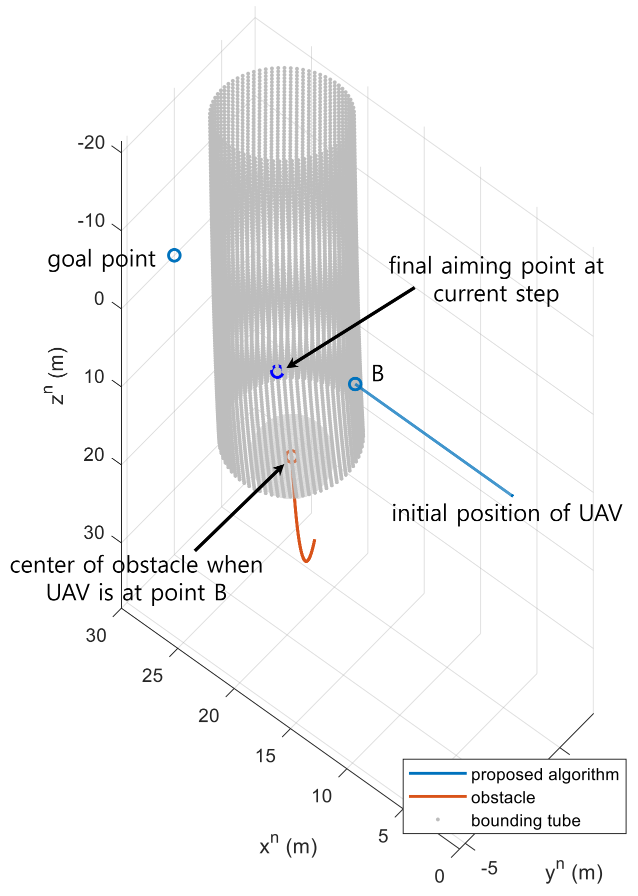


FIGURE 14. Bounding tube construction at point B ($t = 4.35$ s, Simulation I).

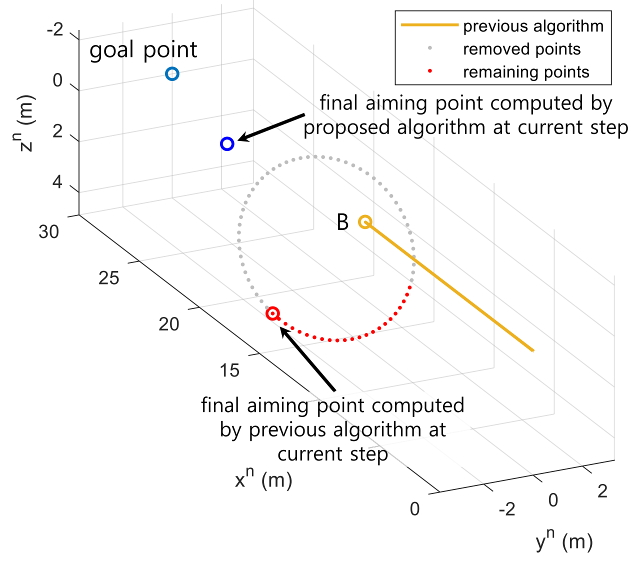


FIGURE 15. Aiming point candidates by previous algorithm at point B ($t = 4.35$ s, Simulation I).

trajectory of UAV from point A to point D maintains the same direction as the UAV continuously maneuvers to avoid obstacle 1.

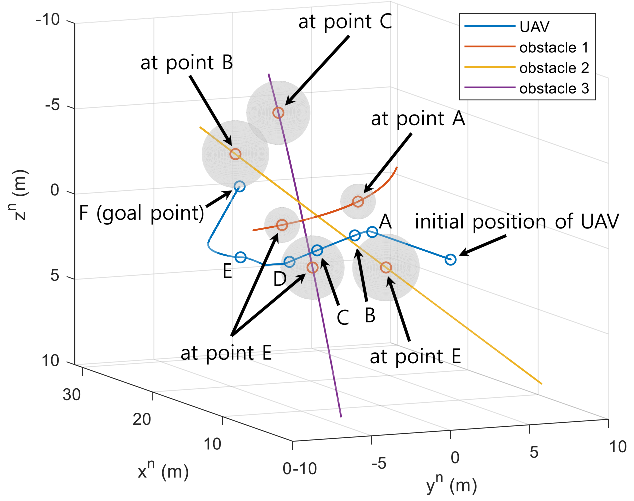


FIGURE 16. Trajectories of the UAV and the obstacles (Simulation II).

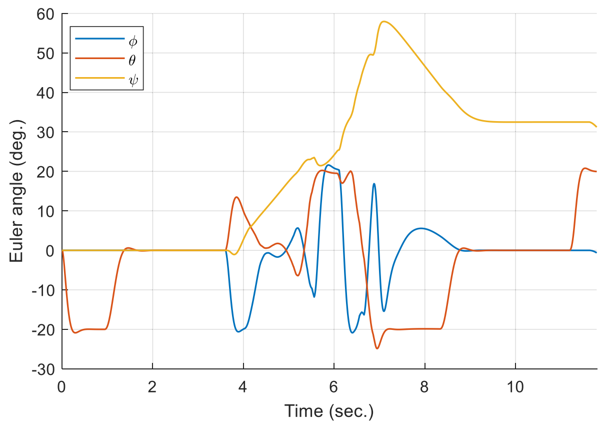


FIGURE 17. Roll, pitch, and yaw angles (Simulation II).

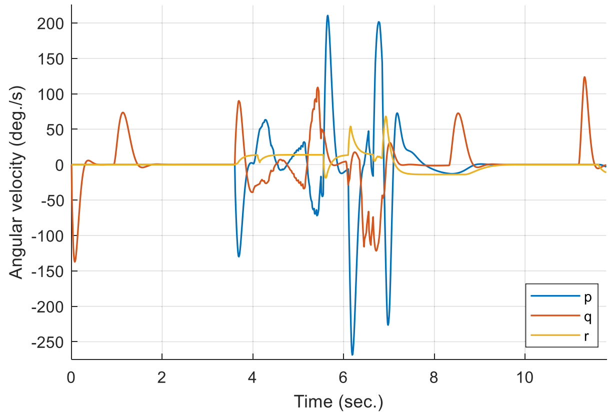


FIGURE 18. Body rate (Simulation II).

At $t = 5.75$ s, obstacle 1 is no longer a threat and the number of clusters on a collision course decreases as shown in Fig. 21. The aiming point is selected among the points on the bounding tubes of obstacle 2 and obstacle 3. And the UAV is guided to the point on the bounding tube of obstacle 2 as shown in Fig. 21. At $t = 6.1$ s, the UAV determines that

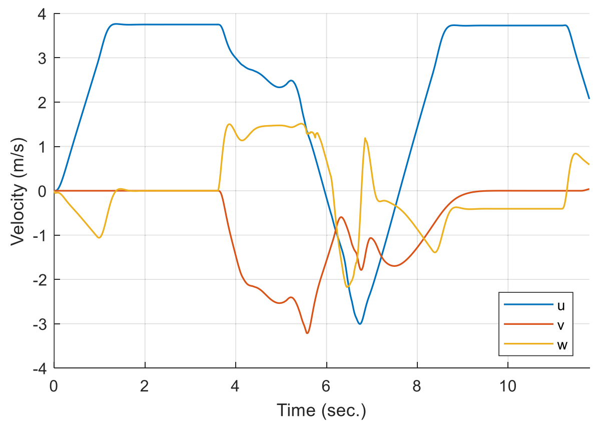


FIGURE 19. Velocity (body-fixed coordinate system, Simulation II).

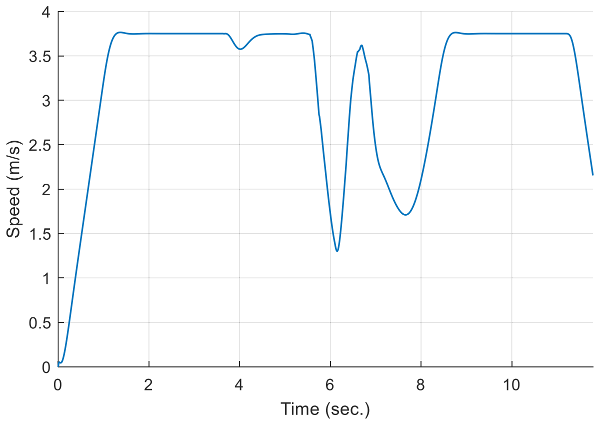


FIGURE 20. Speed (Simulation II).

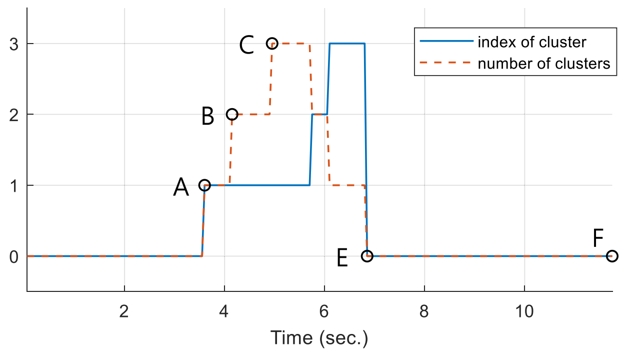


FIGURE 21. Cluster management (Simulation II).

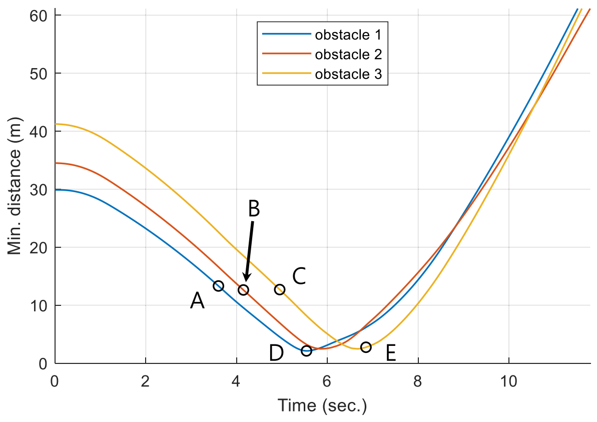


FIGURE 22. Minimum distance between the UAV and the obstacles (Simulation II).

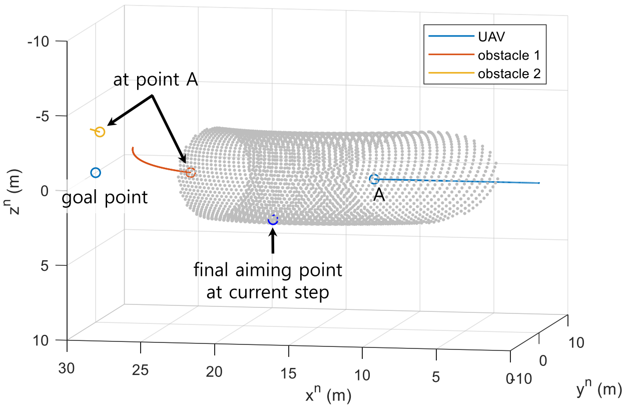


FIGURE 23. Bounding tube construction at point A (t = 3.60 s, Simulation II).

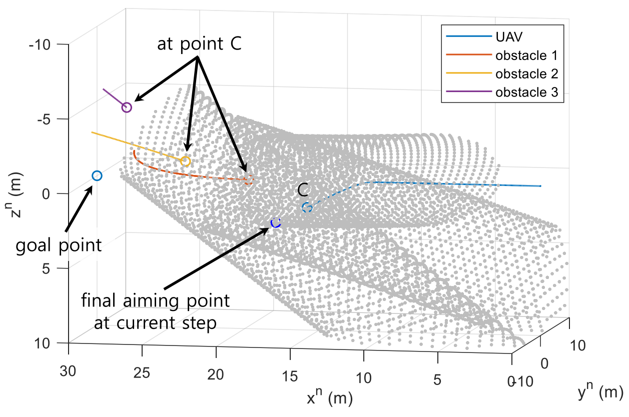


FIGURE 24. Bounding tube construction at point C (t = 4.95 s, Simulation II).

obstacle 2 is also not on a collision course as shown in Fig. 21. Therefore, the aiming point is chosen from the bounding tube of obstacle 3.

At point E ($t = 6.85$ s), the UAV is not on a collision course with all three obstacles, and it is guided to the goal point as shown in Fig. 16 and Fig. 21. CL_4 is obtained at $t = 8.50$ s and $t = 8.55$ s as obstacle 2 and obstacle 1 move outside the sensing range, respectively, as shown in Fig. 22. The corresponding clusters are removed. CL_3 is obtained at $t = 8.85$ s as obstacle 3 gets out of the sensing range, and the last cluster is removed. Finally, the UAV reaches the goal point at $t = 11.79$ s as shown in Fig. 16. Note that the minimum distance between the UAV and obstacle 2 is 2.47 m and the minimum distance between the UAV and obstacle 3 is 2.48 m, as shown in Fig. 22.

VI. CONCLUSION

In this study, a reactive collision avoidance algorithm for multi-rotor UAVs was proposed to avoid multiple dynamic obstacles. Provided that the UAV acquired obstacle data points with a visual sensor of a limited sensing range, the proposed framework performed collision detection and avoidance maneuver planning by considering the kinematic information of the obstacles estimated by processing the sensor measurements. Specifically, the proposed method utilized the aiming point candidates generated on the bounding tube which was introduced in this study to take explicit account of the movement of obstacles at the stage of avoidance maneuver planning. The point with collision-free path causing minimal deviation from the original course was chosen as the final aiming point through the pruning process removing inappropriate candidate points. A hierarchical clustering method was used to deal with multiple obstacles by treating each cluster as a single obstacle. The clusters were associated with the tracks of obstacles for consistency of state estimation, and bounding tubes were constructed for the ones that could potentially collide with the UAV. It was demonstrated through numerical simulations that the proposed collision avoidance framework showed improved capability over the previous algorithm and could successfully be deployed to a multi-rotor UAV flying in the environment cluttered by multiple moving obstacles.

REFERENCES

[1] X. Prats, L. Delgado, J. Ramírez, P. Royo, and E. Pastor, "Requirements, issues, and challenges for sense and avoid in unmanned aircraft systems," *J. Aircr.*, vol. 49, no. 3, pp. 677–687, May 2012.

[2] X. Yu and Y. Zhang, "Sense and avoid technologies with applications to unmanned aircraft systems: Review and prospects," *Prog. Aerosp. Sci.*, vol. 74, pp. 152–166, Apr. 2015.

[3] G. Fasano, D. Accado, A. Moccia, and D. Moroney, "Sense and avoid for unmanned aircraft systems," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 31, no. 11, pp. 82–110, Nov. 2016, doi: 10.1109/MAES.2016.160116.

[4] J. N. Yasin, S. A. S. Mohamed, M.-H. Haghbayan, J. Heikkonen, H. Tenhunen, and J. Plosila, "Unmanned aerial vehicles (UAVs): Collision avoidance systems and approaches," *IEEE Access*, vol. 8, pp. 105139–105155, 2020.

[5] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.*, vol. 30, no. 7, pp. 846–894, Jun. 2011.

[6] Y. Lin and S. Saripalli, "Sampling-based path planning for UAV collision avoidance," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 11, pp. 3179–3192, Nov. 2017.

[7] Y. Choi, H. Jimenez, and D. N. Mavris, "Two-layer obstacle collision avoidance with machine learning for more energy-efficient unmanned aircraft trajectories," *Robot. Auto. Syst.*, vol. 98, pp. 158–173, Dec. 2017.

[8] M. Radmanesh and M. Kumar, "Flight formation of UAVs in presence of moving obstacles using fast-dynamic mixed integer linear programming," *Aerosp. Sci. Technol.*, vol. 50, pp. 149–160, Mar. 2016.

[9] J. Bertram, X. Yang, M. W. Brittain, and P. Wei, "Online flight planner with dynamic obstacles for urban air mobility," in *Proc. AIAA Aviation Forum*, Dallas, TX, USA, Jun. 2019, p. 3625.

[10] M. Cesar Paes Santos, C. Dario Rosales, M. Sarcinelli-Filho, and R. Carelli, "A novel Null-Space-Based UAV trajectory tracking controller with collision avoidance," *IEEE/ASME Trans. Mechatronics*, vol. 22, no. 6, pp. 2543–2553, Dec. 2017.

[11] J. Sun, J. Tang, and S. Lao, "Collision avoidance for cooperative UAVs with optimized artificial potential field algorithm," *IEEE Access*, vol. 5, pp. 18382–18390, 2017.

[12] Y. Du, X. Zhang, and Z. Nie, "A real-time collision avoidance strategy in dynamic airspace based on dynamic artificial potential field algorithm," *IEEE Access*, vol. 7, pp. 169469–169479, 2019.

[13] A. Marchidan and E. Bakolas, "Collision avoidance for an unmanned aerial vehicle in the presence of static and moving obstacles," *J. Guid., Control, Dyn.*, vol. 43, no. 1, pp. 96–110, Jan. 2020.

[14] D. Choi, K. Lee, and D. Kim, "Enhanced potential field-based collision avoidance for unmanned aerial vehicles in a dynamic environment," in *Proc. AIAA Scitech Forum*, 2020, p. 0487.

[15] J. P. Wilhelm and G. Clem, "Vector field UAV guidance for path following and obstacle avoidance with minimal deviation," *J. Guid., Control, Dyn.*, vol. 42, no. 8, pp. 1848–1856, Aug. 2019.

[16] X. Zhu, Y. Liang, and M. Yan, "A flexible collision avoidance strategy for the formation of multiple unmanned aerial vehicles," *IEEE Access*, vol. 7, pp. 140743–140754, 2019.

[17] H. L. N. N. Thanh and S. K. Hong, "Completion of collision avoidance control algorithm for multicopters based on geometrical constraints," *IEEE Access*, vol. 6, pp. 27111–27126, 2018.

[18] V. Tchuiev and T. Shima, "Intercept angle guidance in an obstacle-rich environment," *J. Guid., Control, Dyn.*, vol. 40, no. 6, pp. 1507–1518, Jun. 2017.

[19] S. R. Kumar, M. Weiss, and T. Shima, "Minimum-effort intercept angle guidance with multiple-obstacle avoidance," *J. Guid., Control, Dyn.*, vol. 41, no. 6, pp. 1355–1369, Jun. 2018.

[20] S.-C. Han, H. Bang, and C.-S. Yoo, "Proportional navigation-based collision avoidance for UAVs," *Int. J. Control, Autom. Syst.*, vol. 7, no. 4, pp. 553–565, Aug. 2009.

[21] D. Zuehlke, N. Prabhakar, M. Clark, T. Henderson, and R. J. Prazenica, "Vision-based object detection and proportional navigation for UAS collision avoidance," in *Proc. AIAA Scitech Forum*, San diego, CA, USA, Jan. 2019, p. 0960.

[22] A. Chakravarthy and D. Ghose, "Obstacle avoidance in a dynamic environment: A collision cone approach," *IEEE Trans. Syst., Man, Cybern., A, Syst. Humans*, vol. 28, no. 5, pp. 562–574, Sep. 1998.

[23] A. Chakravarthy and D. Ghose, "Generalization of the collision cone approach for motion safety in 3-D environments," *Auto. Robots*, vol. 32, no. 3, pp. 243–266, Apr. 2012.

[24] J. Park and N. Cho, "Collision avoidance of hexacopter UAV based on LiDAR data in dynamic environment," *Remote Sens.*, vol. 12, no. 6, p. 975, Mar. 2020.

[25] J. Goss, R. Rajvanshi, and K. Subbarao, "Aircraft conflict detection and resolution using mixed geometric and collision cone approaches," in *Proc. AIAA Guid., Navigat., Control Conf. Exhibit*, Providence, RI, USA, Aug. 2004, p. 4879.

[26] Y. Watanabe, A. Calise, and E. Johnson, "Vision-based obstacle avoidance for UAVs," in *Proc. AIAA Guid., Navigat. Control Conf. Exhibit*, Hilton Head, SC, USA, Aug. 2007, p. 6829.

[27] J. Park and H. Baek, "Stereo vision based obstacle collision avoidance for a quadrotor using ellipsoidal bounding box and hierarchical clustering," *Aerosp. Sci. Technol.*, vol. 103, Aug. 2020, Art. no. 105882.

[28] H. Choi, Y. Kim, and I. Hwang, "Reactive collision avoidance of unmanned aerial vehicles using a single vision sensor," *J. Guid., Control, Dyn.*, vol. 36, no. 4, pp. 1234–1240, Jul. 2013.

[29] J. Park and Y. Kim, "Collision avoidance for quadrotor using stereo vision depth maps," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 51, no. 4, pp. 3226–3241, Oct. 2015.

[30] H.-S. Shin, A. Tsourdos, B. White, M. Shanmugavel, and M.-J. Tahk, "UAV conflict detection and resolution for static and dynamic obstacles," in *Proc. AIAA Guid., Navigat. Control Conf. Exhibit*, Aug. 2008, p. 6521.

[31] B. A. White, H.-S. Shin, and A. Tsourdos, "UAV obstacle avoidance using differential geometry concepts," in *Proc. 18th IFAC World Congr.* Milano, Italy: Elsevier, Sep. 2011, pp. 6325–6330.

[32] J. Seo, Y. Kim, S. Kim, and A. Tsourdos, "Collision avoidance strategies for unmanned aerial vehicles in formation flight," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 53, no. 6, pp. 2718–2734, Dec. 2017.

[33] L.-P. Tsao, C.-L. Chou, C.-M. Chen, and C.-T. Chen, "Aiming point guidance law for air-to-air missiles," *Int. J. Syst. Sci.*, vol. 29, no. 2, pp. 95–102, Feb. 1998.

[34] A. Mujumdar and R. Padhi, "Reactive collision avoidance of using nonlinear geometric and differential geometric guidance," *J. Guid., Control, Dyn.*, vol. 34, no. 1, pp. 303–311, Jan. 2011.

[35] J.-W. Park, H.-D. Oh, and M.-J. Tahk, "UAV collision avoidance based on geometric approach," in *Proc. SICE Annu. Conf.*, Aug. 2008, pp. 2122–2126.

[36] M. Sidar and B. Doolin, "On the feasibility of real-time prediction of aircraft carrier motion at sea," *IEEE Trans. Autom. Control*, vol. 28, no. 3, pp. 350–356, Mar. 1983.

[37] R. E. Kalman and R. S. Bucy, "New results in linear filtering and prediction theory," *J. Basic Eng.*, vol. 83, no. 1, pp. 95–108, Mar. 1961.

JONGHO PARK received the B.S., M.S., and Ph.D. degrees in aerospace engineering from the Department of Mechanical and Aerospace Engineering, Seoul National University, Seoul, Republic of Korea, in 2010, 2012, and 2016, respectively. From 2016 to 2019, he was a Senior Researcher with the Guidance and Control Team, Agency for Defense Development, Daejeon, Republic of Korea. He joined the Faculty of Ajou University, in 2019, where he is currently an Assistant Professor with the Department of Military Digital Convergence. His current research interest includes guidance, control, and application of aerospace systems.

NAMHOON CHO (Member, IEEE) received the B.S. and Ph.D. degrees in mechanical and aerospace engineering from Seoul National University, South Korea, in August 2012 and February 2017, respectively. He was a Research Fellow with the Department of Mechanical and Aerospace Engineering, Seoul National University, from March 2017 to February 2019. He is currently a Senior Researcher with the Agency for Defense Development, working on research and development of guidance and control systems. His current research interests include guidance algorithms with applications to UAVs and missiles, unified frameworks for optimal feedback control, model estimation for robust adaptive control, and control architectures combining machine learning paradigms.

SEOKWON LEE received the B.S. and Ph.D. degrees in aerospace engineering from the Department of Mechanical and Aerospace Engineering, Seoul National University, Seoul, South Korea, in 2012 and 2019, respectively. He was a Postdoctoral Researcher with the Institute of Advanced Aerospace Technology, Seoul National University, from September 2019 to October 2020. He is currently a Research Fellow with the Centre for Autonomous and Cyber-Physical Systems, School of Aerospace, Transport and Manufacturing, Cranfield University. His research interests include nonlinear control, guidance and control of missiles, and data-driven control.