Unmanned Systems, Vol. 1, No. 2 (2013) 259–275

DOI: 10.1142/S2301385013500076

UNMANNED SYSTEMS
http://www.worldscientific.com/worldscinet/us

# Dynamic Inversion-Based Nonlinear Aiming Point Guidance of Unmanned Aerial Vehicles for Reactive Obstacle Avoidance

Ramsingh G. Raja, Charu Chawla, Radhakant Padhi*

*Department of Aerospace Engineering Indian Institute of Science, Bangalore, 560012, India*



A dynamic inversion-based three-dimensional nonlinear aiming point guidance law is presented in this paper for reactive collision avoidance of unmanned aerial vehicles. When an obstacle is detected in the close vicinity and collision is predicted, an artificial safety sphere is put around the center of the obstacle. Next, the velocity vector of the vehicle is realigned towards an 'aiming point' on the surface of the sphere in such a way that passing through it can guarantee safe avoidance of the obstacle. The guidance command generation is based on angular correction between the actual and the desired direction of the velocity vector. Note that the velocity vector gets aligned along the selected aiming point quickly (i.e., within a fraction of the available time-to-go), which makes it possible to avoid pop-up obstacles. The guidance algorithm has been verified with simulations carried out both for single obstacles as well as for multiple obstacles on the path and also with different safety sphere sizes around the obstacles. The proposed algorithm has been validated using both kinematic as well as point mass model of a prototype unmanned aerial vehicle. For better confidence, results have also been validated by incorporating a first-order autopilot models for the velocity vector magnitude and directions.

*Keywords*: Dynamic inversion; aiming point guidance; obstacle avoidance; collision avoidance; unmanned aerial vehicles; UAV guidance.

US

## Nomenclature

$X_{ap}$ : Aiming Point (m, m, m)
$t_{go}$ : Time-to-go (s)
$X_{OB}$ : Position of the obstacle in local inertial frame (m, m, m)
$X_v$ : Position of the UAV in local inertial frame (m, m, m)
$\|X_r\|$ : Relative distance between obstacle and UAV (m)
$\eta$ : Plane containing the collision circle
$C^{\eta}$ : Collision Circle on obstacle safety sphere
$\hat{n}$ : Normal vector to the collision plane
$V_T$ : Total velocity of UAV (m/s)
$X_C$ : Center of the collision circle (m, m, m)
$r_s$ : Radius of the safety sphere (m)
$r_c$ : Radius of the collision circle (m)
$r_t$ : Tangential vector along the aiming point on the safety sphere (m, m, m)
$X_{t^*}$ : Intersection point on collision plane $\eta$ (m, m, m)
$X_{vap}$ : Relative aiming point vector (m, m, m)
$X_{goal}$ : Goal Point (m, m, m)
$X_{in}$ : Initial postion of the UAV (m, m, m)
$t_0$ : Initial time (s)
$V_{initial}$ : Initial velocity of the UAV (m/s)
$t^*$ : Time to reach the collision plane (s)
$\chi$ : Course angle (deg)
$\gamma$ : Flight path angle (deg)
$\gamma^c$ : Commanded flight path angle (deg)
$\chi^c$ : Commanded course angle (deg)
$T_{max}$ : Maximum thrust force (N)
$L$ : Lift force (N)
$D$ : Drag force (N)
$\bar{Q}$ : Dynamic Pressure (Pa)
$S$ : Surface area ($m^2$)
$C_L$ : Lift coefficient
$C_D$ : Drag coefficient
$g$ : Acceleration due to gravity ($m/s^2$)
$\rho$ : Atmospheric density ($Kg/m^3$)

Received 18 September 2012; Revised 10 April 2013; Accepted 2 June 2013; Published 10 October 2013. This paper was recommended for publication in its revised form by editorial board member, Kemao Pang.
Email Address: *padhi@aero.iisc.ernet.in

$C_X, C_Z$ : Axial coefficients in body axes frame
$k_\nu, k_\chi, k_{V_T}$ : Gain values of the guidance loop
$k_\mu, k_\alpha, k_T$ : Gain values of the autopilot loop
$W$ : Point on the collision circle
$X_{\text{viap}}$ : Variable Aiming point $(m, m, m)$
$L_{W/B}$ : Transformation matrix from body to wind axes
$\beta$ : Side slip angle (deg)
$C_{X_0}, C_{Z_0}$ : Aerodynamic coefficients
$C_{X_{a1}}, C_{X_{a2}}$ and $C_{Z_{a1}}$ : Aerodynamic derivatives
$\alpha_d$ : Commanded angle of attack (deg)
$\mu_d$ : Commanded Bank angle (deg)
$T_d$ : Commanded Thrust (N)
$R$ : Relative distance beween UAV and the aiming point (m)
$X_a$ : Center of the composite safety sphere in local inertial frame (m, m, m)
$X_{\text{OG}}$ : Center of the global obstacle in local inertial frame (m, m, m)
$X_{\text{New}}$ : New set of individual composite safety sphere centers
$r_{\text{NEW}}$ : New set of individual composite safety sphere radius
$r_a$ : Radius of the composite safety sphere (m)
$r_g$ : Radius of the global obstacle safety sphere (m)
$t_{\text{min}}$ : Time to reach the closest obstacle collision plane (s)

## Acronym

UAV : Unmanned Aerial Vehicle
CG : Center of Gravity
LOS : Line of Sight

## Subscript

max : Maximum
min : Minimum
*i, j, k, p* : Iteration variables

## 1. Introduction

Rapid advancement of various technologies associated with unmanned aerial vehicles (UAVs) has taken place over the last two decades. As a result, UAVs have been deployed in numerous fields such as reconnaissance and surveillance, targeted attacks with less collateral damage, battle damage assessment, traffic monitoring for crime prevention, detection and containment of hazardous leakages in industries, assessment and rehabilitation in case of natural calamities, etc. For various reasons, many UAV missions can be practically more useful in case they can fly autonomously at low altitudes, including the fact that high resolution pictures are possible only from a close distance. However, flying at low altitude may require the UAVs to fly in the vicinity of obstacles such as buildings, trees, etc., making them vulnerable for collisions. Moreover, since information related to all possible collisions cannot be accounted *a priori*, it is vital that UAVs have a good built-in mechanism and associated guidance laws to avoid collisions in their flight path.

The collision avoidance problem addressed in this paper can in fact be embedded into the problem of 'path planning', which is the process of finding a safe flight path to the goal point. It typically consists of two layers (i) a global path planner and (ii) a local path planner. The global path planner finds a path beforehand such that all known obstacles are avoided and the destination is reached. For local path planning, however, the main aim is to avoid collisions with obstacles at close vicinity (especially with the pop-up threats), which is done in the following manner. When an onboard sensor detects obstacle(s) and predicts a possible collision, the guidance law attempts to maneuver the vehicle away from the danger as soon as possible so that the impending collision is averted. Such a guidance is also called as 'reactive guidance', since the available reaction time is very small and decisions have to be taken quickly (i.e., within a fraction of the available time-to-go). Note that reactive collision avoidance guidance typically results in high maneuvers for a small duration of time. Moreover, the guidance law should also ensure that while avoiding obstacles the vehicle should not deviate too much from its original intended path. This is both because other obstacles should not come on its new flight path and it should not deviate too much away from seeking its intended destination (otherwise, putting it back into track becomes difficult). Another critical restriction on such a reactive guidance law is that it should be computationally quite efficient (preferably available in closed form), since the time to react is typically quite low and, if possible, the control and guidance update needs to be done at a higher rate. Moreover, for fixed-wing UAVs, it should also assure that the vehicle keeps moving at sufficiently higher velocity to avoid stalling. In view of these, there is a need for developing a reactive guidance scheme based on sound geometric and mathematical considerations. Preferably it should also have sufficient generality so that it can be applied for a wide range of applications.

Even though they are fundamentally different, a quick review of literature shows that the problem of reactive collision avoidance has been substantially influenced by global path planning algorithms. The artificial potential field method[1] is a popular approach which is tailored for obstacle avoidance such that obstacles have a repulsive field while the destination has an attractive field. The resultant field represents a safe direction for the UAV to move along. However, this algorithm is not strictly reactive, since at

every instant it takes into account the presence of all (or most of) the obstacles in the environment, as well as the destination, before deciding the direction. It usually relies on numerical optimization techniques and hence can get trapped in local minimum and computational issues. Another promising algorithm in collision avoidance and global path planning is the philosophy of rapidly exploring random tree (RRT),[2] which has also been used for reactive collision avoidance. However, the probabilistic nature of RRT does not guarantee the finding of a feasible path within a limited finite time. Algorithms such as proportional navigation (PN) guidance[4] and its variants, which are used heavily in missile guidance, have been attempted for collision avoidance guidance as well. The problem of UAV pursuing its goal has been implemented with intermediate obstacles in the path using PN guidance-based collision avoidance scheme.[5] However, this scheme leads to a very high control effort every time a new target is pursued. Instead, an interesting minimum effort guidance (MEG) approach is proposed in,[6] which minimizes the control effort for the entire trajectory along with avoiding collisions for multiple targets. A collision cone approach[7] is used to detect potential collisions by considering a safety sphere around the obstacle in MEG guidance. The MEG guidance causes the vehicle to maneuver until the point of impact, which is risky as the vehicle may enter the safety sphere. These and many other techniques have also been discussed on global and local path planning in a review paper published recently by one of the authors of this paper with a different co-worker.[3]

The concept and algorithm proposed in this paper is mainly focused on the guidance of UAVs for the reactive collision avoidance. There are a few assumptions, which can be summarized as (i) the information regarding global path, avoiding all known obstacles, is known *a priori* and the task is to avoid only pop-up obstacles before joining back the intended path, (ii) without loss of generality, all obstacles are assumed to be stationary (else, their position in a future point of time needs to be predicted depending on their velocity), (iii) the position information and safety sphere radius (depending on the size of the obstacle) are available through some onboard sensor system, like a vision sensor system for example (iv) a point mass model fairly describes the motion of the UAV for the guidance purpose and (v) all turnings are done by assuming turn co-ordination, i.e., by maintaining sideslip angle at zero (which can in turn be assured by appropriate design of an inner loop autopilot). Note that the details of image processing algorithm for detecting the position of the obstacle(s) is not within the scope of this paper. However, an interested reader can refer to one of the recent works of the third author of this paper along with his co-worker,[18] where the issue is addressed by assuming a simple pin-hole passive camera in the loop and then processing the information through an extended Kalman filter.

In this paper, a reactive collision-avoidance algorithm is proposed, in which UAV invokes the collision-avoidance algorithm only when an obstacle is encountered. The collision avoidance occurs in two steps; first, the collision is detected based on the collision cone approach.[6,7] Next, if there is a threat, a collision avoidance maneuver is performed with respect to an 'aiming point', judiciously selected on the surface of the safety sphere. Note that the concept can be interpreted in the light of 'pursuit guidance', where the objective is to reorient the velocity vector of the UAV to the line-of-sight.[8] In all the cases, the gains are selected such that the obstacles are avoided quickly (i.e., within a fraction of the available time-to-go).

For validation purpose, the guidance commands have been applied to two mathematical models; i.e., the 'kinematic model' as well as the 'point mass model' of a prototype UAV. Simulations with different scenarios with varied safety sphere sizes and different number of obstacles are demonstrated. For all the cases, the UAV avoids the obstacle and reaches the goal point. Results are also validated by incorporating first-order autopilots for the guidance commands in case of point mass dynamics of UAV. Note that a similar concept inspired by the aiming point guidance[9] is also demonstrated in[10] from a different, but related perspective. However, unlike,[10] here the body axis information and some additional restrictive assumptions (like holonomic motion capability of the UAV) are not needed to generate the guidance command. Instead, the guidance command is generated in the velocity and inertial frames accounting for the realistic system dynamics (i.e., nonholonomic motion constraints), which is a natural way of guiding fixed-wing flight vehicles.

## 2. System Dynamics of UAV Motion

In this paper, two standard system dynamics for the vehicle movement have been considered. First, a simplistic "kinematic model" has been considered with autopilot lags in all channels of the state equation for quick validation of the basic philosophy. Subsequently, to be more realistic, the proposed guidance strategy has been reworked using a 'point mass model' of the vehicle and further algebra has been carried out by introducing an additional inner loop to generate the physically meaningful guidance parameters, namely the desired bank angle, angle of attack and the thrust required for the vehicle, as the necessary guidance parameters. Note that to be even more realistic, autopilot lags for these quantities have been assumed as well while validating our proposed guidance strategy, which will be discussed later. In this section, the relevant details of the

mathematical models of both kinematic as well as point mass models used in this research are given in fair detail.

### 2.1. *Kinematic model*

The relevant set of equations in this model are given by

$$\begin{aligned} \dot{x} &= V_T \cos\chi \cos\gamma, \\ \dot{y} &= V_T \sin\chi \cos\gamma, \\ \dot{h} &= V_T \sin\gamma, \\ \dot{V}_T &= k_v(V_T^c - V_T), \\ \dot{\gamma} &= k_\gamma(\gamma^c - \gamma), \\ \dot{\chi} &= k_\chi(\chi^c - \chi), \end{aligned} \tag{1}$$

where $V_T$ is the velocity of the UAV whose directions are given by two angles, namely the flight path angle $\gamma$ and course angle $\chi$. The first three equations of Eq. (1) represent the movement of the center of mass of the vehicle in an inertial frame depending on the velocity vector magnitude and its orientation. The next three equations essentially represent first-order lags, which in turn govern the evolution of $V_T$, $\gamma$ and $\chi$, based on their commanded values $V_T^c$, $\gamma^c$ and $\chi^c$, respectively. The key idea here is to generate $\gamma^c$ and $\chi^c$ from an appropriate guidance formulation, as well as giving an appropriate $V_T^c$ command, so that the obstacles can be avoided.

### 2.2. *Point mass model*

In this case, assuming co-ordinated turn, the model representing vehicle dynamics[12] is given by

$$\begin{aligned} \dot{x} &= V_T \cos\chi \cos\gamma, \\ \dot{y} &= V_T \sin\chi \cos\gamma, \\ \dot{h} &= V_T \sin\gamma, \\ \dot{V}_T &= \frac{1}{m}(T\cos\alpha - D - mg\sin\gamma), \\ \dot{\gamma} &= \frac{1}{mV_T}[(T\sin\alpha + L)\cos\mu - mg\cos\gamma], \\ \dot{\chi} &= \frac{1}{mV_T\cos\gamma}(T\sin\alpha + L)\sin\mu. \end{aligned} \tag{2}$$

One can notice that the first three equations of Eq. (2) are exactly same as those of the first three equations of Eq. (1). However, the terms $\dot{V}_T$, $\dot{\gamma}$ and $\dot{\chi}$ are more realistic as the aerodynamic forces (namely lift and drag), gravitational force (i.e., weight) and thrust forces acting on the vehicle are explicitly accounted for. As a compromise, however, one now needs a set of compatible system parameter values representing a real system. Here the parameter values of a prototype UAV, named as All Electric airplane-2 (AE-2), have been used (see Fig. 1 for a picture of the vehicle). Note that AE-2 UAV was designed, fabricated and wind-tunnel tested at the UAV lab of Indian Institute of Science, Bangalore.[13]

Fig. 1. The AE-2 Unmanned Aerial Vehicle.

This UAV has a reference area of $S = 0.6\,\mathrm{m}^2$ and it has mass $m = 6\,\mathrm{kg}$. Maximum value of thrust $(T_{\max})$ which can be produced by its electric motor and propeller assembly is 15 N. It is assumed that thrust produced has linear relation with the throttle input. Forces $L$ and $D$ acting on the vehicle are given as $L = \bar{Q}SC_L$ and $D = \bar{Q}SC_D$, where $\bar{Q} = (1/2)\rho V_T^2$, $C_L$ and $C_D$ are the dynamic pressure, lift coefficient and drag coefficient, respectively. However, note that the lift and drag coefficients of this vehicle (as found from wind-tunnel experimental data, followed by extensive curve-fitting) is given in the vehicle's body axes system. However, lift and drag typically act in the velocity (wind) axis frame. Hence, these data need be converted to the velocity axis frame, which is done using the following relationship.

$$\begin{bmatrix} C_D \\ C_L \end{bmatrix} = L_{W/B}\begin{bmatrix} -C_X \\ -C_Z \end{bmatrix} = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix}\begin{bmatrix} -C_X \\ -C_Z \end{bmatrix}. \tag{3}$$

Note that Eq. (3) assumes that the side-slip angle is zero. This is very close to reality with the assumption of co-ordinated turn, which is ensured in the formulation while generating the necessary bank angle as the set of equations in Eq. (2) are valid only under this assumption. Using Eq. (3) and using the expanded expressions for $C_X$ and $C_Z$, drag and lift forces can be expressed as follows

$$\begin{bmatrix} D \\ L \end{bmatrix} = \begin{bmatrix} \bar{Q}S[(C_{X_0} + C_{X_{a1}}\alpha + C_{X_{a2}}\alpha^2)\cos\alpha \\ \quad + (C_{Z_0} + C_{Z_{a1}}\alpha)\sin\alpha] \\ \bar{Q}S[-(C_{X_0} + C_{X_{a1}}\alpha + C_{X_{a2}}\alpha^2)\sin\alpha \\ \quad + (C_{Z_0} + C_{Z_{a1}}\alpha)\cos\alpha] \end{bmatrix}, \tag{4}$$

Table 1. Aerodynamic coefficients.

| $C_{X_0}$ | $C_{Z_0}$ | $C_{Z_{a1}}$ | $C_{X\alpha_1}$ | $C_{X\alpha_2}$ |
|---|---|---|---|---|
| 0.0386 | 0.1653 | 0.087138 | −0.0040376 | −0.0010525 |

where $C_{X_0}$, $C_{Z_0}$ are the base coefficients and $C_{X_{a1}}$, $C_{X_{a2}}$ and $C_{Z_{a1}}$ are the angle of attack-dependent derivatives of the aerodynamic coefficients. An interested reader can see[14] for a detail model of the vehicle with all necessary coefficient values (in the sense of six degree-of-freedom modeling). The relevant values that are necessary for the work in this paper have been tabulated in Table 1.[13]

Note that the ultimate aim here is to find the desired values of angle of attack $\alpha$, bank angle $\mu$ and the desired thrust $T$ that will ensure collision avoidance.

## 3. Reactive Collision Avoidance Guidance

In the present work, it is assumed that global path planning has already been done off line (or possibly loaded to the onboard processor through telemetry), where a path towards the goal point has already been found accounting for the known obstacles in the environment. It is also assumed that the UAV is equipped with an onboard camera with necessary image processing algorithms to sense an onforseen obstacle, along with its location, in its close vicinity. The main focus then is to predict a possible collision with the obstacle, and if so, to steer it away with appropriate generation of guidance commands. Details of this guidance algorithm are discussed in this section. Without loss of generality, it is also assumed that the pop-up obstacles are stationary (else, a prediction logic of its movement should also come into picture, which can be considered as a future scope of research).

### 3.1. *Aiming point computation and guidance command generation*

Once the obstacle information is available through onboard sensors, the task is to predict possible collision, and if so, to steer it away by appropriately computing of aiming point and generation of required guidance commands. The philosophy behind computing the aiming point is based on collision cone approach has physical relevance with torch light experiment, where if the torch light is glown to the sphere, the locus of extreme points of the glow area will form a circle on the sphere and the rays passing through the extreme points are tangent to the sphere. The same analogy is replicated in 3D representation of collision cone containing UAV's CG at coordinate frame origin, point obstacle with safety sphere, plane and a circle as shown in Fig. 2. Note that if one draws tangents to this sphere from the CG of the UAV, it results in a 'cone' (hence it is called as the 'collision cone'[6]), and the circle formed by locus of tangents contact point on sphere is named as collision circle and represented as $C^{\eta}$.

The coordinate frame assumed in Fig. 2 is parallel to the intertial frame and located on the center of gravity (CG) of the UAV. Let the UAV fly along the direction of the $V_T$. It is also assumed that the angle of attack and sideslip angles are small so that vehicle body $X$-axis (along which the onboard sensor is installed) can detect obstacles along the $V_T$-direction. Once an obstacle is detected by onboard sensors, collision avoidance algorithm is invoked and a new aiming point is computed. As per collision cone approach the computed aiming point is tangent to the sphere, so it should lie on this collision circle. The center and radius of the collision circle is determined by using the information of obstacle sphere radius and distance between UAV and center of the obstacle. From Fig. 2, let us consider the $\Delta X_V W X_{\mathrm{OB}}$, where, $X_V$ is the UAV position, $W$ is a point on the collision circle and $X_{\mathrm{OB}}$ is the obstacle position. Then by applying Pythagoras theorem on $\Delta X_V W X_{\mathrm{OB}}$

$$\|X_r\|^2 = \|r_t\|^2 + r_s^2, \tag{5}$$

$$\|r_t\|^2 = \|X_r\|^2 - r_s^2, \tag{6}$$

$$\|X_r\| = m_1 + m_2, \tag{7}$$

where $r_s$ is the radius of the safety sphere, $r_t$ is the tangent vector joining UAV position $X_V$ and $W$, the line-of-sight (LOS) $X_r$, by definition, is the line joining the CG of the UAV to the center of the obstacle. $X_C$ is the center of the collision circle and the Euclidean distance between this center to $X_V$ is $m_1$ and to $X_{\mathrm{OB}}$ is $m_2$, respectively. Here $m_1$ and $m_2$ are also the components of the line segment $\|X_r\|$. The following mathematical expression shows that $m_1$ is associated with $\Delta X_V W X_C$ and $m_2$ is associated with $\Delta X_C W X_{\mathrm{OB}}$. Therefore, it is easy to establish a relationship between these two triangles. After carrying out simple algebra the expression for $m_1$ and eventually the expression for $m_2$ can be deduced.

Applying Pythagoras theorem on $\Delta X_V W X_C$

$$\begin{aligned} \|r_t\|^2 &= m_1^2 + r_c^2 \\ r_c^2 &= \|r_t\|^2 - m_1^2, \end{aligned} \tag{8}$$

where $r_c$ is the radius of the collision circle, from $\Delta X_C W X_{\mathrm{OB}}$

$$r_s^2 = m_2^2 + r_c^2, \tag{9}$$

$$\begin{aligned} m_2^2 &= r_s^2 - r_c^2 \\ &= r_s^2 - \|r_t\|^2 + m_1^2. \end{aligned} \tag{10}$$

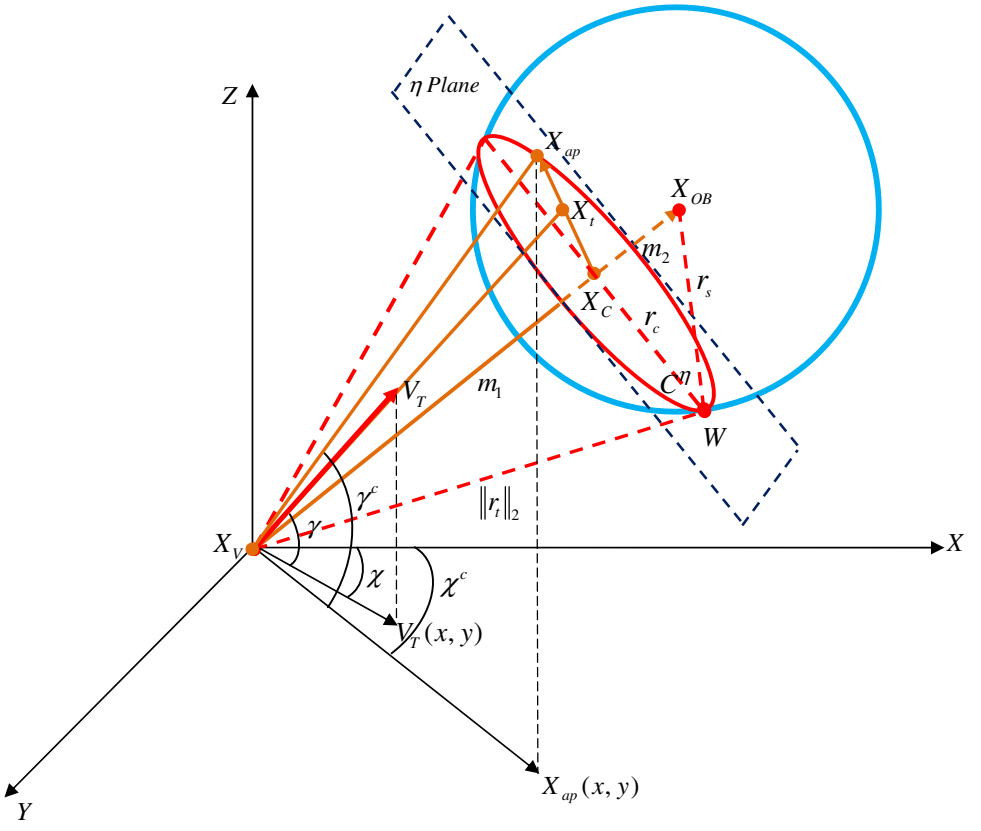



Fig. 2. 3D view of collision cone with UAV and obstacle.

To find the value of $m_1$ and $m_2$, squaring Eq. (7) on both sides and substituting it in Eq. (10), yields

$$\begin{aligned} \|X_r\|^2 &= (m_1+m_2)^2 = m_1^2+m_2^2+2m_1m_2 \\ &= m_1^2+r_s^2-\|r_t\|^2+m_1^2+2m_1(\|X_r\|-m_1) \\ &= r_s^2-\|r_t\|^2+2m_1\|X_r\|, \\ m_1 &= \frac{\|X_r\|^2-r_s^2+\|r_t\|^2}{2\|X_r\|}=\frac{\|r_t\|^2}{\|X_r\|}. \end{aligned} \tag{11}$$

The $m_2$ value can be find out by substituting the value of $m_1$ in Eq. (7)

$$\begin{aligned} m_2 &= \|X_r\|-m_1=\|X_r\|-\frac{\|r_t\|^2}{\|X_r\|}, \\ m_2 &= \frac{r_s^2}{\|X_r\|}. \end{aligned} \tag{12}$$

The radius of the collision circle can be obtained by substituting the value of $m_1$ in Eq. (8)

$$\begin{aligned} r_c &= \sqrt{\|r_t\|^2-\frac{\|r_t\|^4}{\|X_r\|^2}}, \\ r_c &= \frac{\|r_t\|r_s}{\|X_r\|}. \end{aligned} \tag{13}$$

The center of the collision circle is given as

$$X_C = X_V+\frac{m_1}{\|X_r\|}(X_{\mathrm{OB}}-X_V). \tag{14}$$

Let us consider the plane $\eta$, which contains the collision circle $C^\eta$. Due to geometrical symmetry, the vector joining collision circle center $(x_c,y_c,z_c)$ and obstacle center $(x_{ob}, y_{ob}, z_{ob})$ are always perpendicular to plane $\eta$. Therefore, the normal vector $\hat{n}$ to the plane can be described as

$$\hat{n} = (a,b,c) = (x_{ob}-x_c, y_{ob}-y_c, z_{ob}-z_c). \tag{15}$$

The general plane equation passing through a point $(x_c,y_c, z_c)$ is

$$a(x-x_c)+b(y-y_c)+c(z-z_c)=0. \tag{16}$$

To find the intersection point between the plane $\eta$ and velocity vector projected in time $t$ from the UAV's current position. The position vector of UAV expressed in terms of instantaneous velocity vector projected in time $t$ is expressed as

$$X_t = X_V+tV. \tag{17}$$

Substitute above equation into plane equation and solve for desired time $t^*$, which is termed as time to reach

the plane

$$a(x_v + t^*u - x_c) + b(y_v + t^*v - y_c) + c(z_v + t^*w - z_c) = 0, \tag{18}$$

$$t^* = -\frac{a(x_v - x_c) + b(y_v - y_c) + c(z_v - z_c)}{au + bv + cw}, \qquad t^* = -\frac{\hat{n} \cdot (X_V - X_C)}{\hat{n} \cdot V}. \tag{19}$$

The intersection point is given as

$$X_{t^*} = X_V + t^* V. \tag{20}$$

Once intersection point is computed, the following conflict conditions are evaluated

$$\begin{aligned} &(1)\ \|X_{t^*} - X_C\| < \text{Radius of the collision circle } C^\eta, \\ &(2)\ t^* > 0. \end{aligned} \tag{21}$$

If the above conditions are satisfied then the obstacle under consideration is said to be critical, then the new aiming point is computed as follows

$$X_{\text{ap}} = X_c + r_c \frac{X_{t^*} - X_C}{\|X_{t^*} - X_C\|}. \tag{22}$$

Note that, when no obstacle is critical, the goal point becomes the aiming point, i.e., $X_{\text{ap}} = X_{\text{goal}}$. One of the practical issues which may occur in the implementation of the collision cone approach is the violation of the safety sphere after aiming point is reached. Once the UAV passes an aiming point, it immediately looks to maneuver towards the next aiming point. This may result in a brief violation of the first obstacle's safety bound if the direction of the new aiming point lies in the blind zone of the safety sphere. Such a scenario is illustrated in Fig. 3.

In order to solve this problem, a sphere-tracking algorithm[10] is activated when $\|X_r\| < r_s$. The sphere tracking algorithm computes a new aiming point, called the virtual aiming point which is a point on the surface of the safety sphere. This is found by radially extending the original relative distance line $\|X_r\|$ until it meets the surface of the safety sphere at $X_{\text{viap}}$. UAV then aims for the virtual aiming point until $\|X_r\| \geq r_s$ before moving towards the next aiming point ($X_{\text{ap2}}$). Mathematical Implementation of sphere tracking algorithm is provided in following steps.

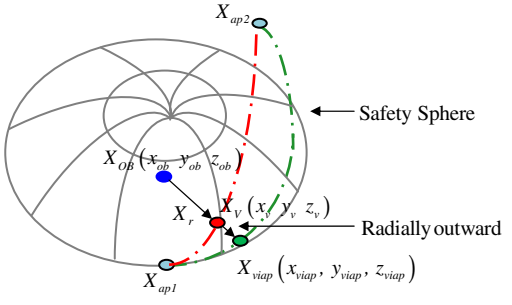



Fig. 3. Safety sphere violation after aiming point is reached.

The equation of the line passing through two points ($X_V$) and ($X_{\text{OB}}$) is given as

$$\frac{x - x_{ob}}{x_v - x_{ob}} = \frac{y - y_{ob}}{y_v - y_{ob}} = \frac{z - z_{ob}}{z_v - z_{ob}} = l. \tag{23}$$

The equation of the safety sphere is

$$(x - x_{ob})^2 + (y - y_{ob})^2 + (z - z_{ob})^2 = r_s^2. \tag{24}$$

Substituting Eq. (23) in Eq. (24) and solving for $l$ yields

$$l = \pm \frac{r_s}{\sqrt{(x_v - x_{ob})^2 + (y_v - y_{ob})^2 + (z_v - z_{ob})^2}}. \tag{25}$$

The virtual aiming point is computed as

$$\begin{aligned} x_{\text{viap}} &= l(x_v - x_{ob}) + x_{ob}, \\ y_{\text{viap}} &= l(y_v - y_{ob}) + y_{ob}, \\ z_{\text{viap}} &= l(z_v - z_{ob}) + z_{ob}. \end{aligned} \tag{26}$$

There are two solution sets for $X_{\text{viap}}$ depending on sign of $l$. The solution chosen is the point closer to the vehicle position.

Hence, the collision avoidance problem therefore can be interpreted as a sequential target interception problem or way point guidance problem with angle constraints at intermediate targets or way points. However, the basic difference is the online fixing of these intermediate points and then quickly realigning the velocity vector to visit them this is done by the computation of required guidance commands as follows.

Once the aiming point is computed, the task then is to follow the philosophy of 'aiming point guidance' of missile guidance literature[11] (which has a close resemblance to pursuit guidance[8]) and align the $V_T$ vector towards this aiming point. For carrying out the necessary algebra (see Fig. 2), the obstacle coordinates are given by $X_{\text{OB}} = [x_{ob} y_{ob} z_{ob}]$ and the relative aiming point vector is given by $X_{\text{vap}} = (X_{\text{ap}} - X_V)$. Once these co-ordinates are known (with respect to the coordiante frame at CG that is parallel to the intertial frame), the desired flight path angle $\gamma^c$ and course angle $\chi^c$ can be calculated as:

$$\gamma^c = \tan^{-1}(z_{\text{vap}}/\sqrt{x_{\text{vap}}^2 + y_{\text{vap}}^2}), \tag{27}$$

$$\chi^c = \tan^{-1}(y_{\text{vap}}/x_{\text{vap}}). \tag{28}$$

Note that even though it may sound logical to slow down (i.e., to reduce $V_T$) until the collision is avoided, in this work it is proposed to hold $V_T$ at its initial value at the start of guidance (i.e., $V_T^c = V_T(0)$). This is because the time availability is small and very less correction can be done for it within the available small time-to-go (usually thrust can be

varied only slowly). The angles $\gamma^c$ and $\chi^c$ as well as $V_T^c$ are the necessary guidance commands in the formulation using the *kinematic model*. In Fig. 2, $V_{T(x,y)}$ is the projection of the velocity vector and $X_{\text{vap}(x,y)}$ is the projection of the revised LOS in the horizontal plane, respectively. For the formulation using the *point mass model*, however, one needs to carry out further algebra to generate the physically meaningful guidance commands, i.e., the angle of attack, bank angle and thrust commands. Note that the alignment of velocity vector needs to be quickly carried out within a fraction of the available time-to-go, thereby making it possible to avoid pop-up obstacles. For this to happen, the information regarding time-to-go to the aiming point first need to be known, which is obtained in following manner. Note that $V_T$ needs to be aligned to the vector, which is located at the CG of the vehicle and points towards the aiming point. This vector can be interpreted as 'Revised LOS', the magnitude of which is given by

$$R = \|X_{\text{vap}}\|_2 = \sqrt{x_{\text{vap}}^2 + y_{\text{vap}}^2 + z_{\text{vap}}^2}. \tag{29}$$

Assuming that $V_T$ remains fairly constant (which is true, as our formulation attempts to assure it), the time-to-go to reach the aiming point can be computed as follows.

$$t_{\text{go}} = \frac{(X_{\text{vap}} \cdot V_T)}{\|V_T\|^2} = \frac{R}{V_T}. \tag{30}$$

Next, the settling times of the imposed error dynamics are selected as a fraction of this time-to-go and gain values are selected accordingly.

### 3.2. *Command generation using point mass model*

As pointed out in Sec. 3.1, generation of $\gamma^c$, $\chi^c$ and $V_T^c$ are sufficient as they serve as the necessary guidance commands. For the point mass model, however, it is necessary that the actual vehicle parameters $\gamma$, $\chi$ and $V_T$ track these commanded values as soon as possible, thereby generating the necessary physically meaningful guidance parameters for the vehicle, namely the desired angle of attack command $\alpha_d$, the desired bank angle command $\mu_d$ as well as the desired thrust command $T_d$.[17] Note that $\alpha$, $\mu$ and $T$ can be interpreted as 'control variables' in the point mass model and hence a control synthesis problem for tracking $\gamma^c$, $\chi^c$ and $V_T^c$ can generate $\alpha_d$, $\mu_d$ and $T_d$. This is done as follows.

Using the philosophy of dynamic inversion,[14,15] the tracking objective of $\gamma \to \gamma^c$, $\chi \to \chi^c$ and $V_T \to V_T^c$ can be ensured by enforcing the following set of first-order error dynamics:

$$(\dot{\gamma} - \dot{\gamma}^c) + k_\gamma(\gamma - \gamma^c) = 0, \tag{31}$$

$$(\dot{\chi} - \dot{\chi}^c) + k_\chi(\chi - \chi^c) = 0, \tag{32}$$

$$(\dot{V}_T - \dot{V}_T^c) + k_{V_T}(V_T - V_T^c) = 0, \tag{33}$$

where $k_{V_T} > 0$, $k_\gamma > 0$ and $k_\chi > 0$ are the gains (tuning parameters) that need to be selected carefully. Next, assuming quasi-steady approximation at every grid point of time for $\gamma^c$, $\chi^c$ and $V_T^c$ (i.e., $\dot{\gamma}^c = \dot{\chi}^c = \dot{V}_T^c = 0$), using the expressions for $\dot{V}_T$, $\dot{\gamma}$ and $\dot{\chi}$ from Eq. (2) and carrying out the necessary algebra using Eqs. (31)–(33), it can be shown that

$$\mu = \tan^{-1}\left(\frac{mV_T \cos\gamma(-k_\chi(\chi - \chi^c))}{mV_T(-k_\gamma(\gamma - \gamma^c)) + mg\cos\gamma}\right), \tag{34}$$

$$T\sin\alpha + L = \sqrt{\begin{array}{l}(mV_T\cos\gamma(-k_\chi(\chi - \chi^c)))^2 \\ \quad + (mV_T(-k_\gamma(\gamma - \gamma^c)) + mg\cos\gamma)^2\end{array}}, \tag{35}$$

$$T\cos\alpha - D = m(-k_V(V_T - V_T^c)) + mg\sin\gamma. \tag{36}$$

The expression for $\mu$ in Eq. (34) serves as the desired bank angle command $\mu_d$. However, it can be seen that Eqs. (35) and (36) are nonlinear and have two unknowns $T$ and $\alpha$. Hence, the desired angle of attack and thrust commands $\alpha_d$ and $T_d$ are generated by solving Eqs. (35) and (36) numerically using the standard Newton–Raphson method.[16] Note that Newton–Raphson method has 'quadratic convergence' and usually converges fast. With the advancement of computing power, it has found popularity for online applications as well. The initial guess values of the variables for Newton–Raphson method[16] are taken as the values as existing at the detection of the obstacle and learning rate parameter of 0.8 is introduced for both the variables for better convergence. The iterative process is truncated when the conditions $|\Delta\alpha/\alpha| < \text{Tol}_1$ and $|\Delta T/T| < \text{Tol}_2$ are obtained, where $\text{Tol}_1$ and $\text{Tol}_2$ are respective pre-defined tolerance values of the relative errors, which are selected as variable quantities with the time-to-go to reach the aiming point (set to higher values when $t_{\text{go}}$ is small and vice versa). As a safety measure for onboard computational requirement, a maximum value of 25 iterations is also put in the simulation studies. Other details of this mechanization are omitted for brevity.

After generating the guidance commands, a set of first-order autopilot loop has been constructed to validate the proposed algorithm. In other words, the actual values of $\mu$, $\alpha$ and $V_T$ that are taken to simulate the system dynamics in Eq. (2) are computed by propagating the following set of equations:

$$\dot{\mu} = -k_\mu(\mu - \mu_d), \tag{37}$$

$$\dot{\alpha} = -k_\alpha(\alpha - \alpha_d), \tag{38}$$

$$\dot{T} = -k_T(T - T_d), \tag{39}$$

Note that the gains for the autopilots, i.e., $k_\mu > 0$, $k_\alpha > 0$ and $k_T > 0$ are selected higher than the gains for the guidance loop, i.e., $k_\chi$, $k_\gamma$ and $k_{V_T}$, respectively, as the inner loop must be faster than the outer loop.

### 3.3. *Extension of the algorithm for multiple obstacles*

Without loss of generality, multiple stationary obstacles are considered here in the problem formulation. The formulation can be treated as 'reconfigurable' in the sense that if some of the obstacles are in close proximity with each other, then one composite obstacle is considered encircling all of these obstacles rather than individual obstacles. The collision avoidance algorithm is implemented considering only the composite obstacles and remaining individual obstacles.

#### 3.3.1. *Formulation of reconfigurable obstacles*

Let us consider the onboard sensor mounted on the UAV senses $N$ obstacles. The information of position and safety sphere radius of each obstacle are obtained through the onboard sensors. First all the $N$ obstacles' position and safety radius are assigned in a vector form as $X$ and $R_v$, respectively. One of the obstacles is considered as global obstacle say $X_{\mathrm{OG}}$ and the distance between this obstacle center and remaining other $N-1$ obstacles centers are computed. If any of the $N-1$ obstacles are in close proximity of the global obstacle (i.e., distance between the two obstacle centers is less than the summation of two obstacle safety sphere radius and minimum safety distance considered between them) then one composite safety sphere is formed, which encompassing the safety spheres of both global obstacle and the close proximity obstacle in following manner. Let us assume that $i$th obstacle in obstacle position vector $X(:,i)$ is in close proximity to global obstacle, then a unit vector along the line joining the center of the global obstacle and close proximity obstacle is given as

$$X_{rv} = \frac{X(:,i) - X_{\mathrm{OG}}}{\|X(:,i) - X_{\mathrm{OG}}\|}. \tag{40}$$

The radius and center of composite safety sphere are computed as follows

$$r_a = (r_g + R_v(i) + \|X(:,i) - X_{\mathrm{OG}}\|)/2, \tag{41}$$

$$X_a = X_{\mathrm{OG}} + (r_a - r_g)X_{rv}, \tag{42}$$

where $R_v(i)$ and $r_g$ are the safety sphere radius of the close proximity and global obstacle, respectively. $X_a$ and $r_a$ are the center and radius of the composite safety sphere, respectively. There may be situation such that either one of the obstacle safety sphere completely lie inside the other obstacle safety sphere. In such scenario the computed $r_a$ comes either less than or equal to the bigger obstacle safety sphere radius. Therefore, choose the center and the radius of the bigger obstacle safety sphere as the composite safety sphere. This composite safety sphere is updated as the global obstacle and the process is repeated for remaining obstacles considering one by one steps.

The obstacles which are not in close proximity to the considered global obstacle are regarded as separate obstacles. The position and safety radius information of the separate obstacles are stored in a separate vector as $X_{\mathrm{sep}}$ and $r_{\mathrm{sep}}$, respectively. Further this separate vector $X_{\mathrm{sep}}$ along with global obstacle is put together and it forms a new set of obstacles. This method is iterated for this new set in one by one steps considering each one as global obstacle. Finally, the $N$ obstacle set is transformed into a new final set say $X_{\mathrm{New}}$ which includes clustered global obstacles as well as un-clustered global obstacles. The entire logic is described in Fig. 4 in flowchart format.

#### 3.3.2. *Collision avoidance algorithm for multiple individual obstacles*

Initialize the UAV aiming point as a goal point $X_{\mathrm{ap}} = X_{\mathrm{goal}}$ and corresponding time-to-go as minimum time to reach $t_{\min} = t_{\mathrm{go}}$ and initialize the final set of obstacles, $X_{\mathrm{New}}$. Then compute the aiming point $X_{\mathrm{ap}k}$, time to reach the plane $t_k^*$ and check the collision criteria for the $k$th global obstacle in new set $X_{\mathrm{New}}$ using Eqs. (19) to (22) in Sec. 3.1. If the collision criteria is satisfied and computed $t_k^*$ is lesser than $t_{\min}$ then update the UAV aiming point as $X_{\mathrm{ap}} = X_{\mathrm{ap}k}$ and the minimum time to reach as $t_{\min} = t_k^*$. The above procedure is repeated for remaining all global obstacles in the new set $X_{\mathrm{New}}$ one by one. More detailed logic is given in flowchart as shown in Fig. 5.

## 4. Simulation Results

Exhaustive numerical experiments have been carried out to validate the proposed guidance strategies. It is assumed that the instantaneous position of the obstacles are obtained through the onboard sensors. The experiment was carried out first with kinematic model to ensure the working of guidance design and then validated with point mass model of a real UAV. Simulations with multiple obstacles with different safety sphere sizes around them were also carried out for enhanced confidence.

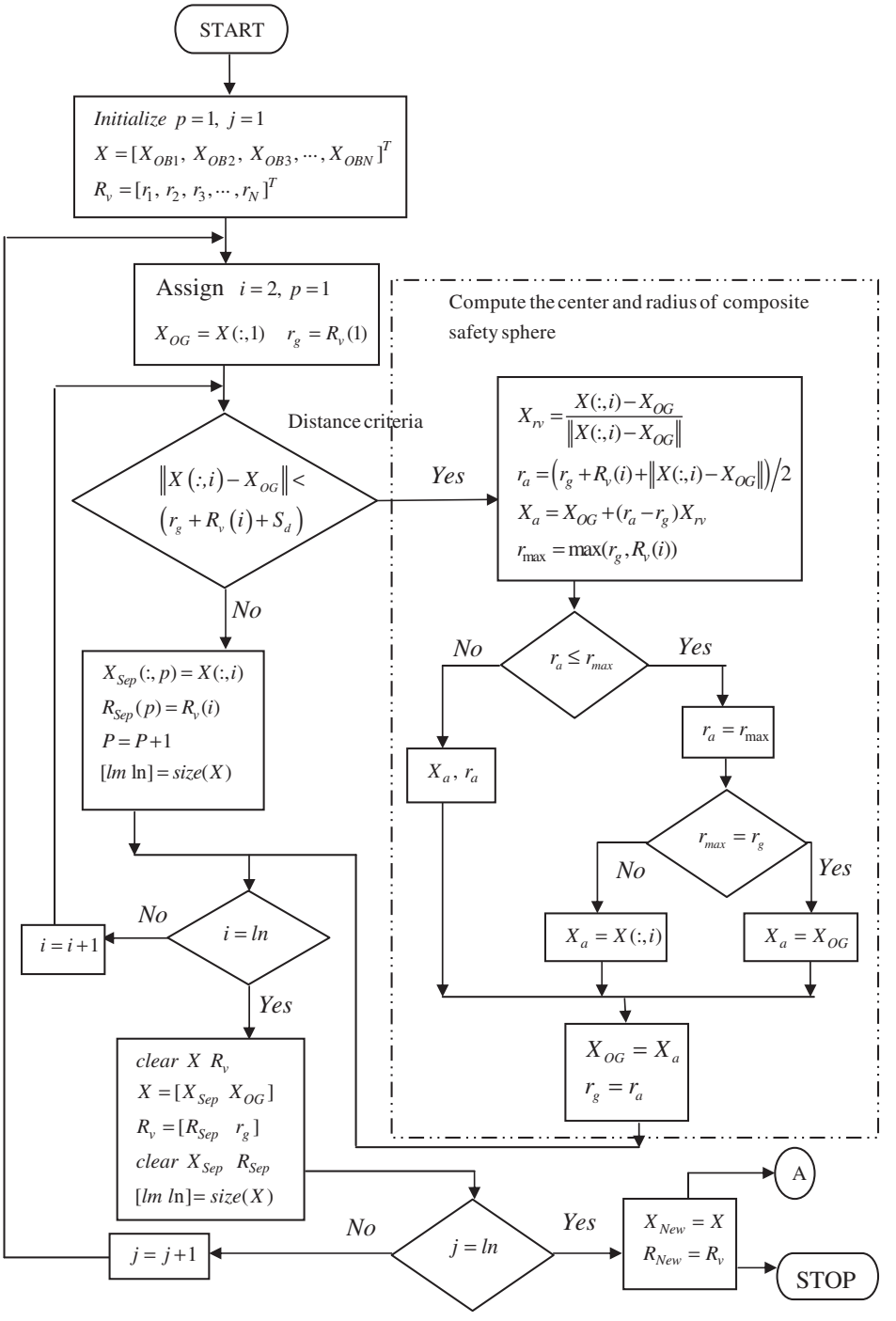
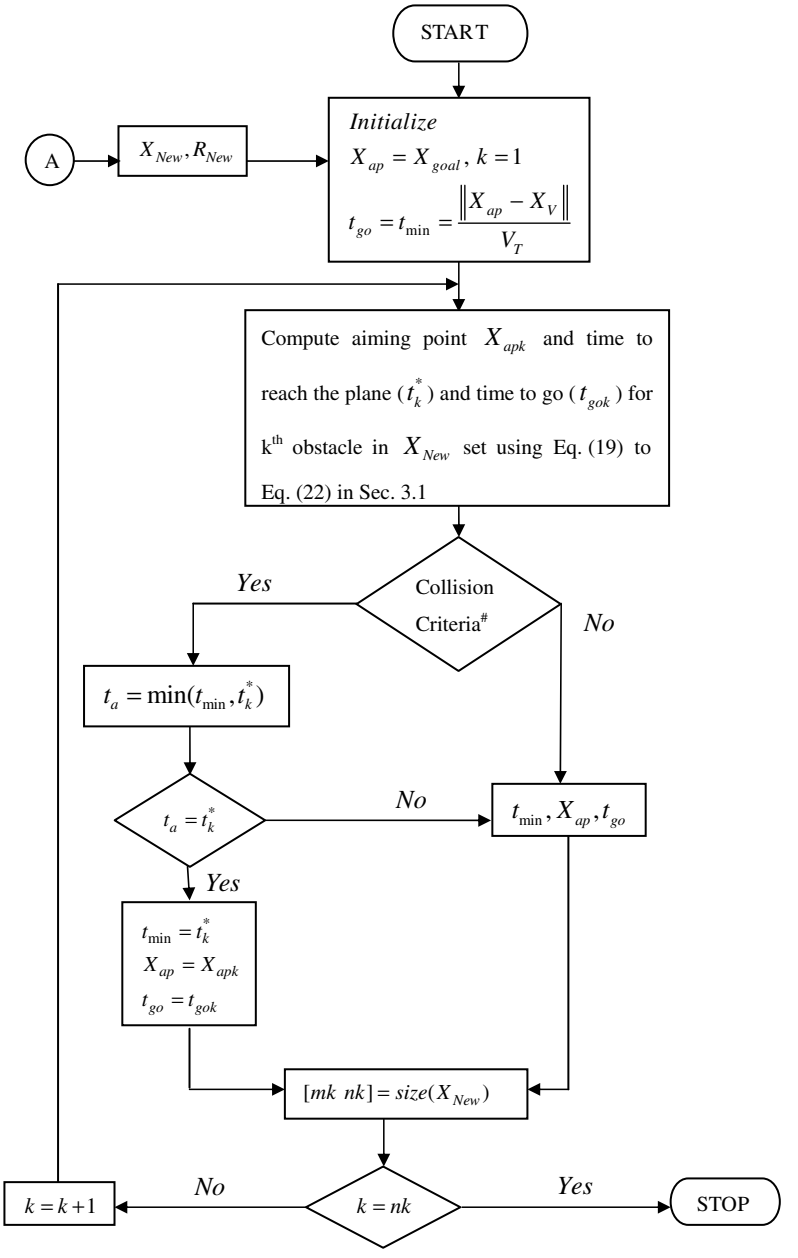



Fig. 4. Flowchart of formulation of reconfigurable obstacles.

### 4.1. *Simulations using kinematic model*

**Case 1.** The position of the stationary obstacle is $X_{\mathrm{OB}} = [78\ 5\ 6]^T$. The starting point of the UAV where it senses the obstacle is $X_{\mathrm{in}} = [0\ 1\ 3]^T$ and the goal point is $X_{\mathrm{goal}} = [200\ 5\ 1]^T$. Initial velocity $V_{\mathrm{initial}} = 20$ m/s, flight path angle $\gamma = 0$ and course angle $\chi = 0$. The radius of the sphere around the obstacle is 40 m.

Figure 6 shows the effectiveness of the guidance algorithm in avoiding the obstacle (with and without sphere




[#] Collision criteria is evaluated using Eq. (21) in Sec. 3.1

Fig. 5. Flowchart for collision avoidance of multiple individual obstacles.

tracking algorithm) and reaching the goal point. However, the sphere tracking algorithm is used here to steer the UAV to avoid any safety violation as well.

Figures 7 and 8 show plot for flight path angle $\gamma$ and course angle $\chi$ with and without sphere tracking algorithm.

**Case 2.** The position of the obstacles are $X_{\text{OB1}} = [30\ \ -2\ 0]^T$ and $X_{\text{OB2}} = [70\ 4\ 3]^T$. The starting point of the UAV where it senses the obstacle is $X_{\text{in}} = [0\ 0\ 2]^T$ and the goal point is $X_{\text{goal}} = [100\ 1\ 2]^T$. Initial velocity is $V_{\text{initial}} = 20$ m/s, flight path angle $\gamma = 0$ and course angle $\chi = 0$. The radius of the sphere around the obstacle is 6 m. Figure 9 shows the effectiveness of the guidance algorithm in avoiding both the obstacles and reaching the goal point.

In Figs. 10 and 11 it can be seen that the flight path angle $\gamma$ and course angle $\chi$ track the guidance command like a first-order system, which leads to successful achievement of the aiming point and the goal point.

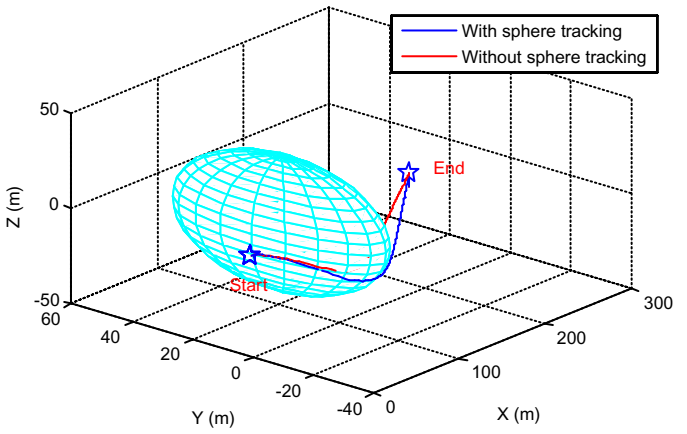



Fig. 6. 3D scenario of obstacle avoidance with and without sphere tracking algorithm.

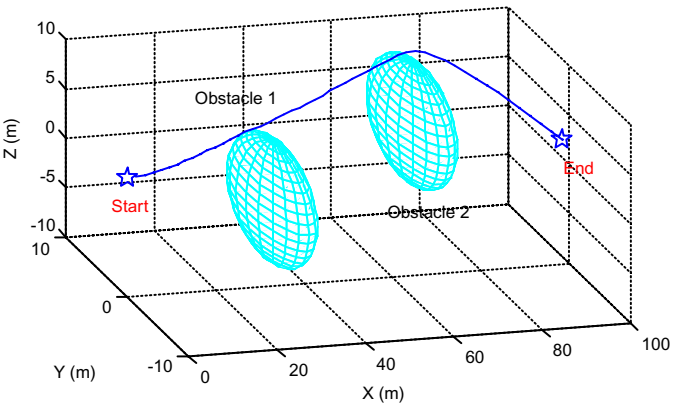



Fig. 9. 3D scenario of obstacles avoidance.

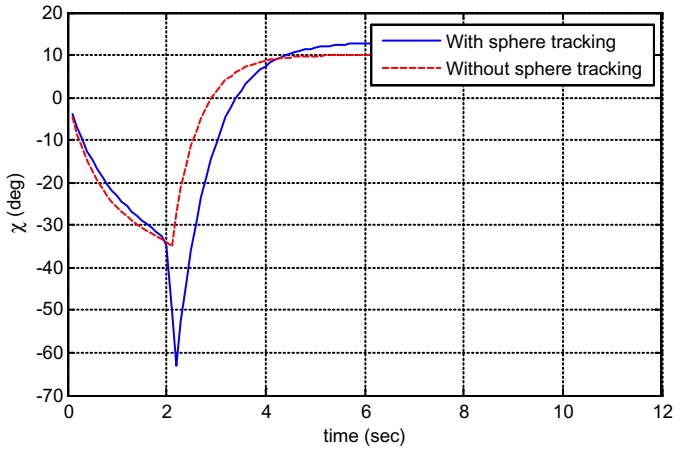



Fig. 7. Flight path angle with and without sphere tracking algorithm.

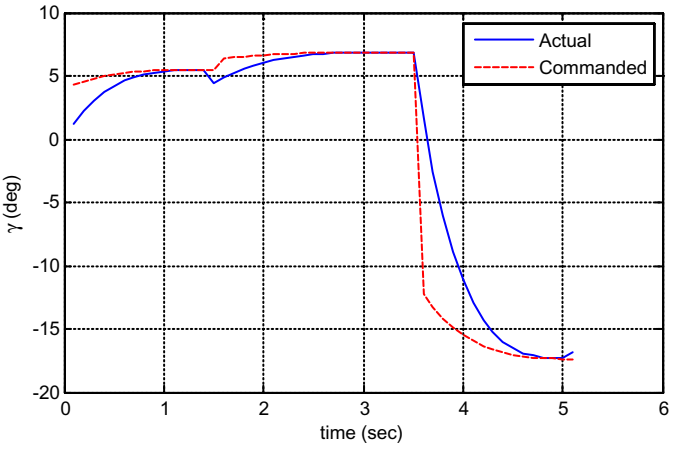



Fig. 10. Flight path angle tracking.

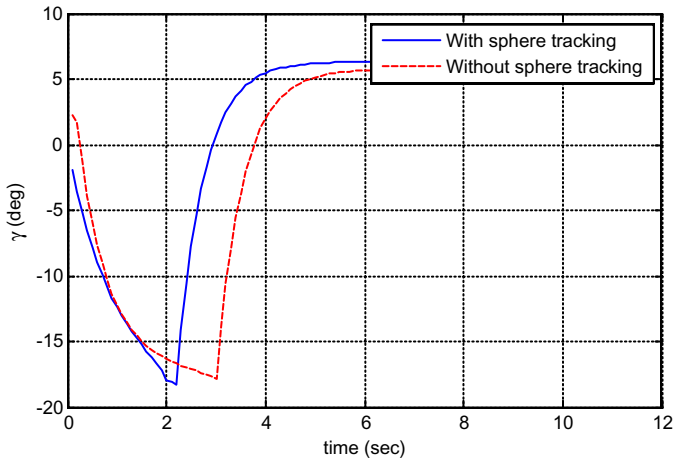



Fig. 8. Course angle with and without sphere tracking algorithm.

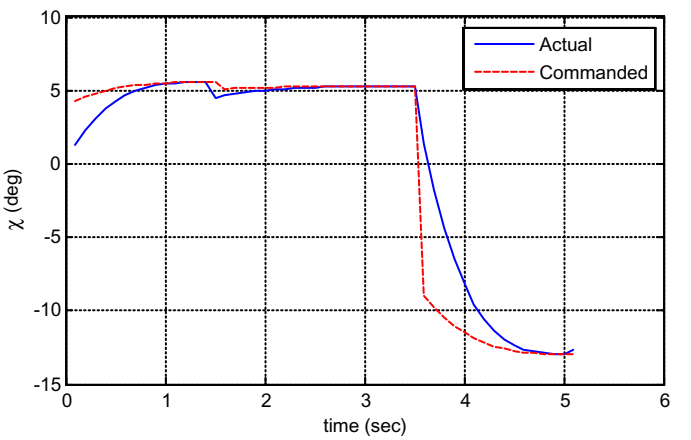



Fig. 11. Course angle tracking.

### 4.2. *Simulations using point mass model*

Numerical simulations are carried out with the realistic UAV model (AE-2).[12,13] Initial velocity of UAV is $V_{\text{initial}} = 20$ m/s, flight path angle $\gamma = 2.85°$ and course angle $\chi = 2.86°$. Two cases are considered based on multiple obstacles configuration. Results with first-order autopilot and with ideal autopilot (without the effect of autopilot i.e., the control with infinite bandwidth) for all the control variables are demonstrated in all the different cases.

**Case 1.** The starting point of the UAV where it senses the obstacle is $X_{\text{in}} = [0\ 1\ 3]^T$ and the goal point is $X_{\text{goal}} = [200\ 4\ 4]^T$. The position of the stationary obstacles are $X_{\text{OB1}} = [31\ 7\ 7]^T$, $X_{\text{OB2}} = [78\ 5\ 6]^T$, $X_{\text{OB3}} = [115\ -4\ 1]^T$ and $X_{\text{OB4}} = [150\ 5\ 3]^T$, and the radius of the corresponding safety spheres are $r_1 = 8$ m, $r_2 = 7$ m, $r_3 = 6$ m and $r_4 = 7$ m. Figure 12 represents the case with four obstacles at a time which are averted while UAV reaches the goal point.

Figure 13 represents the $\alpha_d$ control profile which responds whenever the obstacle is averted. It can be seen that the control with autopilot is smoother than the control with ideal autopilot.

Figure 14 represents the $\mu_d$ control profile which responds whenever the obstacle is avoided. Control is smoother in auto pilot case compared to ideal auto pilot.

Figure 15 shows the thrust profile $T_d$. The response of the thrust with autopilot becomes more distinct compared to the case of ideal autopilot.

Figure 16 shows the tracking of guidance commands by the direction angles of the velocity as a first-order system. The command changes for avoiding each obstacle sometimes shows a sudden jump as the goal point becomes the aiming point.

Figure 17 represents the zoomed plot of four relative distances between the UAV and the obstacles. In all of the cases UAV does not enter into safety sphere of the



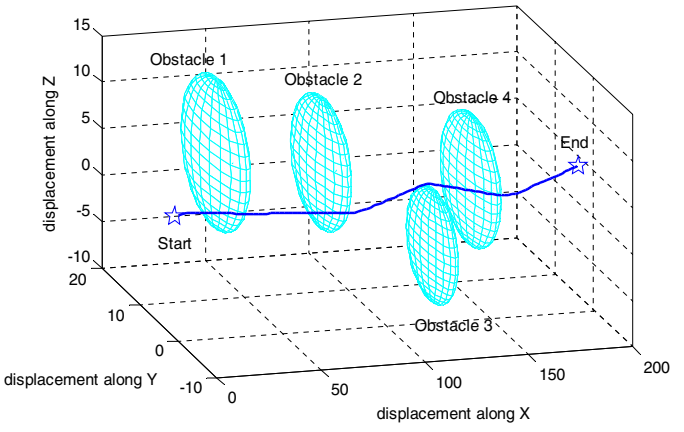



Fig. 12. 3D scenario of obstacles avoidance.

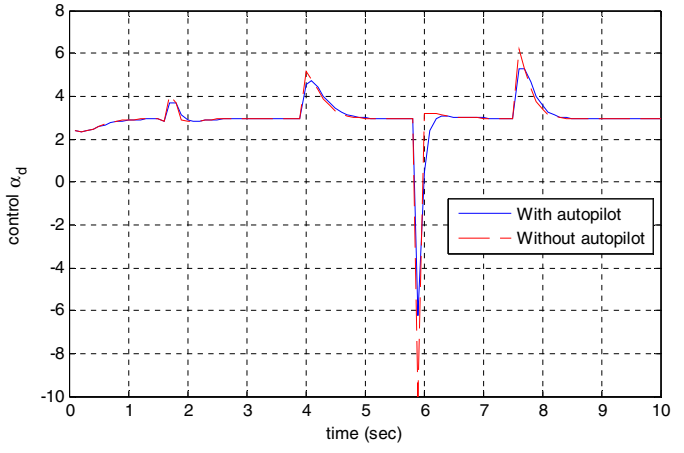



Fig. 13. Commanded angle of attack $\alpha_d$.

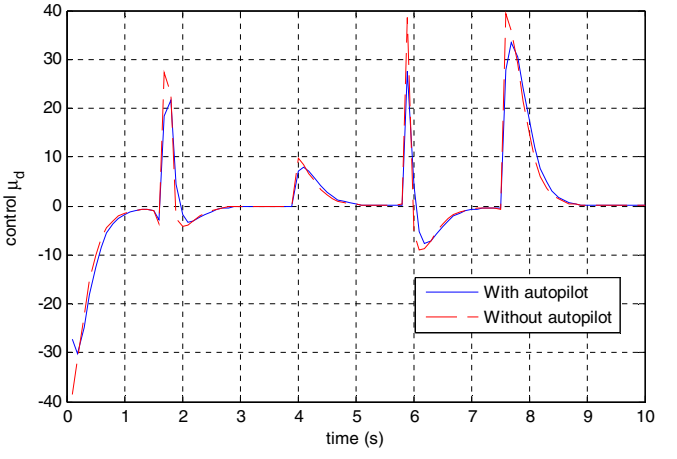



Fig. 14. Commanded bank angle $\mu_d$.

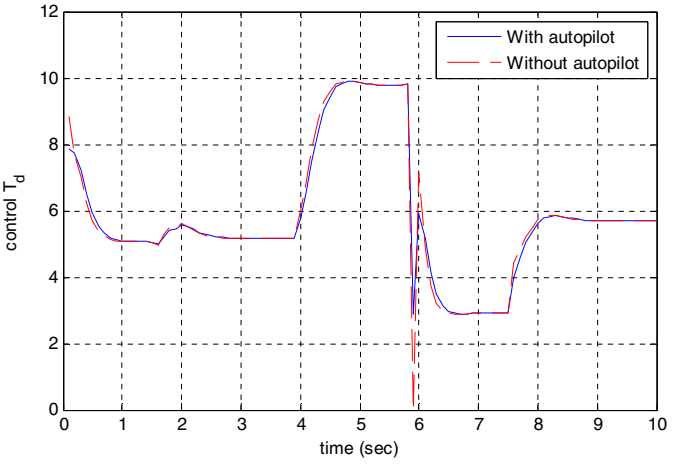



Fig. 15. Commanded thrust profile $T_d$.

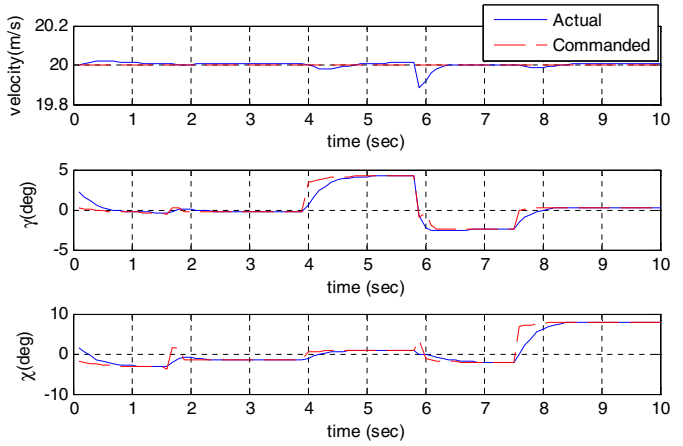



Fig. 16. Tracking of guidance commands.

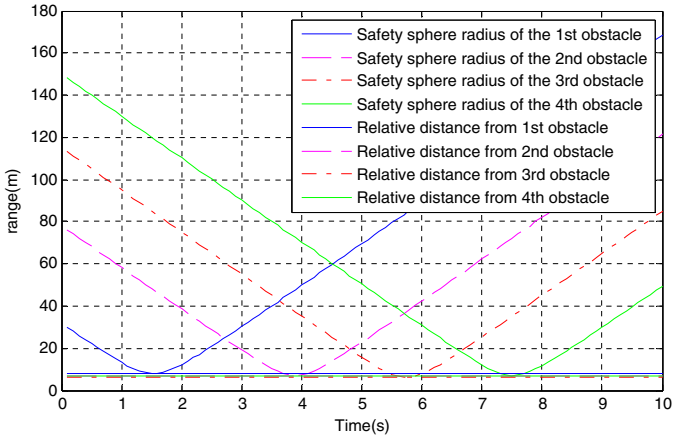



Fig. 17. (Color online) Relative distance of UAV from obstacles.

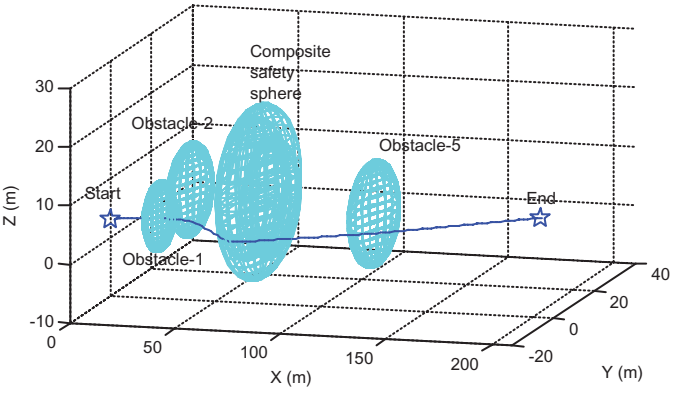



Fig. 18. 3D scenario of obstacles avoidance.

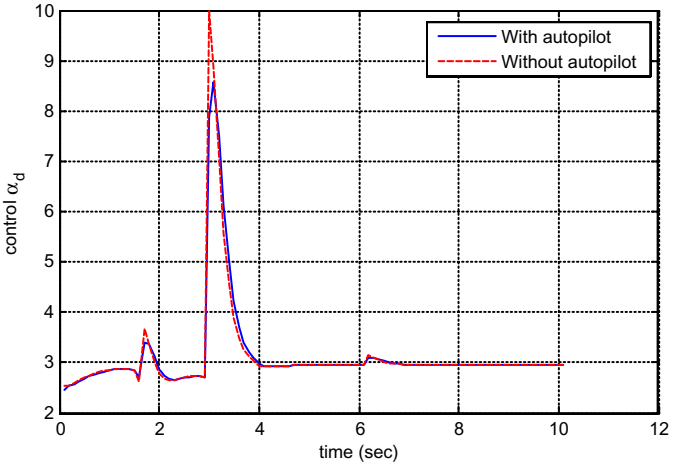



Fig. 19. Commanded angle of attack $\alpha_d$.

respective obstacles. Note that Figs. 16 and 17 represent the cases with the autopilot delay.

**Case 2.** The starting point of the UAV where it senses the obstacle is $X_{\text{in}} = [0\ 0\ 3]^T$ and the goal point is $X_{\text{goal}} = [200\ 4\ 6]^T$. The position of the stationary obstacles are $X_{\text{OB1}} = [31\ 7\ 7]^T$, $X_{\text{OB2}} = [31\ -8\ 7]^T$, $X_{\text{OB3}} = [68\ 8\ 6]^T$, $X_{\text{OB4}} = [61\ 6\ 8]^T$ and $X_{\text{OB5}} = [120\ 5\ 5]^T$, and the corresponding radius of the spheres are $r_1 = 8$ m, $r_2 = 6$ m, $r_3 = 10$ m, $r_4 = 12$ m and $r_5 = 9$ m. Figure 18 shows the 3D scenario of multiple obstacle avoidance with autopilot in loop. It can be clearly seen that the obstacles 3 and 4 are in close proximity to each other, so the algorithm forms one composite safety sphere which encloses both the obstacles. Finally five obstacles turn out to be a four obstacles case and UAV is capable of maneuvering away from all the obstacles using newly computed aiming point.

It can be seen from Fig. 19 that the control with autopilot is smoother than the control with ideal autopilot.

Note that, the control demand changes four times; for the first three instances it is for maneuvering away from the obstacles and for fourth instant it is used for driving towards the final goal point.

Similarly, in Fig. 20 the demand in bank angle also changes four times, for the first three instances for collision avoidance and for fourth instant for driving towards the final goal point. The bank angle rate $\dot{\mu}_d$ is slightly higher in case of ideal autopilot than autopilot. With the autopilot, the rate $\dot{\mu}_d$ reduces which makes the profile of $\mu_d$ smoother.

Figure 21 represents the thrust profile. The thrust profile with ideal autopilot shows the chattering behavior which is smoothened out, in case of autopilot in loop.

Figure 22 represents the velocity profile which remains almost constant. The direction given by flight path and course angles tracks their commanded values as well.

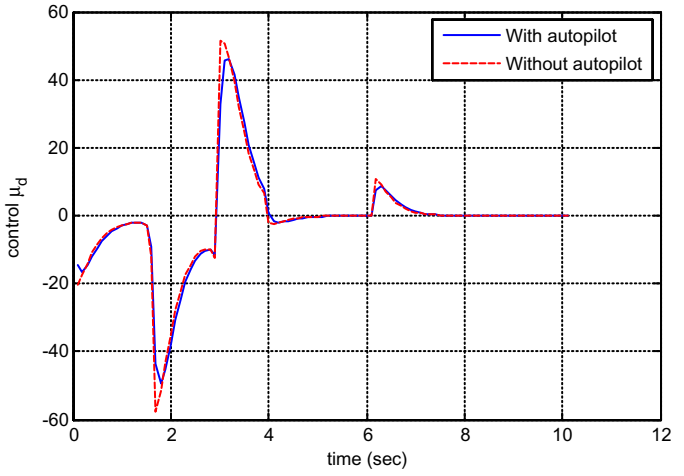



Fig. 20. Commanded bank angle $\mu_d$.

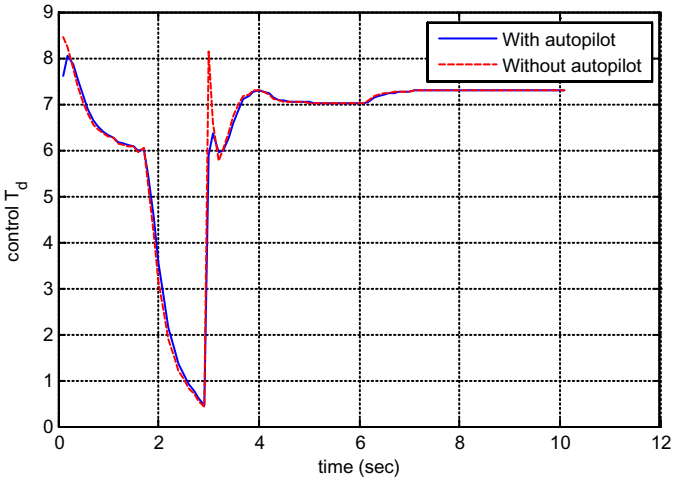



Fig. 21. Commanded thrust profile $T_d$.

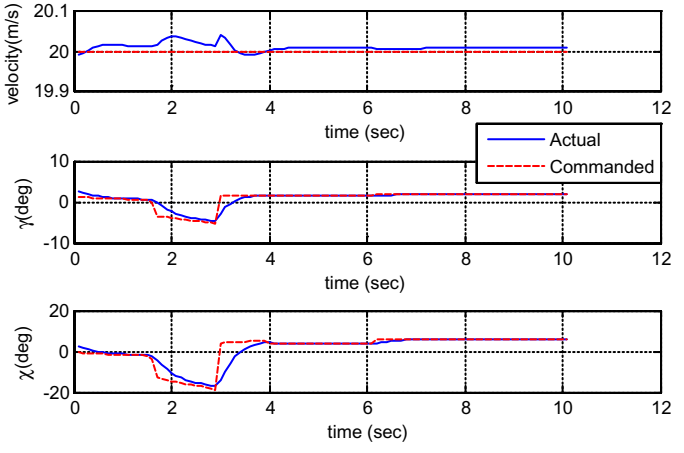



Fig. 22. Tracking of guidance commands.

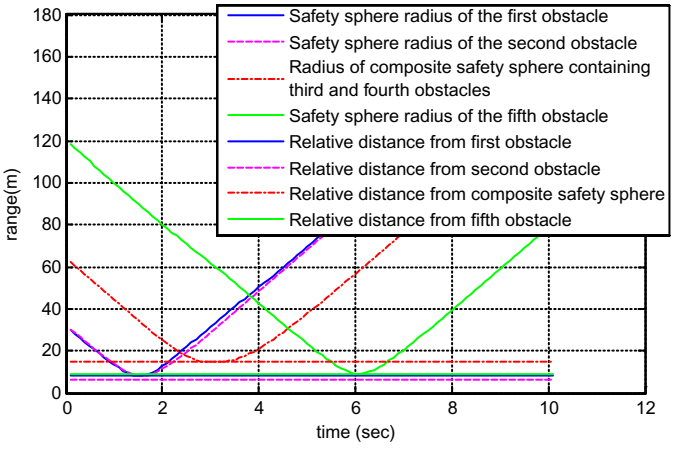



Fig. 23. (Color online) Relative distance of UAV from obstacles.

Figure 23 shows the relative zoomed plot of the distance between the UAV and the obstacles. In all of the cases, UAV does not enter into the safety spheres of the respective obstacles.

## 5. Conclusion

Based on nonlinear dynamic inversion, an aiming point guidance algorithm is presented in this paper for reactive collision avoidance of unmanned aerial vehicles. When a pop-up obstacle is detected in the vicinity, an imaginary safety sphere is put around the center of the obstacle. Next, the velocity vector of the vehicle is realigned towards an 'aiming point' within a fraction of the available time-to-go, which guarantees safe avoidance of the obstacle without too much of deviation from the original path. The resulting guidance law is in closed form and hence it is possible to implement it online without any computational complexity issues. In case of multiple obstacles scenario, the presented logic can find a set of obstacles which consist of both composite safety spheres (enclosing of close proximity obstacles) as well as the individual obstacles. This will eventually reduce the complexity of the problem by reducing the number of obstacles. Further, the aiming point computation for UAV to the closer obstacle is done by checking the collision criterions for all obstacles in the set and then computing the aiming point of the conflict obstacle which poses the minimum time to reach the collision plane. The logic presented here is more realistic, intuitive and simulation results obtained are quite promising.

## Acknowledgment

This work is supported by AOARD/AFRL, USA under the contract number FA-2386-11-1-4096, which operated at

The Society for Innovation and Development (SID), Indian Institute of Science, Bangalore, with the project code SID/PC 99203. We would also like to acknowledge the help and technical support received from Mr. Amit Kumar Tripathi, Ph.D. Student, Indian Institute of Science, Bangalore, while preparing this manuscript.

## References

[1] S. Scherer, S. Singh, L. Chamberlain and M. Elgersma, Flying fast and low among obstacles: Methodology and experiments, *Int. J. Robot. Res.* **27**(5) (2008) 549–574.

[2] J. N. Amin, J. D. Boskovic and R. K. Mehra, A fast and efficient approach to path planning for unmanned vehicles, in *Proc. AIAA Guidance, Navigation, and Control Conf. Exhibit*, Keystone, CO, August 2006.

[3] A. Mujumdar and R. Padhi, Evolving philosophies on autonomous obstacle/collision avoidance of unmanned aerial vehicles, *J. Aerosp. Comput. Inform. Commun.* **8**(2) (2011) 17–41.

[4] P. Zarchan, *Tactical and Strategic Missile Guidance*, 5th edn. (AIAA, 2007).

[5] S. C. Han and H. Bang, Proportional navigation-based collision avoidance for UAVs, *Int. J. Control, Autom. Syst.* **7**(4) (2009) 553–565.

[6] Y. Watanabe, A. J. Calise and E. N. Johnson, Minimum effort guidance for vision-based collision avoidance, in *Proc. AIAA Atmospheric Flight Mechanics Conf. Exhibit*, Keystone, Colorado, August 2006.

[7] A. Chakravarthy and D. Ghose, Obstacle avoidance in a dynamic environment: A collision cone approach, *IEEE Trans. Syst. Man Cybern. A, Syst. Humans* **28**(1) (1998) 562–574.

[8] N. A. Shenoydor, *Missile Guidance and Pursuit: Kinematics, Dynamics and Control* (Horwood Publishing Limited, 1998).

[9] 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.* **29**(2) (1998) 95–102.

[10] A. Mujumdar and R. Padhi, Reactive collision avoidance using nonlinear geometric and differential geometric guidance, *J. Guid. Contr. Dynam.* **34**(1) (2011) 303–310.

[11] C. Carbone, U. Ciniglio, F. Corraro and S. Luongo, A novel 3D geometric algorithm for aircraft autonomous collision avoidance, in *Proc. 45th IEEE Conf. Decision and Control*, San Diego, CA, USA, December 2006.

[12] D. G. Hull, *Fundamentals of Airplane Flight Mechanics* (Springer, 2007).

[13] V. Surendranath, S. P. Govindaraju, M. S. Bhat and C. S. N. Rao, Configuration development of all electric mini airplane, Project report, Project Ref. No: ADEO/MAE/VSU/001, Dept. of Aerospace Eng., Indian Institute of Science, Bangalore.

[14] S. Singh and R. Padhi, Automatic path planning and control design for autonomous landing of UAVs using dynamic inversion, in *Proc. American Control Conf.* St. Louis, USA (2009).

[15] R. Padhi and S. N. Balakrishnan, Implementation of pilot commands in aircraft control: A new approach based on dynamic inversion, *AIAA Guidance, Navigation, and Control Conf. Exhibit*, Austin, Texas, 11–14 August 2003.

[16] K. E. Atkinson, *An Introduction to Numerical Analysis* (John Wiley & Sons, 2001).

[17] S. P. Williams, R. F. Antoniewicz, E. L. Duke and P. K. A. Menon, Study of a Pursuit-Evasion guidance law for high performance aircraft, in *Proc. American Control Conf.* Pittsburg, PA, USA, 21–23 June 1989.

[18] R. Padhi and A. Gupta, Dynamic estimation of obstacle position with vision sensing for reactive collision avoidance of UAVs, *J. Aerosp. Sci. Technol.* **64**(3) (2012) 187–200.



**Ramsingh G. Raja** received his Bachelor's degree in Electronic and Instrumentation Engineering from Anna University, Chennai, India in 2009, followed by his Master's degree in Signal Processing and Control from the National Institute of Technology, Hamirpur, India in 2012. He is currently working with Dr. R. Padhi as a Project Associate in the Department of Aerospace Engineering, at the Indian Institute of Science, Bangalore. Mr. Raja is involved in the development of the Collision Avoidance Algorithm for unmanned aerial vehicles and has also worked on the design of an Optimal Guidance Law for reusable launch vehicles. His research interests are nonlinear control, optimal control and image processing.

**Charu Chawla** completed her Masters in Aerospace Engineering from the Indian Institute of Science Bangalore, India, in 2010. She joined the company, General Motors Technical Centre India, as Senior Engineer in August 2010, where she is currently working on the control algorithm centric development of hybrid vehicles.

Ms. Chawla worked as a Project Assistant from 2006 to 2008 with Dr. R. Padhi on designing of advanced guidance and control algorithms for aerospace vehicles. She first worked on an Indian Space Research Organisation project where an optimal guidance law for a reusable launch vehicle in the descent phase was designed. This was followed by a project from the Defence R&D Organisation of India, where she focussed on designing a partially integrated guidance and control (PIGC) law for the terminal phase of interceptors against high speed re-entry targets. Subsequently, she did her M.S. with Dr. Padhi, with her research focussed on the obstacle avoidance of unmanned aerial vehicles, where the PIGC concept was successfully explored as a solution for this problem as well. The PIGC algorithms were designed on the basis of optimal and nonlinear control theories and takes into account the full nonlinear Six-DOF models of the vehicles. Ms. Chawla has 10 publications in international journals and conference proceedings.

**Dr. Radhakant Padhi** received his Ph.D. in Aerospace Engineering from the Missouri University of Science and Technology (MST), USA, in 2001. After two years of postdoctoral studies in MST, he joined the Dept. of Aerospace Engineering at the Indian Institute of Science, Bangalore, as a faculty member in December 2003, where he is currently an Associate Professor.

Dr. Padhi's research interests are in the broad area of systems theory and applications, with emphasis on synthesis algorithms in optimal control, nonlinear control, intelligent control and state estimation. He works on diverse application areas such as the guidance and control of aerospace vehicles, target tracking for guidance, integrated guidance, control and estimation, automatic drug delivery for biomedical systems, control of distributed parameter systems and industrial process control. Dr. Padhi is an Associate Fellow of AIAA, Senior Member of IEEE, Member of the IFAC Technical Committee on Aerospace, and Life member of both the Aeronautical Society of India and the Systems Society of India. He is also the vice-president of the Automatic Control and Dynamic Optimization Society, which is the NMO of IFAC in India. Dr. Padhi has more than 150 publications in international journals and conference proceedings. The guidance and control algorithms developed by Dr. Padhi have found good acceptance in the Indian Space Research Organisation and the Defence R&D Organisation of India. He has received several awards for his valuable research contributions as well.