

Article

# Collision Avoidance of Hexacopter UAV Based on LiDAR Data in Dynamic Environment

Jongho Park <sup>1</sup> and Namhoon Cho <sup>2,\*</sup><sup>1</sup> Department of Military Digital Convergence, Ajou University, Suwon 16499, Korea; parkjo05@ajou.ac.kr<sup>2</sup> The 1st R&D Institute, Agency for Defense Development, Daejeon 34186, Korea

\* Correspondence: nhcho91@gmail.com; Tel.: +82-42-821-4821

Received: 3 March 2020; Accepted: 13 March 2020; Published: 18 March 2020



**Abstract:** A reactive three-dimensional maneuver strategy for a multirotor Unmanned Aerial Vehicle (UAV) is proposed based on the collision cone approach to avoid potential collision with a single moving obstacle detected by an onboard sensor. A Light Detection And Ranging (LiDAR) system is assumed to be mounted on a hexacopter to obtain the obstacle information from the collected point clouds. The collision cone approach is enhanced to appropriately deal with the moving obstacle with the help of a Kalman filter. The filter estimates the position, velocity, and acceleration of the obstacle by using the LiDAR data as the associated measurement. The obstacle state estimate is utilized to predict the future trajectories of the moving obstacle. The collision detection and obstacle avoidance maneuver decisions are made considering the predicted trajectory of the obstacle. Numerical simulations, including a Monte Carlo campaign, are conducted to verify the performance of the proposed collision avoidance algorithm.

**Keywords:** unmanned aerial vehicle; hexacopter; collision avoidance; guidance algorithm; dynamic environment

## 1. Introduction

In the last decade, Unmanned Aerial Vehicles (UAVs) have achieved significant advances in scientific research as well as in real-world applications. Especially, rotary-wing UAVs have been employed for a wide range of application areas due to their vertical takeoff and landing capability. Nowadays, various types of UAVs with different structures keep emerging. For instance, UAVs with six or more rotors have attracted attention due to their higher payload capacity [1–5]. Furthermore, various sensors have been mounted on the UAVs for applications such as autonomous formation flight, surveillance, target tracking, reconnaissance, disaster assistance, etc.

Light Detection And Ranging (LiDAR) is one type of active remote sensing and has been widely employed as a primary ranging device in many autonomous robotic vehicles including UAVs [6–10]. Wallace et al. [11] presented a UAV-borne LiDAR system to collect measurements for forest inventory applications. A Sigma Point Kalman Smoother (SPKS) was applied to combine observations from different sensors. Then, Wallace et al. [12] developed a tree detection algorithm using high-density LiDAR data. The effect of point density on the accuracy of tree detection and delineation was evaluated. On the other hand, Chen et al. [13] integrated LiDAR and UAV to extract detailed surface geologic information. The random sample consensus method was adopted to ascertain the best-fit plane of bedding. Hening et al. [14] presented a data fusion technique for the position and velocity estimation of UAVs. Local position information was provided by a LiDAR sensor, and an adaptive Kalman filter was applied for the integration. Zheng et al. [15] built an obstacle detection system using LiDAR with the consideration of a moving point cloud.

To fully take advantage of UAVs in the real world, a high degree of flight safety related to a close encounter with another UAV should be guaranteed. In a real situation, the obstacle information is unknown prior to deployment and it should be detected by a sensor mounted on the UAV during the mission. The UAV must be capable of avoiding the detected obstacle using the online-collected information. In other words, obstacle detection and avoidance is one of the most important capabilities for the success of UAV applications. Therefore, it is crucial to develop a collision avoidance strategy for the UAV to perform a given mission in a dynamic environment.

Collision avoidance against a moving obstacle has been investigated by various researchers. In the robotics field, Cherubini et al. [16] designed a set of paths during navigation to take into account the obstacle velocity, which was estimated by a Kalman filter. The estimated velocity was used to predict the obstacle's position within a tentacle-based approach. Ji et al. [17] presented a path planning and tracking framework for autonomous vehicles. The tracking task was formulated as a multiconstrained model predictive control problem, and the front wheel steering angle was calculated to prevent the vehicle from colliding with a moving obstacle. Malone et al. [18] proposed a scheme to incorporate the effect of likely obstacle motion into the desired path of a robot. The method combined an artificial potential field with stochastic reachable set for online path planning. Li et al. [19] developed a controller based on a nonlinear model predictive control for obstacle avoidance. A moving trend function was constructed to predict dynamic obstacle position variances in the prediction horizon. Zhang et al. [20] focused on the tracking of an object in the presence of multiple dynamic obstacles. A method based on conservation of energy was proposed to adjust the motion states of the robot manipulators.

On the other hand, in the aerospace field, Mujumdar and Padhi [21] developed nonlinear geometric guidance and differential geometric guidance, based on aiming point guidance. The guidance law of the UAV was made applicable for moving obstacles by incorporating the Point of Closest Approach (PCA). Wang and Xin [22] addressed the formation control problem where other UAVs are treated as moving obstacles. A nonquadratic avoidance cost function was constructed by an inverse optimal control approach so that the optimal control law was obtained in an analytical form. Yang et al. [23] designed a switching controller to achieve a spatial collision avoidance strategy for fixed-wing UAV. The UAV maintained a safe distance from an intruder by keeping the desired relative bearing and elevation during a collision course. Gageik et al. [24] demonstrated an autonomous flight with low-cost sensors. The movement of an obstacle was handled by the parametrization of the weighted filter. Radmanesh and Kumar [25] formulated a mixed-integer linear program for the path planning problem by incorporating the collision avoidance requirements as the constraints. The constraint relations enforced the formation to visit waypoints and to avoid collision with any intruder aircraft. Lin and Saripalli [26] proposed a sampling-based path planning method based on the Closed-Loop Rapidly-Exploring Random Tree (CL-RRT) algorithm. Using the reachable set, the candidate trajectory method was made robust against unexpected obstacle maneuvers. Santos et al. [27] presented a trajectory tracking controller based on a time-variant artificial potential field with hierarchical objectives using a behavior-based approach. Obstacle motion was taken into account through the variations of the potential function. Marchidan and Bakolas [28] utilized a parameterized vector field that was generated from the decomposition of agent kinematics into normal and tangential components. With a parametric proximity-based eigenvalue function, the agent's maneuver could be adjusted with little computation effort.

The capability of hovering over a designated position is the unique property of the rotary-wing UAVs enabling various useful applications. For example, with the development of communication techniques, UAV Base Station (UAV-BS) becomes a promising approach to provide wireless connectivity to ground user equipment or other aerial nodes [29–32]. Moreover, UAV-BS can be utilized in enhancing the coverage and rate performance of the wireless communication networks in scenarios such as an emergency situation. As a base station, it is important for the UAV to perform station keeping at its current location to serve the incoming flow. That is, the performance of the UAV-BS depends on the UAV's flight time. In this time period, the UAV also estimates the network state and feeds this information to a central agent. Considering its deployment to dangerous sites as a first responder

service, the airborne base station should remain intact even in the environment where the presence of an unknown dynamic obstacle is highly likely. Therefore, it is necessary to develop a collision avoidance algorithm which is appropriate during the hovering maneuver of the UAV.

In this study, an enhanced collision avoidance algorithm based on the collision cone approach [33,34] is proposed to deal with a three-dimensional dynamic environment. A hexacopter is considered as the flying platform [35]. The UAV is equipped with a Global Positioning System (GPS) and Attitude and Heading Reference System (AHRS) so that the position and attitude information can be obtained. A conventional Proportional-Derivative (PD) control system is designed to enable waypoint guidance where position and yaw angle commands are fed from the proposed collision avoidance algorithm. An obstacle is completely unknown to the hexacopter initially, and it is detected by a LiDAR system during the flight. From a practical standpoint, the LiDAR has limited Field-of-View (FoV) and sensing range. The obtained obstacle data points are utilized to form a spherical bounding box and collision cone. Also, they are used in an estimator to obtain the state of the obstacle. A discrete-time Kalman filter is adopted with its measurement being the center of the bounding box [36]. The obstacle state estimate is utilized to check the collision condition. The predicted trajectory of the obstacle is considered to detect collision. If a potential collision is detected, the state estimate is used to generate aiming point candidates for the avoidance maneuver. The proposed algorithm removes the candidate points that overlap with the predicted path of the obstacle. This enables the hexacopter to consider the kinematics of the obstacle in choosing the direction of the avoidance maneuver. The point closest to the current velocity of the UAV is chosen as the final aiming point. Numerical simulations are performed to demonstrate the performance of the proposed collision avoidance algorithm. As the previous collision cone approach [37] utilizes the velocity vector of the UAV, its application during the hovering maneuver is difficult, as the velocity is nearly a zero vector. However, the proposed algorithm works well in this situation due to the usage of the relative geometry. That is, the proposed algorithm appropriately handles the avoidance against the moving obstacle, in contrast to the previous algorithm. In addition, a Monte Carlo simulation is performed to verify the robustness of the proposed algorithm.

The remainder of this paper is organized as follows. Section 2 describes the dynamics, control system, and data acquisition system of a hexacopter. The background of the collision cone approach is briefly explained in Section 3. In Section 4, the structure of a discrete-time Kalman filter is presented for the estimation. Then, the collision avoidance algorithm for a moving obstacle is proposed in Section 5. The simulation results are presented in Section 6, and the conclusion is given in Section 7.

## 2. UAV System

### 2.1. Dynamics

A hexacopter is configured with six rotors, where three rotors rotate in clockwise direction and the others rotate in counterclockwise direction. A platform of the Tarot 680 hexacopter is shown in Figure 1 as an example. A North-East-Down (NED) coordinate system is considered in an inertial frame. Note that flat-Earth and rigid body are assumed in this study. A body-fixed coordinate system is defined with its origin at the hexacopter's center of mass and its axes ( $x^b$ ,  $y^b$ , and  $z^b$ ) are aligned with the reference direction vector, as shown in Figure 1.

The rigid hexacopter dynamics is governed by the following translational and rotational differential equations [35].

$$\dot{\vec{r}}^n = C_{b/n}^T \vec{v}^b \quad (1)$$

$$\dot{\vec{v}}^b = \frac{1}{m} \vec{F}^b + C_{b/n} \vec{g}^n - \vec{\omega}^b \times \vec{v}^b \quad (2)$$

$$\dot{\vec{\omega}}^b = J^{-1} (\vec{M}^b - \vec{\omega}^b \times J \vec{\omega}^b) \quad (3)$$

where  $\vec{r}^n \equiv [x^n \ y^n \ z^n]^T$  is the position vector of the hexacopter represented in the NED coordinate system;  $\vec{v}^b \equiv [u \ v \ w]^T$  and  $\vec{\omega}^b \equiv [p \ q \ r]^T$  are the velocity vector and the angular velocity of the hexacopter

represented in the body-fixed coordinate system, respectively;  $m$  and  $J \equiv \text{diag}[J_x, J_y, J_z]$  are the mass and inertia matrix of the hexacopter, respectively;  $\vec{F}^b$  is the external force of the hexacopter represented in the body-fixed coordinate system;  $\vec{M}^b$  is the external moment of the hexacopter represented in the body-fixed coordinate system;  $\vec{g}^n \equiv [0 \ 0 \ g]^T$  is the gravity vector represented in the NED coordinate system; and  $C_{b/n} (= C_{n/b}^T)$  is the direction cosine matrix performing coordinate transformation from the NED coordinate system to the body-fixed coordinate system, which is defined with a (3-2-1) Euler sequence as follows,

$$C_{b/n} = \begin{bmatrix} \cos \theta \cos \psi & \cos \theta \sin \psi & -\sin \theta \\ -\cos \phi \sin \psi + \sin \phi \sin \theta \cos \psi & \cos \phi \cos \psi + \sin \phi \sin \theta \sin \psi & \sin \phi \cos \theta \\ \sin \phi \sin \psi + \cos \phi \sin \theta \cos \psi & -\sin \phi \cos \psi + \cos \phi \sin \theta \sin \psi & \cos \phi \cos \theta \end{bmatrix} \quad (4)$$

where Euler angle  $\vec{\Phi} \equiv [\phi \ \theta \ \psi]^T$  is defined between the NED and body-fixed coordinate systems.

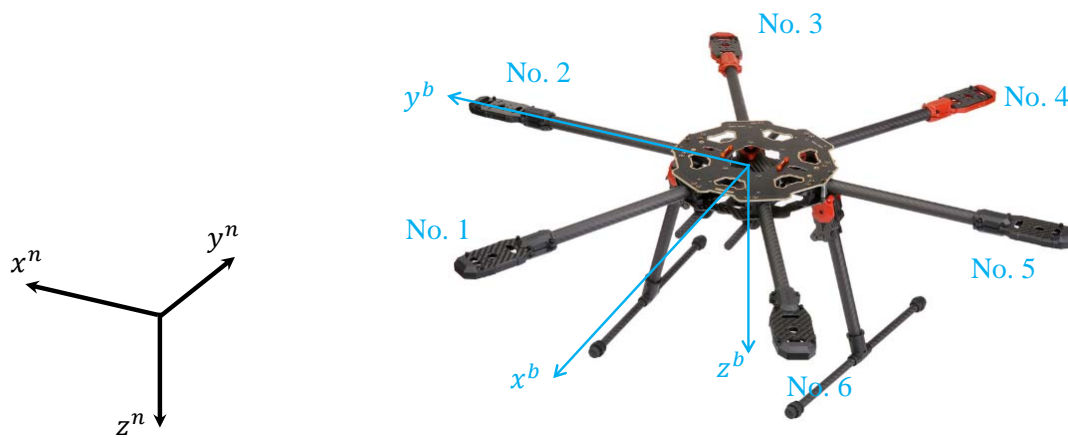


Figure 1. Tarot 680 hexacopter platform and its coordinate systems.

The external force and moment of the hexacopter can be expressed as follows.

$$\vec{F}^b = \begin{bmatrix} 0 \\ 0 \\ -T \end{bmatrix}, \quad \vec{M}^b = \begin{bmatrix} L \\ M \\ N \end{bmatrix} \quad (5)$$

Then,  $\vec{F}^b$  and  $\vec{M}^b$  have the following relationship with the angular velocity of the rotor,

$$\begin{bmatrix} T \\ L \\ M \\ N \end{bmatrix} = \begin{bmatrix} k_f & k_f & k_f & k_f & k_f & k_f \\ -\frac{1}{2}k_f l & -k_f l & -\frac{1}{2}k_f l & \frac{1}{2}k_f l & k_f l & \frac{1}{2}k_f l \\ \frac{\sqrt{3}}{2}k_f l & 0 & -\frac{\sqrt{3}}{2}k_f l & -\frac{\sqrt{3}}{2}k_f l & 0 & \frac{\sqrt{3}}{2}k_f l \\ k_\tau & -k_\tau & k_\tau & -k_\tau & k_\tau & -k_\tau \end{bmatrix} \begin{bmatrix} w_1^2 \\ w_2^2 \\ w_3^2 \\ w_4^2 \\ w_5^2 \\ w_6^2 \end{bmatrix} \quad (6)$$

where  $k_f$  and  $k_\tau$  are the thrust and drag coefficients of the rotor, respectively;  $l$  is the distance from the hexacopter's center of mass to the rotor; and  $w_i$  is the angular velocity of the  $i$ -th rotor with  $i \in \{1, \dots, 6\}$ .

## 2.2. Control Design

In this study, a PD controller is designed to track the desired position vector  $(x_d^n, y_d^n, \text{ and } z_d^n)$  and yaw angle  $(\psi_d)$ . First, the collective thrust command,  $T$ , is computed to handle the desired altitude of the hexacopter, which is represented by  $z_d^n$ .

$$T = m \frac{2\zeta_v \omega_v \dot{z}^n + \omega_v^2 (z^n - z_d^n) + g}{\cos \phi \cos \theta} \quad (7)$$

where  $\zeta_v$  and  $\omega_v$  are design parameters. The horizontal position controller equations are given as

$$a_{x,cmd} = -2\zeta_h \omega_h \dot{x}^n + \omega_h^2 (x_d^n - x^n) \quad (8)$$

$$a_{y,cmd} = -2\zeta_h \omega_h \dot{y}^n + \omega_h^2 (y_d^n - y^n) \quad (9)$$

$$\begin{bmatrix} a_{u,cmd} \\ a_{v,cmd} \\ a_{w,cmd} \end{bmatrix} = C_{b/n} \begin{bmatrix} a_{x,cmd} \\ a_{y,cmd} \\ 0 \end{bmatrix} \quad (10)$$

where  $\zeta_h$  and  $\omega_h$  are design parameters. Now, the desired roll and pitch angles,  $\phi_d$  and  $\theta_d$ , respectively, can be obtained as follows.

$$\phi_d = \text{atan2}(a_{v,cmd}, g) \quad (11)$$

$$\theta_d = \text{atan2}(-a_{u,cmd}, g) \quad (12)$$

Then, the commanded roll, pitch, and yaw moments,  $L$ ,  $M$ , and  $N$ , respectively, can be computed for the attitude loop as follows,

$$L = J_x \left( -2\zeta_\phi \omega_\phi \dot{p} + \omega_\phi^2 (\phi_d - \phi) \right) \quad (13)$$

$$M = J_y \left( -2\zeta_\theta \omega_\theta \dot{q} + \omega_\theta^2 (\theta_d - \theta) \right) \quad (14)$$

$$N = J_z \left( -2\zeta_\psi \omega_\psi \dot{p} + \omega_\psi^2 (\psi_d - \psi) \right) \quad (15)$$

where  $\zeta_\phi$ ,  $\zeta_\theta$ ,  $\zeta_\psi$ ,  $\omega_\phi$ ,  $\omega_\theta$ , and  $\omega_\psi$  are design parameters. Finally, the angular velocities of the six rotors are allocated by using Equation (6) with  $T$ ,  $L$ ,  $M$ , and  $N$  obtained in Equations (7), (13)–(15), respectively. Note that  $x_d^n$ ,  $y_d^n$ ,  $z_d^n$ , and  $\psi_d$  are delivered from the proposed collision avoidance guidance system described in Section 5.

## 2.3. LiDAR Data Sensing

Let us assume that an obstacle consists of a point cloud that represents its external surface. The LiDAR with limited FoV is mounted on the hexacopter and directed along the  $+x^b$ -axis to acquire obstacle data. The visibility of the obstacle point by the LiDAR can be determined by examining whether the obstacle data point is inside a rectangular pyramid whose vertices are  $o^b$ ,  $v_1$ ,  $v_2$ ,  $v_3$ , and  $v_4$ , as shown in Figure 2.  $v_1$ ,  $v_2$ ,  $v_3$ , and  $v_4$  in the body-fixed coordinate system can be obtained as follows.

$$\begin{aligned} v_1 &= \left[ d_a \quad -d_a \tan(H_{fv}/2) \quad -d_a \tan(V_{fv}/2) \right]^T \\ v_2 &= \left[ d_a \quad d_a \tan(H_{fv}/2) \quad -d_a \tan(V_{fv}/2) \right]^T \\ v_3 &= \left[ d_a \quad d_a \tan(H_{fv}/2) \quad d_a \tan(V_{fv}/2) \right]^T \\ v_4 &= \left[ d_a \quad -d_a \tan(H_{fv}/2) \quad d_a \tan(V_{fv}/2) \right]^T \end{aligned} \quad (16)$$

where  $H_{fv}$  and  $V_{fv}$  are the horizontal and vertical FoV of the LiDAR, respectively, and  $d_a$  is a sensing range of the LiDAR. The hexacopter performs its mission with constant obstacle data acquisition from the LiDAR.

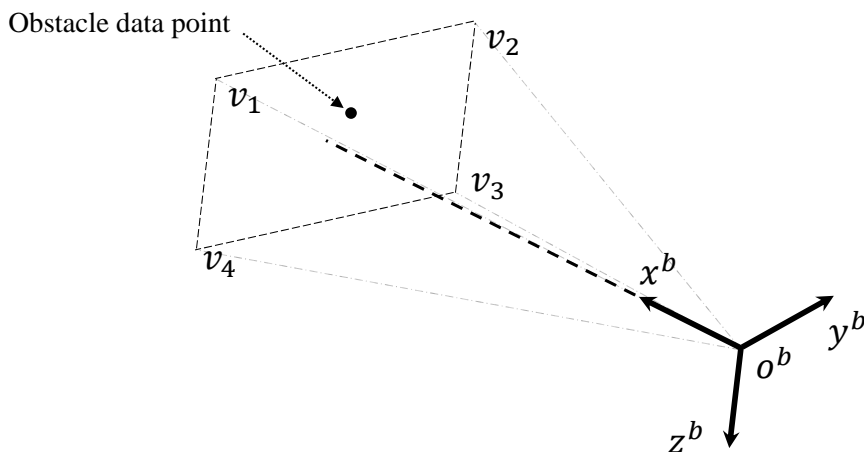


Figure 2. Obstacle data acquisition with Light Detection And Ranging (LiDAR).

### 3. Review of Basic Collision Cone Approach

In this study, the collision cone method is adopted as the basis of the collision detection and avoidance logic [33,34]. Let us assume that  $n_c$  data points are accumulated by the LiDAR, and that the position vector of the  $k$ -th obstacle data point in the NED coordinate system is defined as  $\vec{r}_{o,k}^n$ . Among  $n_c$  data points, there are two points associated with the longest pairwise Euclidean distance. Let us define these points as  $\vec{r}_{o,1}^n$  and  $\vec{r}_{o,2}^n$ . Then, the center of a spherical bounding box can be obtained as follows.

$$\vec{r}_{o,c}^n = \frac{\vec{r}_{o,1}^n + \vec{r}_{o,2}^n}{2} \tag{17}$$

Also, the radius of the bounding box can be designed as follows,

$$r_s = \frac{\|\vec{r}_{o,1}^n - \vec{r}_{o,2}^n\|_2}{2} + d_{sm} \tag{18}$$

where  $d_{sm}$  is a safety margin. Finally, a collision cone can be constructed with a set of tangent lines from the hexacopter to the bounding box, as shown in Figure 3.

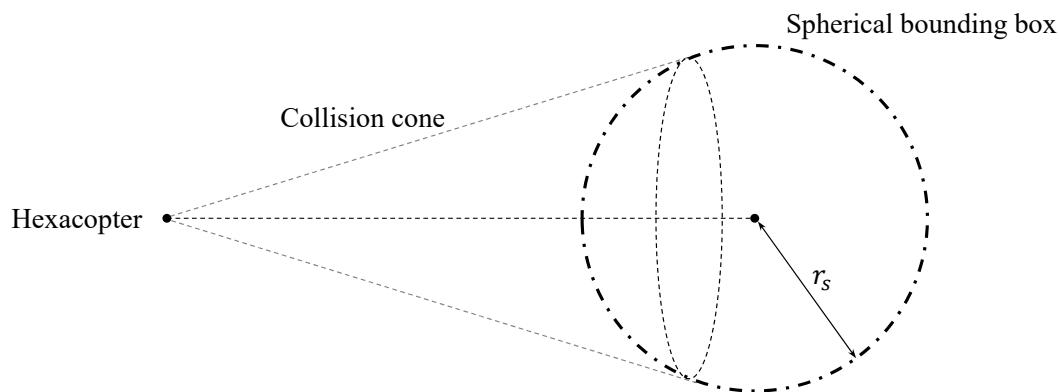


Figure 3. Constructed bounding box and collision cone.

If the velocity vector of the hexacopter is contained within the collision cone, the detected obstacle is a potential threat to the UAV unless it changes its trajectory. In this case, the hexacopter should

perform a collision avoidance maneuver. During the collision avoidance maneuver, the hexacopter is guided to an aiming point instead of the predefined goal point. The aiming point is selected from the points on the intersection curve of the bounding box and the collision cone. In this study, the point closest to the velocity vector of the hexacopter is chosen as the aiming point to minimize the UAV's maneuver. If the velocity vector of the hexacopter is outside the collision cone, then the UAV is guided to the original goal point. This logic is repeated until the hexacopter reaches the goal point.

#### 4. Obstacle State Estimator

As a moving obstacle is considered in this study, its state should be estimated to implement the proposed collision avoidance algorithm described in Section 5. A discrete-time Kalman filter is constructed for the estimation [36]. First, the filter state is defined as follows,

$$\hat{x}_k^n \equiv \left[ x_{e,k}^n \quad \dot{x}_{e,k}^n \quad \ddot{x}_{e,k}^n \quad y_{e,k}^n \quad \dot{y}_{e,k}^n \quad \ddot{y}_{e,k}^n \quad z_{e,k}^n \quad \dot{z}_{e,k}^n \quad \ddot{z}_{e,k}^n \right]^T \quad (19)$$

where  $\hat{r}_{e,k}^n \equiv [x_{e,k}^n \ y_{e,k}^n \ z_{e,k}^n]^T$ ,  $\hat{v}_{e,k}^n \equiv [\dot{x}_{e,k}^n \ \dot{y}_{e,k}^n \ \dot{z}_{e,k}^n]^T$ , and  $\hat{a}_{e,k}^n \equiv [\ddot{x}_{e,k}^n \ \ddot{y}_{e,k}^n \ \ddot{z}_{e,k}^n]^T$  are the position, velocity, and acceleration estimates of the obstacle, respectively, represented in the NED coordinate system. Now, the discrete-time state equation for the estimator can be expressed as follows,

$$\hat{x}_{k+1}^n = \begin{bmatrix} 1 & \Delta t_l & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \Delta t_l & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta t_l & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t_l & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t_l & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t_l \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \hat{x}_k^n + \vec{w}_e \quad (20)$$

where  $\Delta t_l$  is the sampling time of LiDAR data acquisition, and  $\vec{w}_e \equiv [w_x \ w_{\dot{x}} \ w_{\ddot{x}} \ w_y \ w_{\dot{y}} \ w_{\ddot{y}} \ w_z \ w_{\dot{z}} \ w_{\ddot{z}}]^T$  is a process noise characterized by  $N(0, Q_e)$ . On the other hand, the position vector of the center of the bounding box obtained in Equation (17) by the LiDAR system is used as a measurement. Therefore, the measurement equation can be constructed as follows,

$$\vec{r}_k^n = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \hat{x}_k^n + \vec{v}_e \quad (21)$$

where  $\vec{v}_e \sim N(0, R_e)$  is a measurement noise. In this study, an unknown obstacle is detected by LiDAR sensor that has a limited sensing range. The bounding box enclosing the set of detected data points gradually grows in its size as more of the obstacle comes within the sensing range. In this case,  $\vec{r}_{o,c}^n$ , computed in Equation (17), may differ from the actual center of the obstacle. This error can be treated as a measurement noise,  $\vec{v}_e$ , in the observation model. Finally, using Equations (20) and (21), the discrete-time Kalman filter can be implemented to estimate the state of the obstacle.

The Kalman filter is initiated when the obstacle is detected by the LiDAR for the first time. Then, it is updated at every  $\Delta t_l$  with newly measured  $\vec{r}_{o,c}^n$ . If  $\vec{r}_{o,c}^n$  cannot be found at a certain step, then virtual measurement,  $\hat{y}_k^n$ , is modeled to keep the estimator running. This situation may occur when the UAV completely avoids the obstacle and every obstacle data point is out of the LiDAR's

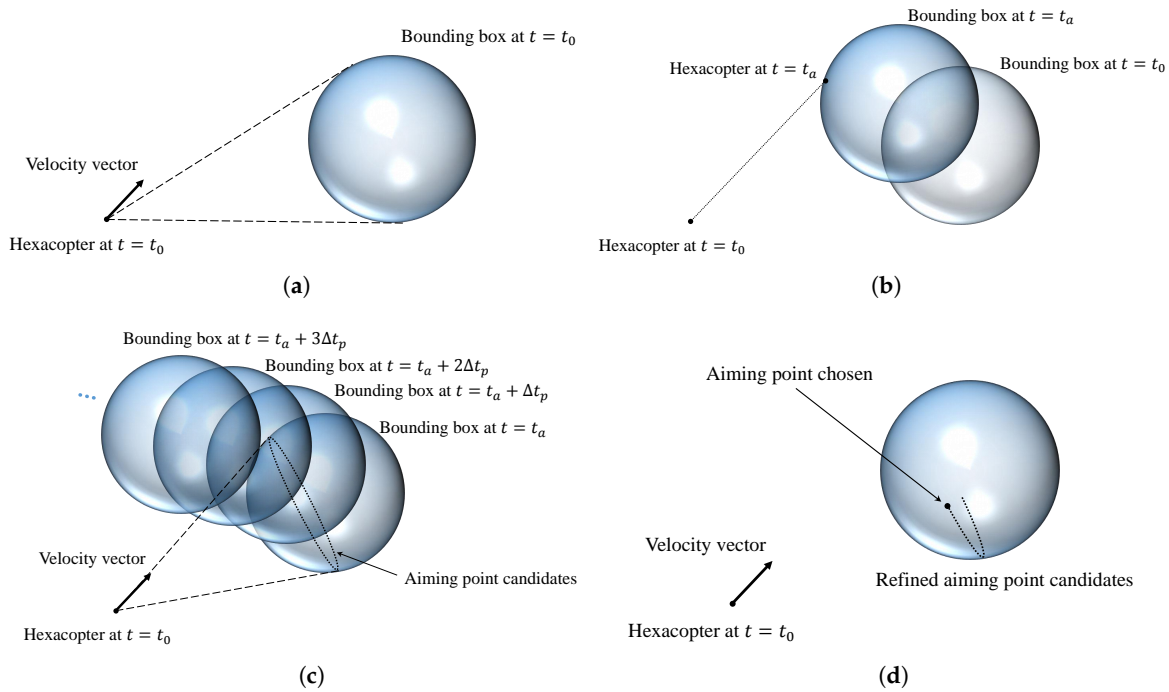
sensing range. The virtual measurement,  $\hat{y}_k^n$ , is modeled with the state estimate at the previous step as follows.

$$\hat{y}_k^n = \frac{1}{2} \Delta t_l^2 \begin{bmatrix} \dot{x}_{e,k-1}^n \\ \dot{y}_{e,k-1}^n \\ \dot{z}_{e,k-1}^n \end{bmatrix} + \Delta t_l \begin{bmatrix} \dot{x}_{e,k-1}^n \\ \dot{y}_{e,k-1}^n \\ \dot{z}_{e,k-1}^n \end{bmatrix} + \begin{bmatrix} x_{e,k-1}^n \\ y_{e,k-1}^n \\ z_{e,k-1}^n \end{bmatrix} \quad (22)$$

When the obstacle is detected again,  $\hat{y}_k^n$  is discarded and the Kalman filter operates with the normal measurement.

## 5. Collision Avoidance Strategy Against Moving Obstacle

As the Kalman filter provides the obstacle information, these state estimates are utilized in the collision detection algorithm and avoidance law considering the dynamic environment. The proposed strategy proceeds in several stages as shown in Figure 4.



**Figure 4.** Collision avoidance strategy using predicted trajectory of obstacle: (a) velocity vector outside the collision cone at  $t = t_0$ , (b) collision at  $t = t_a$ , (c) pruning process, and (d) final aiming point candidates.

Although an obstacle is within the range of the LiDAR, the velocity vector of the hexacopter is outside the collision cone at  $t = t_0$ , as shown in Figure 4a. Then, the previous approach determines that the UAV is not on a collision course to the obstacle and allows the UAV to continue flying toward the goal point, as it only incorporates the current position and velocity in collision detection. On the contrary, the method proposed in this study utilizes the instantaneous short-term predictions for the obstacle trajectory to determine whether the UAV is on the course to close proximity of the obstacle. The trajectory of the obstacle is predicted by utilizing the state estimate as

$$\hat{r}_e^n = \frac{1}{2} \hat{a}_{e,k}^n \Delta t_d^2 + \hat{v}_{e,k}^n \Delta t_d + \hat{r}_{e,k}^n \quad (23)$$

with the prediction horizon,  $\Delta t_d$ , of various lengths. Also, the future position of the hexacopter can be predicted as follows,

$$\vec{r}_e^n = \vec{v}^n \Delta t_d + \vec{r}^n \quad (24)$$



where  $\vec{v}^n$  is the velocity vector of the hexacopter represented in the NED coordinate system. Note that the acceleration vector is not used in Equation (24) because the abrupt change in the acceleration by the proposed algorithm may result in poor prediction. If the Euclidean distance between the predicted obstacle position,  $\hat{p}_e^n$ , and the predicted UAV position,  $\vec{p}_e^n$ , is smaller than the bounding sphere radius,  $r_s$ , at a certain  $\Delta t_d$ , the UAV determines that it is on a collision course. Let us define this moment as  $t = t_a$ , as shown in Figure 4b. Additionally, note that  $r_s$  is constantly calculated using Equation (18) whenever the obstacle is detected by the LiDAR and the algorithm keeps updating the maximum observed  $r_s$  value for safety reasons, yet this approach might be conservative. The hexacopter is inside the spherical bounding box of the obstacle in Figure 4b, which means a collision.

Now, a collision avoidance maneuver is required. To perform the avoidance maneuver, the collision cone is constructed with the bounding box at  $t = t_a$  and the position of the UAV at  $t = t_0$ , as shown in Figure 4c. The aiming point for the UAV chosen in the direction of the obstacle's future trajectory is likely to increase the chance of a close encounter with the obstacle at the next time step, which is disadvantageous from the viewpoint of minimizing the probability of a possible collision. An additional pruning process is therefore conducted after creating the aiming point candidates to prevent undesirable and ineffective collision avoidance maneuver due to a wrong choice of aiming point. The trace of bounding boxes starting from  $t = t_a$  can be created as shown in Figure 4c with the prediction horizon given by  $\Delta t_p$ . These boxes can then be used to trim the candidate points. If an aiming point candidate is inside the predicted envelope of the bounding boxes at any moment, then that point is eliminated from the candidate set.

Some candidate points may be removed by the pruning process as shown in Figure 4d. Then, the final aiming point is chosen from the remaining candidate points. Even though the point closest to the velocity vector of the hexacopter is chosen, the chosen point may be far from the velocity vector because of the elimination step, as shown in Figure 4d.

The aiming point provided as the final result of the collision avoidance strategy comprises the desired position part of the guidance command. Meanwhile, the desired yaw angle is designed so that the LiDAR faces toward the center of the spherical bounding box.

$$\psi_d = \arctan \frac{y_{e,k}^n - y^n}{x_{e,k}^n - x^n} \quad (25)$$

The reasoning behind the yaw angle command design is to keep the obstacle inside of the LiDAR's FoV for the consecutive state estimation.

The overall flowchart of the proposed collision avoidance algorithm is shown in Figure 5.

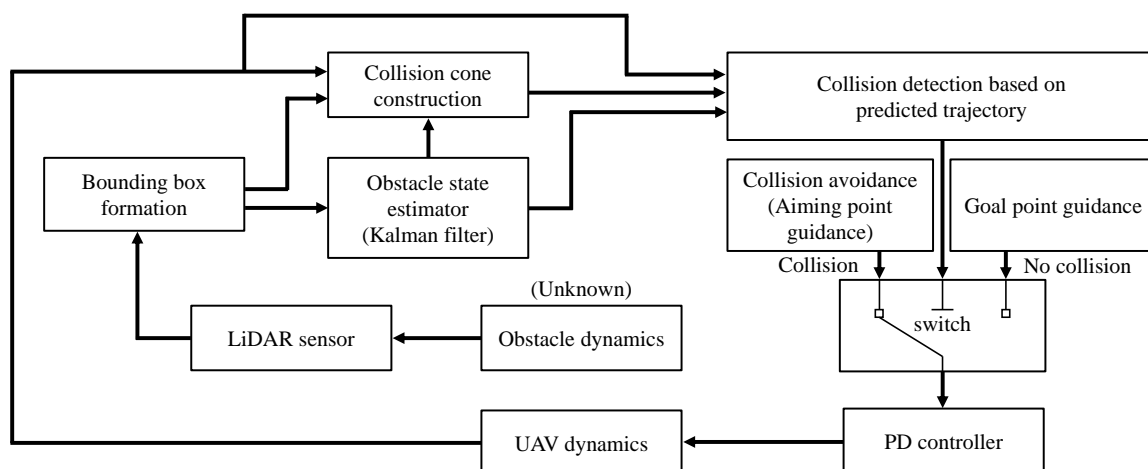


Figure 5. Flowchart of the proposed collision avoidance process.

## 6. Numerical Simulation

Numerical simulations are conducted to demonstrate the performance of the proposed collision avoidance algorithm. The parameters of the hexacopter dynamics and control system, estimator, and collision avoidance used in the numerical simulation are listed in Table 1.

The time step for numerical integration and control update is 0.01 s, whereas the sampling time of the LiDAR data acquisition,  $\Delta t_l$ , is set to 0.1 s. The slower rate for the LiDAR measurement update is due to the large amount of computational effort required to process the point cloud data in each cycle. Not only the obstacle state estimator, but also the collision avoidance guidance algorithm, is triggered by the arrival of new LiDAR data. Consequently, they have the same update rate of 10Hz, and the UAV's autopilot utilizes the latest updated command.

For the control system of the hexacopter,  $\phi_d$  and  $\theta_d$  in Equations (11) and (12) are limited in the range of  $\pm 20^\circ$  to avoid excessive maneuver. Meanwhile,  $P_0^+$  in Table 1 is the covariance of the initial estimation error of  $\hat{x}_0^n$ , where diag refers to a square matrix with the elements inside the square bracket being the diagonal entries. As an obstacle is completely unknown initially, there is no choice but to set  $\hat{x}_0^n$  as a zero vector. Additionally, the Kalman filter takes time for the state estimate to converge to a steady-state value. Therefore, the proposed collision avoidance algorithm begins to incorporate the obstacle state estimate a few time steps after the first detection of the obstacle in order to allow a period for the filter state to converge. In this study,  $5\Delta t_l$  (5 steps) is chosen as this period.

For every simulation in this section, the initial position of the hexacopter is set to  $[0\ 0\ -3]^T$  in the NED coordinate system. Also, the initial velocity, angular velocity, and Euler angle of the hexacopter are set to zero.

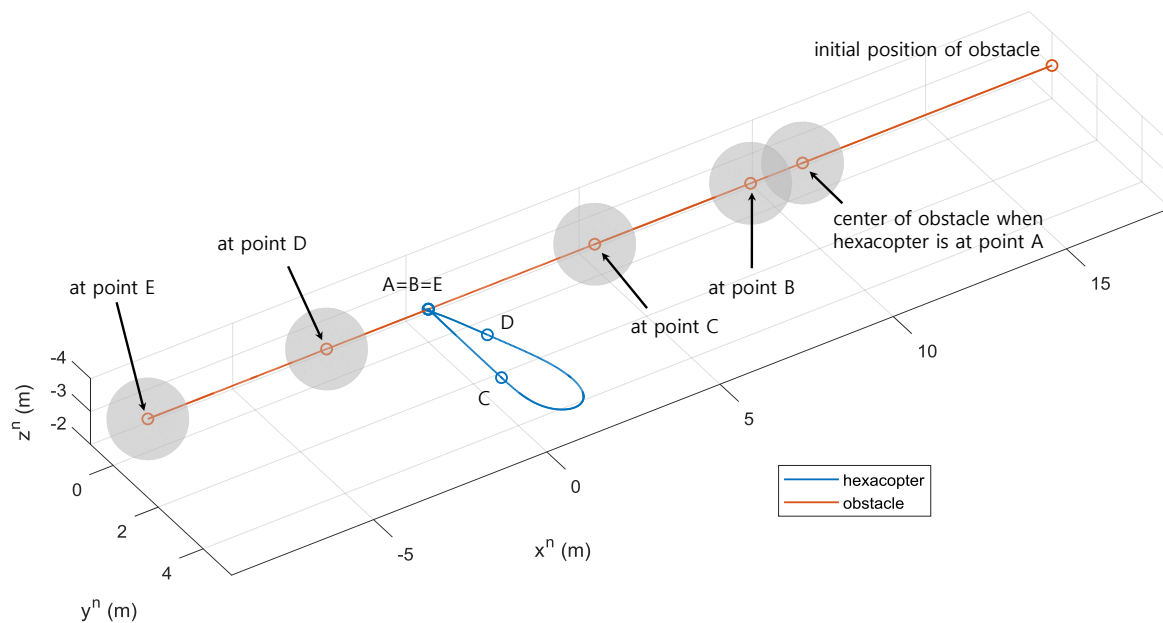
**Table 1.** Parameters of the hexacopter, Kalman filter, and collision avoidance used in numerical simulation.

Hexacopter dynamics							
$m$	2.356 kg	$J_x$	$1.676 \times 10^{-1} \text{ kg m}^2$	$J_y$	$1.676 \times 10^{-1} \text{ kg m}^2$	$J_z$	$2.974 \times 10^{-1} \text{ kg m}^2$
$l$	0.5 m	$k_f$	$1.1 \times 10^{-1} \text{ N s}^2$	$k_\tau$	$5.2 \times 10^{-2} \text{ N s}^2 \text{ m}$	$g$	$9.8 \text{ m/s}^2$
Hexacopter controller							
$\omega_v$	3 rad/s <sup>2</sup>	$\zeta_v$	0.8	$\omega_h$	3 rad/s <sup>2</sup>	$\zeta_h$	0.8
$\omega_\phi$	15 rad/s <sup>2</sup>	$\zeta_\phi$	0.7	$\omega_\theta$	15 rad/s <sup>2</sup>	$\zeta_\theta$	0.7
$\omega_\psi$	5 rad/s <sup>2</sup>	$\zeta_\psi$	0.9				
LiDAR sensor							
$\Delta t_l$	0.1 s	$d_a$	10 m	$H_{fv}$	170°	$V_{fv}$	30°
Kalman filter							
$\hat{x}_0^n$	$[0\ 0\ 0\ 0\ 0\ 0\ 0\ 0]^T$			$P_0^+$	diag[0, 0, 0, 0, 0, 0, 0, 0]		
$Q_e$	diag[ $1 \times 10^{-8}$ , $1 \times 10^{-4}$ , 1, $1 \times 10^{-8}$ , $1 \times 10^{-4}$ , 1, $1 \times 10^{-8}$ , $1 \times 10^{-4}$ , 1]						
$R_e$	diag[ $1 \times 10^{-8}$ , $1 \times 10^{-8}$ , $1 \times 10^{-8}$ ]						
Collision avoidance							
$d_{sm}$	2 m	Max. $\Delta t_d$	30 s	$\Delta t_p$	0.1 s		

### 6.1. Simulation I: Hovering Situation

In Simulation I, the hovering situation is considered, which is the case where the goal point is given by the initial position of the UAV. Namely, the hexacopter stays at  $[0\ 0\ -3]^T$  if a detected obstacle is not considered as a potential threat. An obstacle whose shape is a sphere with the radius of 1 m is considered. Its initial position is set to  $[18\ 0\ -3]^T$  with a constant velocity of  $[-3\ 0\ 0]^T$  so that it is on a collision course.

Figure 6 shows the trajectories of the hexacopter and the obstacle in the NED coordinate system.



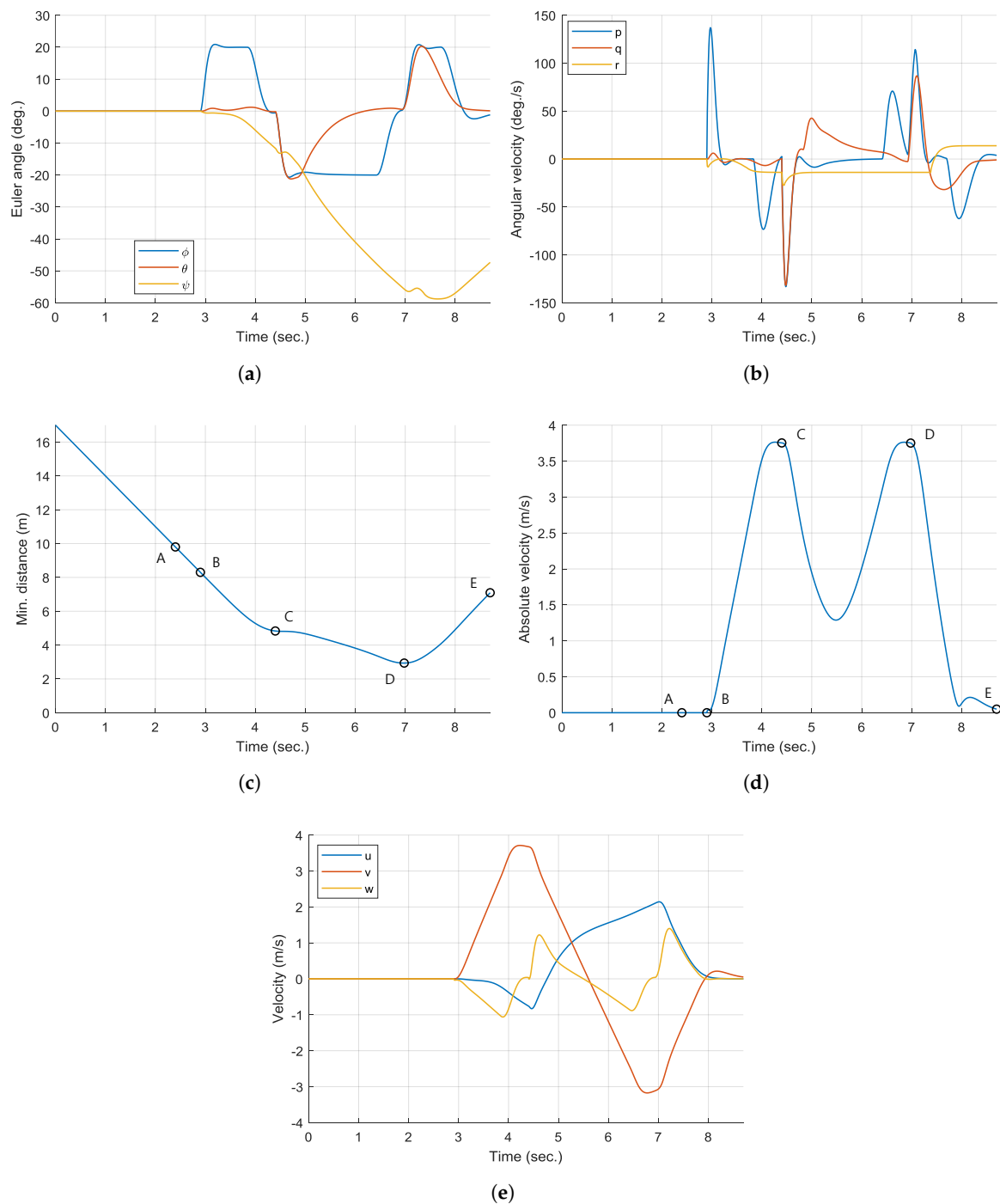
**Figure 6.** Trajectories of the hexacopter and the obstacle (azimuth:  $-33^\circ$ , elevation:  $37^\circ$ , Simulation I).

Additionally, Figure 7 shows the time responses of the attitude, angular velocity, velocity, and absolute velocity of the hexacopter, and the minimum distance between the UAV and the obstacle. Also, Figure 8 shows the time responses of the estimated state along with the true state plotted as a reference of comparison. Note that all state estimates are set to zero at initial state.

Initially ( $t = 0$  s), the obstacle is outside the sensing range of the LiDAR, because  $d_a$  is set to 10 m in Table 1. Then, the obstacle is first detected by the hexacopter at point A ( $t = 2.4$  s), as shown in Figure 7c. However, the UAV does not immediately perform avoidance maneuver, but simply continues hovering. This is because the collision avoidance algorithm incorporates the estimated obstacle information only after the filter transients are died out, as explained above. Moreover, only a small part of the obstacle enters the range of the LiDAR at point A. The computed  $\vec{r}_{o,c}^n$  at point A is  $[9.98 \ 0 \ -3]^T$ , whereas its true state is  $[10.8 \ 0 \ -3]^T$ . This difference may result in a poor estimation. As shown in Figure 8, the state estimates are still at initial zero values at point A. However, after 0.5 s, which is denoted as point B, most of the state estimates are converged to the true states and these are used in the collision avoidance algorithm. Now, the hexacopter predicts the trajectory of the obstacle, and determines that it is on a collision course.

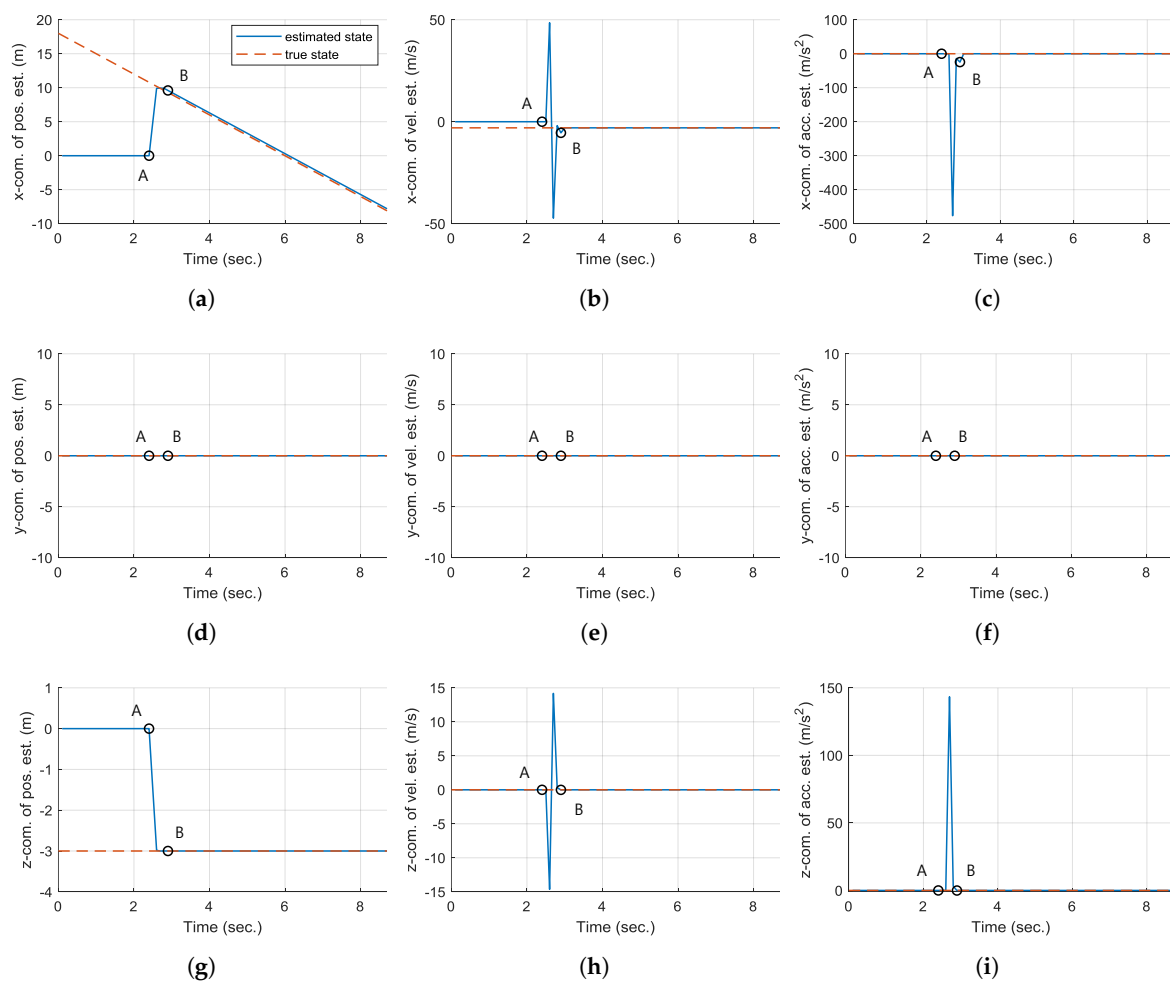
At point B ( $t = 2.9$  s), the hexacopter triggers the collision avoidance maneuver by guiding to the aiming point, instead of the goal point. At the same time, the yaw angle command is generated to direct toward  $[x_{e,k}^n \ y_{e,k}^n \ z_{e,k}^n]^T$  so that the detected obstacle remains inside the LiDAR's FoV, as shown in Figure 7a. At point C ( $t = 4.4$  s), the UAV determines that the future position of the obstacle does not collide with it. That is, the hexacopter is outside of the predicted bounding boxes of the obstacle. Although the UAV heads back to the goal point from this moment, it produces the different trajectory compared to the past one as shown in Figure 6. This is due to the control system of the hexacopter, which generates the change of  $\theta$  in advance of  $\phi$ , as shown in Figure 7a. Simultaneously,  $+x^b$ -axis of the hexacopter is aligned with the goal point by adjusting  $\psi$ .

At point D ( $t = 6.98$  s), the hexacopter reaches the minimum distance of 2.9353 m as shown in Figure 7c. However, considering the direction of the obstacle's velocity vector, it is no longer a threat to the UAV. Finally, at point E ( $t = 8.7$  s), the hexacopter reaches the goal point as shown in Figure 6 and is able to perform hovering maneuver again.



**Figure 7.** Time histories of UAV-related variables (Simulation I): (a) attitude, (b) body rate, (c) minimum distance between UAV and obstacle, (d) absolute velocity, and (e) velocity (body-fixed coordinate system).

In the hovering situation, the velocity vector of the hexacopter is a zero vector, which means that it is difficult to utilize the previous collision cone approach [37]. On the contrary, the proposed algorithm gives consideration to the relative motion as it uses the obstacle information estimated by the Kalman filter. Therefore, the proposed algorithm can be adopted while the UAV is in the hovering state.



**Figure 8.** Time histories of the state estimate (Simulation I): (a)  $x_{e,k'}^n$ , (b)  $\dot{x}_{e,k'}^n$ , (c)  $\ddot{x}_{e,k'}^n$ , (d)  $y_{e,k'}^n$ , (e)  $\dot{y}_{e,k'}^n$ , (f)  $\ddot{y}_{e,k'}^n$ , (g)  $z_{e,k'}^n$ , (h)  $\dot{z}_{e,k'}^n$ , and (i)  $\ddot{z}_{e,k'}^n$ .

## 6.2. Simulation II: Waypoint Guidance

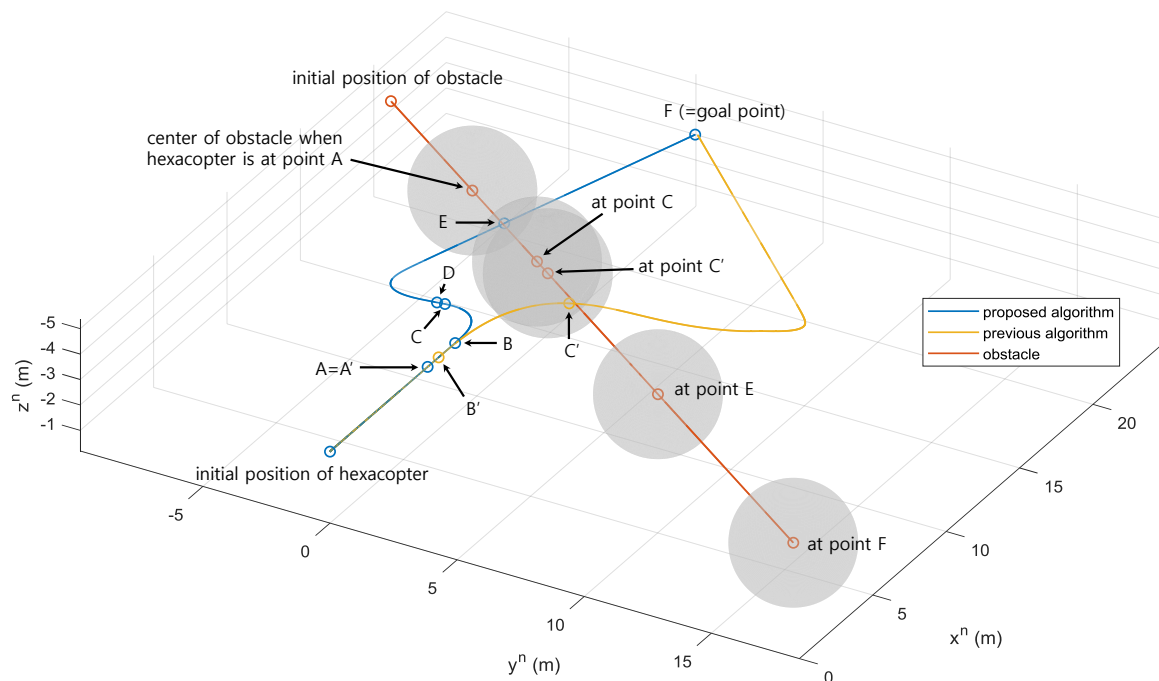
In Simulation II, the proposed collision avoidance algorithm is compared to the previous algorithm [37]. The goal point is set to  $[25 \ 0 \ -3]^T$ , and the hexacopter initially heads for the goal point. A sphere with a radius of 2.2 m is initially located at  $[21 \ -10 \ -3]^T$  and flies toward the path of the UAV. The magnitudes of the velocity vector and the acceleration vector are 2.4 m/s and 0.25 m/s<sup>2</sup>, respectively. Also, their unit vector is  $[-0.56 \ 0.83 \ 0.03]^T$ .

Figure 9 shows the trajectories of the hexacopter with the proposed algorithm and the previous algorithm, and the obstacle in the NED coordinate system. Also, Figure 10 shows the time responses of the attitude, angular velocity, velocity, and absolute velocity of the hexacopter, and the minimum distance between the UAV and the obstacle. Figure 11 shows the time responses of the state estimate of the Kalman filter.

The obstacle enters the range and FoV of the LiDAR at point A (= point A',  $t = 2.4$  s) for both the proposed and previous algorithms, as shown in Figures 9 and 10c. Then, the hexacopter with the proposed algorithm starts collision avoidance maneuver at point B ( $t = 2.9$  s), as the UAV collides with one of the predicted bounding boxes. As shown in Figure 11, the state estimates show sufficient convergence to the true states from point B, which are appropriate to be utilized in the collision avoidance algorithm. On the other hand, the hexacopter with the previous algorithm triggers the avoidance maneuver at point B' ( $t = 2.6$  s), which is earlier than the one with the proposed algorithm. This is because the previous algorithm does not utilize the obstacle state estimator. As soon as the

computed collision cone includes the velocity vector of the UAV, the UAV heads for the aiming point as shown in Figure 9.

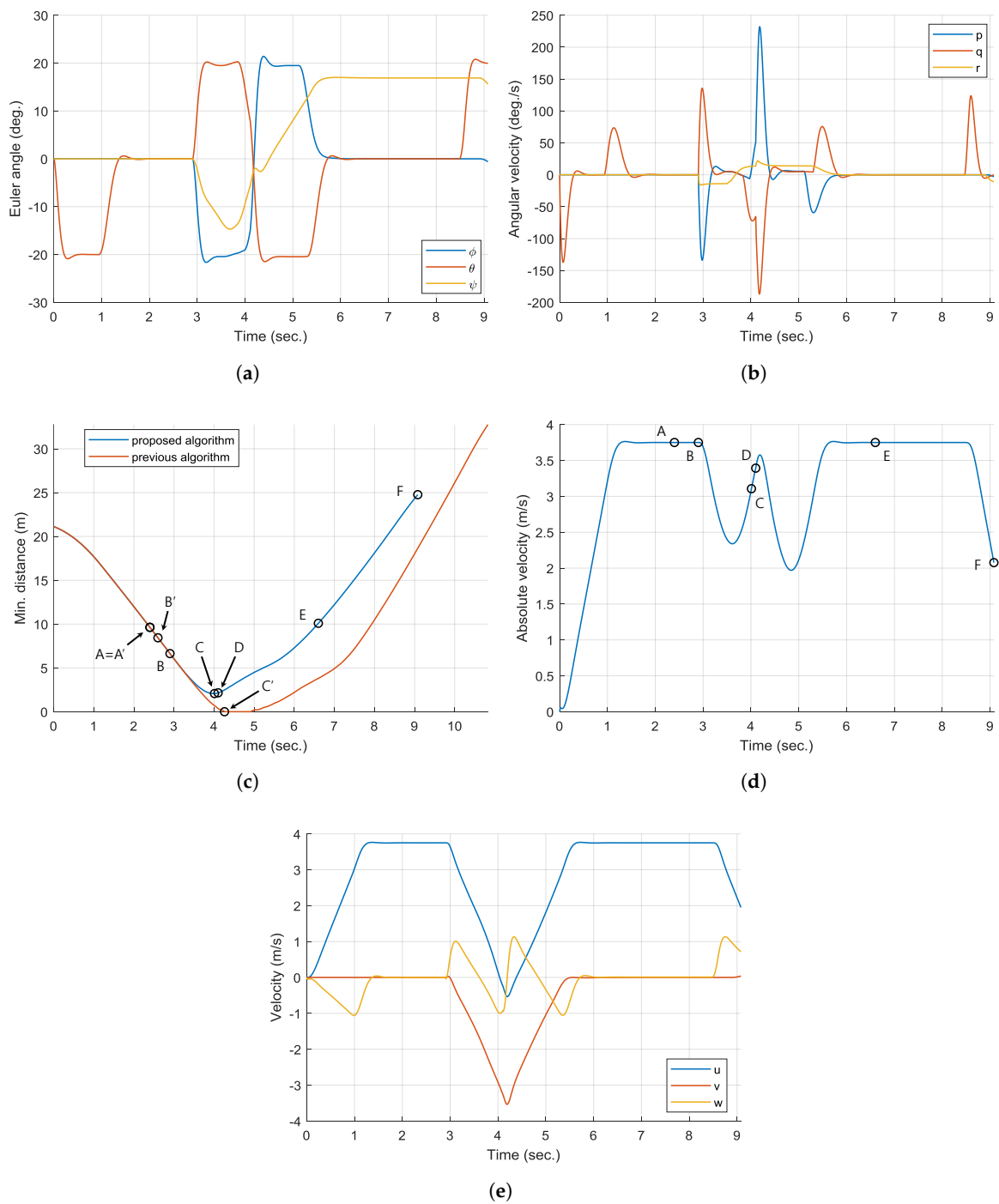
The difference can be witnessed from this moment. As a benefit of incorporating the pruning step in the aiming point processing pipeline, the proposed algorithm selects the aiming point on the opposite side of the obstacle's predicted trajectory. However, the previous algorithm chooses the aiming point without considering the short-term future movement of the obstacle. As a consequence, the UAV cannot withstand the acceleration and the velocity of the obstacle and eventually collides with it at point  $C'$  ( $t = 4.26$  s), as shown in Figure 9. Note that the minimum distance between the UAV and the obstacle is 0 m at point  $C'$  in Figure 10c. That is, although the previous algorithm shows earlier transition to the collision avoidance mode than the proposed algorithm, it fails to avoid the moving obstacle.



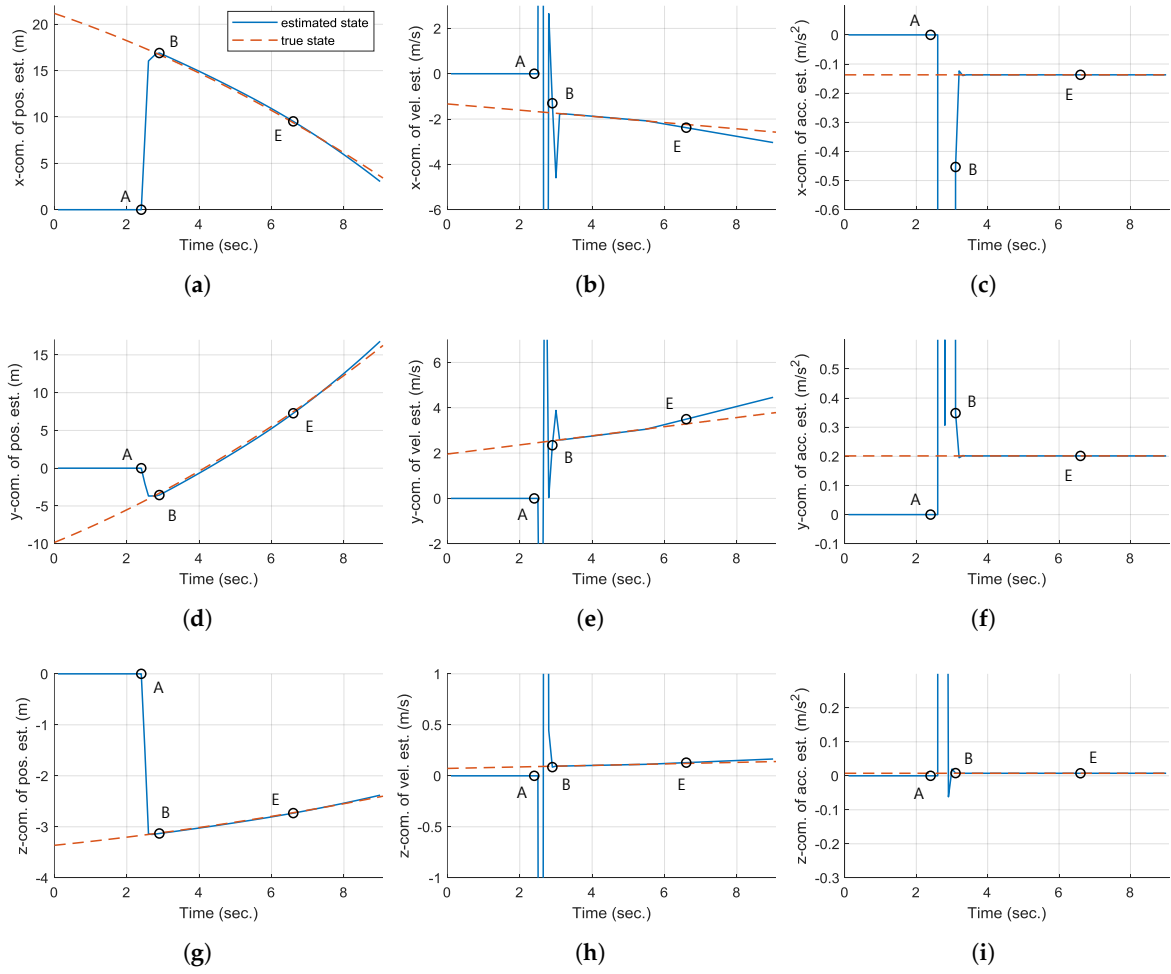
**Figure 9.** Trajectories of the hexacopter and the obstacle (azimuth:  $-60^\circ$ , elevation:  $30^\circ$ , Simulation II).

On the contrary, the UAV with the proposed algorithm successfully avoids the obstacle and reaches the minimum distance of 2.0733 m at point C ( $t = 4.01$  s) as shown in Figure 10c. This is a reasonable value, considering that  $d_{sm}$  is set to 2 m in Table 1. At point D ( $t = 4.1$  s), the hexacopter determines that the obstacle is no longer a threat, and switches to the goal point guidance. Note that there is a time lag between point C and point D due to the sampling time of the LiDAR,  $\Delta t_l$ .

At point E ( $t = 6.6$  s), the obstacle deviates from the sensing range of the LiDAR, as shown in Figure 10c, and  $\vec{r}_{o,c}^m$  cannot be obtained from this point. Then, the virtual measurement,  $\hat{y}_k^n$ , is calculated using Equation (22) to produce the measurement of Equation (21) for the Kalman filter. However, the deviation from the true state can be observed before point E, especially in Figures 11b,e. This is because of the incorrect calculation of  $\vec{r}_{o,c}^m$  with partial detection of the obstacle. Nevertheless, the goal point guidance is maintained. Finally, the UAV reaches the point F ( $t = 9.08$  s) as shown in Figure 9.



**Figure 10.** Time histories of UAV-related variables (Simulation II): (a) attitude, (b) body rate, (c) minimum distance between UAV and obstacle, (d) absolute velocity, and (e) velocity (body-fixed coordinate system).



**Figure 11.** Time histories of the state estimate (Simulation II): (a)  $x_{e,k}^n$ , (b)  $\dot{x}_{e,k}^n$ , (c)  $\ddot{x}_{e,k}^n$ , (d)  $y_{e,k}^n$ , (e)  $\dot{y}_{e,k}^n$ , (f)  $\ddot{y}_{e,k}^n$ , (g)  $z_{e,k}^n$ , (h)  $\dot{z}_{e,k}^n$ , and (i)  $\ddot{z}_{e,k}^n$ .

### 6.3. Simulation III: Monte Carlo Simulation

A Monte Carlo simulation is carried out to investigate the robustness of the proposed algorithm. Five-hundred different scenarios are generated, where the following properties of the obstacle are assigned using a continuous uniform distribution: impact point, initial position, radius, velocity, and acceleration. First, the impact point with the UAV is randomly chosen between the initial position of the hexacopter,  $[0 \ 0 \ -3]^T$ , and the goal point,  $[25 \ 0 \ -3]^T$ , as shown in Figure 12.

Let us define this impact point as  $\vec{r}_p^n \equiv [\ell_p \ 0 \ -3]^T$  where  $\ell_p \in [0, 25]$  is shown in Figure 12. Then, the time-to-go of the UAV to the point can be calculated as follows,

$$t_{go} = \frac{\ell_p}{v_{avg}} \tag{26}$$

where  $v_{avg}$  is the average speed of the hexacopter. Referring to Figures 7d and 10d,  $v_{avg}$  is roughly chosen as 3.5 m/s. Note that the generated cases with  $t_{go} \leq 2.0$  s are excluded, considering a reasonable initial gap between the UAV and the obstacle. Then, the ranges of the magnitude of the acceleration and the velocity are set to  $[0, 2]$  and  $[2, 6]$ , respectively. If these are denoted as  $a_o$  and  $v_o$ , the length between  $\vec{r}_p^n$  and the initial position of the obstacle can be computed as follows.

$$L_p = \frac{1}{2}a_o t_{go}^2 + v_o t_{go} \tag{27}$$



Now,  $\theta_p$  and  $\varphi_p$  in Figure 12 are randomly chosen from  $[0, \pi]$  and  $[0, 2\pi]$ , respectively, which determine the impact angle. The initial position of the obstacle,  $\vec{r}_o^n$ , can be obtained as follows.

$$\vec{r}_o^n = \begin{bmatrix} L_p \sin \theta_p \cos \varphi_p \\ L_p \sin \theta_p \sin \varphi_p \\ L_p \cos \theta_p \end{bmatrix} + \vec{r}_p^n \tag{28}$$

Also, the acceleration,  $\vec{a}_o^n$ , and the velocity,  $\vec{v}_o^n$ , of the obstacle can be computed as follows.

$$\vec{a}_o^n = \begin{bmatrix} -a_o \sin \theta_p \cos \varphi_p \\ -a_o \sin \theta_p \sin \varphi_p \\ -a_o \cos \theta_p \end{bmatrix}, \quad \vec{v}_o^n = \begin{bmatrix} -v_o \sin \theta_p \cos \varphi_p \\ -v_o \sin \theta_p \sin \varphi_p \\ -v_o \cos \theta_p \end{bmatrix} \tag{29}$$

Finally, the radius of the obstacle is chosen in the range of  $[1, 3]$ . Note that the cases with the obstacle approaching from the direction outside the FoV of the LiDAR are also excluded from the result.

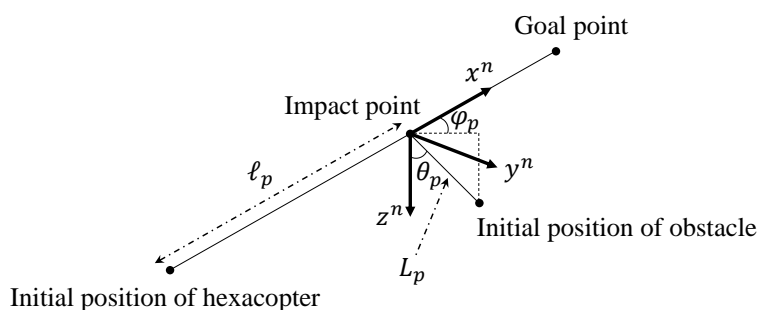


Figure 12. Generating obstacle properties for Monte Carlo simulation (Simulation III).

The Monte Carlo simulation result is shown in Figure 13, which is the time responses of the minimum distance between the hexacopter and the obstacle for each case. In every case, the hexacopter reaches the goal point without violating the safety requirement, as the minimum separation is greater than the added safety margin,  $d_{sm}$ , at all times. The Monte Carlo simulation result confirms the successful collision avoidance performance of the proposed algorithm.

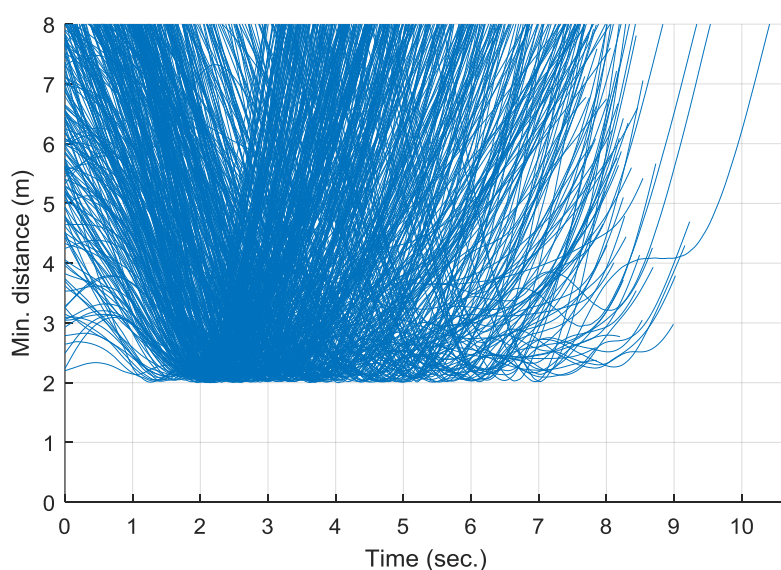


Figure 13. Time history of minimum distance between UAV and obstacle (Simulation III).

## 7. Conclusions

In this study, a collision avoidance algorithm of a hexacopter UAV is proposed. The algorithm is based on the collision cone approach, and is improved to handle a moving obstacle by considering the short-term predictions for the obstacle trajectory with the aid of an obstacle state estimator. A PD controller is used to generate angular velocities of the rotors where the command is fed from the proposed guidance algorithm. A LiDAR system on the UAV obtains the obstacle data points to form the spherical bounding box around the detected region of the obstacle and also the collision cone. Sometimes, the LiDAR system may lose the obstacle due to its limited sensing range and FoV. In this case, the Kalman filter produces a virtual measurement with the previous state estimate. A collision is detected by examining the predicted position of both the UAV and the obstacle. The obstacle state estimate also contributes to an avoidance maneuver of UAV by removing the points that are unreasonable to be taken as candidates of the aiming point. The numerical simulation results, including a Monte Carlo campaign, are shown to verify the proposed guidance algorithm. The results demonstrate that the hexacopter properly handles the moving obstacle; unlike the previous method that does not explicitly consider the predicted relative movement. For future research, the proposed algorithm will be extended to deal with multiple obstacles. An appropriate decision-making algorithm should be developed to assess the obstacles in terms of the level of threat.

**Author Contributions:** Conceptualization, J.P.; methodology, J.P. and N.C.; validation, N.C.; investigation, J.P. and N.C.; resources, J.P.; writing—original draft preparation, J.P.; writing—review and editing, N.C.; visualization, J.P. and N.C.; supervision, N.C.; funding acquisition, J.P. All authors have read and agreed to the published version of the manuscript.

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

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Omari, S.; Hua, M.; Ducard, G.; Hamel, T. Hardware and Software Architecture for Nonlinear Control of Multirotor Helicopters. *IEEE/ASME Trans. Mech.* **2013**, *18*, 1724–1736. [[CrossRef](#)]
2. Giribet, J.I.; Sanchez-Pena, R.S.; Ghersin, A.S. Analysis and Design of a Tilted Rotor Hexacopter for Fault Tolerance. *IEEE Trans. Aerosp. Electron. Syst.* **2016**, *52*, 1555–1567. [[CrossRef](#)]
3. Lee, H.; Kim, H.; Kim, H.J. Planning and Control for Collision-Free Cooperative Aerial Transportation. *IEEE Trans. Autom. Sci. Eng.* **2018**, *15*, 189–201. [[CrossRef](#)]
4. Zhang, J.; Gu, D.; Deng, C.; Wen, B. Robust and Adaptive Backstepping Control for Hexacopter UAVs. *IEEE Access* **2019**, *7*, 163502–163514. [[CrossRef](#)]
5. Chen, Y.; Zhang, G.; Zhuang, Y.; Hu, H. Autonomous Flight Control for Multi-Rotor UAVs Flying at Low Altitude. *IEEE Access* **2019**, *7*, 42614–42625. [[CrossRef](#)]
6. Li, Z.; Tan, J.; Liu, H. Rigorous Bore-sight Self-Calibration of Mobile and UAV LiDAR Scanning Systems by Strip Adjustment. *Remote Sens.* **2019**, *11*, 442. [[CrossRef](#)]
7. Lei, L.; Qiu, C.; Li, Z.; Han, D.; Han, L.; Zhu, Y.; Wu, J.; Xu, B.; Feng, H.; Yang, H.; Yang, G. Effect of Leaf Occlusion on Leaf Area Index Inversion of Maize Using UAV–LiDAR Data. *Remote Sens.* **2019**, *11*, 1067. [[CrossRef](#)]
8. Yang, Q.; Su, Y.; Jin, S.; Kelly, M.; Hu, T.; Ma, Q.; Li, Y.; Song, S.; Zhang, J.; Xu, G.; Wei, J.; Guo, Q. The Influence of Vegetation Characteristics on Individual Tree Segmentation Methods with Airborne LiDAR Data. *Remote Sens.* **2019**, *11*, 2880. [[CrossRef](#)]
9. Lin, Y.C.; Cheng, Y.T.; Zhou, T.; Ravi, R.; Hasheminasab, S.M.; Flatt, J.E.; Troy, C.; Habib, A. Evaluation of UAV LiDAR for Mapping Coastal Environments. *Remote Sens.* **2019**, *11*, 2893. [[CrossRef](#)]
10. Yan, W.; Guan, H.; Cao, L.; Yu, Y.; Li, C.; Lu, J. A Self-Adaptive Mean Shift Tree-Segmentation Method Using UAV LiDAR Data. *Remote Sens.* **2020**, *12*, 515. [[CrossRef](#)]
11. Wallace, L.; Lucieer, A.; Watson, C.; Turner, D. Development of a UAV-LiDAR System with Application to Forest Inventory. *Remote Sens.* **2012**, *4*, 1519–1543. [[CrossRef](#)]

12. Wallace, L.; Lucieer, A.; Watson, C.S. Evaluating Tree Detection and Segmentation Routines on Very High Resolution UAV LiDAR Data. *IEEE Trans. Geosci. Remote Sens.* **2014**, *52*, 7619–7628. [[CrossRef](#)]
13. Chen, N.; Ni, N.; Kapp, P.; Chen, J.; Xiao, A.; Li, H. Structural Analysis of the Hero Range in the Qaidam Basin, Northwestern China, Using Integrated UAV, Terrestrial LiDAR, Landsat 8, and 3-D Seismic Data. *IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens.* **2015**, *8*, 4581–4591. [[CrossRef](#)]
14. Hening, S.; Ippolito, C.; Krishnakumar, K. 3D LiDAR SLAM Integration with GPS/INS for UAVs in Urban GPS-Degraded Environments. In Proceedings of the AIAA Infotech@Aerospace Conference, Grapevine, TX, USA, 9–13 January 2017.
15. Zheng, L.; Zhang, P.; Tan, J.; Li, F. The Obstacle Detection Method of UAV Based on 2D Lidar. *IEEE Access* **2019**, *7*, 163437–163448. [[CrossRef](#)]
16. Cherubini, A.; Spindler, F.; Chaumette, F. Autonomous Visual Navigation and Laser-Based Moving Obstacle Avoidance. *IEEE Trans. Intell. Transp. Syst.* **2014**, *15*, 2101–2110. [[CrossRef](#)]
17. Ji, J.; Khajepour, A.; Melek, W.W.; Huang, Y. Path Planning and Tracking for Vehicle Collision Avoidance Based on Model Predictive Control With Multiconstraints. *IEEE Trans. Veh. Technol.* **2017**, *66*, 952–964. [[CrossRef](#)]
18. Malone, N.; Chiang, H.; Lesser, K.; Oishi, M.; Tapia, L. Hybrid Dynamic Moving Obstacle Avoidance Using a Stochastic Reachable Set-Based Potential Field. *IEEE Trans. Robot.* **2017**, *33*, 1124–1138. [[CrossRef](#)]
19. Li, S.; Li, Z.; Yu, Z.; Zhang, B.; Zhang, N. Dynamic Trajectory Planning and Tracking for Autonomous Vehicle With Obstacle Avoidance Based on Model Predictive Control. *IEEE Access* **2019**, *7*, 132074–132086. [[CrossRef](#)]
20. Zhang, H.; Jin, H.; Liu, Z.; Liu, Y.; Zhu, Y.; Zhao, J. Real-Time Kinematic Control for Redundant Manipulators in a Time-Varying Environment: Multiple-Dynamic Obstacle Avoidance and Fast Tracking of a Moving Object. *IEEE Trans. Ind. Inform.* **2020**, *16*, 28–41. [[CrossRef](#)]
21. Mujumdar, A.; Padhi, R. Reactive Collision Avoidance Using Nonlinear Geometric and Differential Geometric Guidance. *J. Guid. Control Dyn.* **2011**, *34*, 303–310. [[CrossRef](#)]
22. Wang, J.; Xin, M. Integrated Optimal Formation Control of Multiple Unmanned Aerial Vehicles. *IEEE Trans. Control Syst. Technol.* **2013**, *21*, 1731–1744. [[CrossRef](#)]
23. Yang, X.; Alvarez, L.M.; Bruggemann, T. A 3D Collision Avoidance Strategy for UAVs in a Non-Cooperative Environment. *J. Intell. Robot. Syst.* **2013**, *70*, 315–327. [[CrossRef](#)]
24. Gageik, N.; Benz, P.; Montenegro, S. Obstacle Detection and Collision Avoidance for a UAV With Complementary Low-Cost Sensors. *IEEE Access* **2015**, *3*, 599–609. [[CrossRef](#)]
25. Radmanesh, M.; Kumar, M. Flight Formation of UAVs in Presence of Moving Obstacles Using Fast-Dynamic Mixed Integer Linear Programming. *Aerosp. Sci. Technol.* **2016**, *50*, 149–160. [[CrossRef](#)]
26. Lin, Y.; Saripalli, S. Sampling-Based Path Planning for UAV Collision Avoidance. *IEEE Trans. Intell. Transp. Syst.* **2017**, *18*, 3179–3192. [[CrossRef](#)]
27. Santos, M.C.P.; Rosales, C.D.; Sarcinelli-Filho, M.; Carelli, R. A Novel Null-Space-Based UAV Trajectory Tracking Controller With Collision Avoidance. *IEEE/ASME Trans. Mech.* **2017**, *22*, 2543–2553. [[CrossRef](#)]
28. Marchidan, A.; Bakolas, E. Collision Avoidance for an Unmanned Aerial Vehicle in the Presence of Static and Moving Obstacles. *J. Guid. Control Dyn.* **2020**, *43*, 96–110. [[CrossRef](#)]
29. Al-Hourani, A.; Kandeepan, S.; Lardner, S. Optimal LAP Altitude for Maximum Coverage. *IEEE Wirel. Commun. Lett.* **2014**, *3*, 569–572. [[CrossRef](#)]
30. Mozaffari, M.; Saad, W.; Bennis, M.; Debbah, M. Efficient Deployment of Multiple Unmanned Aerial Vehicles for Optimal Wireless Coverage. *IEEE Commun. Lett.* **2016**, *20*, 1647–1650. [[CrossRef](#)]
31. Mozaffari, M.; Saad, W.; Bennis, M.; Debbah, M. Wireless Communication Using Unmanned Aerial Vehicles (UAVs): Optimal Transport Theory for Hover Time Optimization. *IEEE Trans. Wirel. Commun.* **2017**, *16*, 8052–8066. [[CrossRef](#)]
32. Saxena, V.; Jaldén, J.; Klessig, H. Optimal UAV Base Station Trajectories Using Flow-Level Models for Reinforcement Learning. *IEEE Trans. Cogn. Commun. Netw.* **2019**, *4*, 1101–1112. [[CrossRef](#)]
33. Chakravarthy, A.; Ghose, D. Obstacle Avoidance in a Dynamic Environment: A Collision Cone Approach. *IEEE Trans. Syst. Man Cybern. Part A Syst. Hum.* **1998**, *28*, 562–574. [[CrossRef](#)]
34. Chakravarthy, A.; Ghose, D. Generalization of the Collision Cone Approach for Motion Safety in 3-D Environments. *Autonomous Robots* **2012**, *32*, 243–266. [[CrossRef](#)]
35. Stevens, B.L.; Lewis, F.L. *Aircraft Control and Simulation*; Wiley: Hoboken, NJ, USA, 2003; pp. 1–481.

36. Simon, D. *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*; John Wiley & Sons: Hoboken, NJ, USA, 2006; pp. 123–129.
37. Park, J.; Kim, Y. Collision Avoidance for Quadrotor Using Stereo Vision Depth Maps. *IEEE Trans. Aerosp. Electron. Syst.* **2015**, *51*, 3226–3241. [[CrossRef](#)]



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).