Journal of Intelligent & Robotic Systems (2020) 97:227–248
https://doi.org/10.1007/s10846-019-01055-5

Three Dimensional Collision Avoidance for Multi Unmanned Aerial Vehicles Using Velocity Obstacle

Chee Yong Tan[1] · Sunan Huang[2] · Kok Kiong Tan[3] · Rodney Swee Huat Teo[2]

Received: 30 September 2018 / Accepted: 17 June 2019 / Published online: 9 January 2020

Abstract

Recently, multi-UAV systems are attracting growing interests. It offers wide range of applications in civilian and military environment, carrying out dangerous missions that manned aircraft cannot offer. Currently, an important challenge is the collision avoidance algorithm. The idea is to use the collision avoidance algorithm to control the multi-UAV systems. This will guarantee the safety of the UAVs. The UAVs will complete the missions without colliding with any moving or static obstacles. This topic has motivated the development of various collision avoidance algorithms. In this paper, we proposed some improvements on the three-dimensional velocity obstacle algorithm proposed in Jenie et al. (J. Guid. Control Dyn. **39**(10), 2312–2323 2016). Our improvements are threefold. First, we indicate the limitations of the original 3D collision avoidance method and present the modifications on the algorithm. Second, we develop the velocity obstacle method to be capable of handling cube obstacles in 3D space. Third, a real flight test is conducted to verify the effectiveness of the proposed three-dimensional velocity obstacle method.

Keywords Three-dimensional collision avoidance · Three-dimensional velocity obstacle method · Unmanned aerial vehicles

1 Introduction

Currently, research in Unmanned aerial vehicles (UAVs) is a growing area. This technology offers a wide range of applications, inspiring researchers to develop various ideas for the UAV systems [1–5]. Employing multi-UAV systems is increasingly possible with the advancement in computer hardware and communication technology. An important challenge in multi-UAV systems is collision avoidance issue between the UAV itself and obstacles.

Collision avoidance should guarantee the safety of multi-UAV systems. Usually, when the UAVs are performing its mission, there exist multiple dynamic and static obstacles in the environment. Various collision avoidance algorithms have been reported in recent years. A report [6] surveys the development of this research area, discussing sensors, decision algorithms, and path following. Here, we discuss several typical approaches. In [7], the authors propose a trajectory generation and tracking method for avoiding control; in [8], the authors present a varying cell strategy which is integrated into trajectory planning to avoid the obstacles; in [9], the authors develop a path planning method by using discrete stage and continuous stage, where the discrete planner designs a path for each vehicle, while the continuous stage focuses on the trajectory planning of each vehicle, avoiding collisions and respecting downwash constraints; in [10, 11], the authors develop a reactive collision avoidance that acts when the possibility of a collision is detected; in [15], the authors introduce a rolling-horizon policy for multi-UAV system, where the legendre pseudospectral collocation techniques are used to obtain efficient trajectory for avoiding

A short version of this paper was presented in ICUAS 2018.

✉ Sunan Huang
tslhs@nus.edu.sg

Chee Yong Tan
tancheeyong.10@gmail.com

Kok Kiong Tan
kktan@nus.edu.sg

Rodney Swee Huat Teo
tsltshr@nus.edu.sg

[1] DSO National Laboratories, Singapore, Singapore

[2] Temasek Laboratories, National University of Singapore, Singapore, Singapore

[3] Department of Electrical and Computer Engineering, National University of Singapore, Singapore, Singapore

obstacles; in [12, 13] the authors propose the conflict detection to determine the trajectory when future collision is detected; in [16, 17], the authors assumes that the vehicles are in a field of force which composes of attractive and repulsive force, and develop an improved potential field method, where the vehicle experiences attractive force to its destination and repulsive force to avoid static and moving obstacles; in [23], the authors introduce collision cone concept for collision avoidance design; in [14, 18, 20–22], based on [23], the authors develop the velocity obstacle method in two-dimensional (2-D) environments; in [19, 24, 25], the velocity obstacle algorithm is developed to handle the case where moving or static obstacles are in three-dimensional (3-D) form. Due to the geometric concept being simpler and less computational demanding than the other algorithms, the velocity obstacle method is becoming popular in collision avoidance.

In this paper, we will focus on velocity obstacle method for coping with 3D obstacles, extending the collision avoidance design of the work proposed in [24, 25]. We take into consideration the presence of static and dynamic obstacles in the 3D environment. The main contributions of this paper are:

- Improvement on the 3-D velocity obstacle **VO** method proposed in [25];
- Development of a collision avoidance algorithm that works with static obstacles by building the pyramid cone method;
- Implement the proposed algorithm in a real UAV platform.

This paper structure is as follows. In Section 2, we review the original 3-D velocity obstacle **VO** method. In Section 3, we concentrate on the improvement of existing method in handling moving obstacles. These changes include VO cone's symmetry line, avoiding an incomplete parabolic or hyperbolic shape, and simplifying spherical protected zone. In Section 4, we focus on dealing with static 3D obstacles. The simulation results are presented in Section 5. In Section 6, a test with real UAV is presented, in order to validate the proposed algorithms. The conclusions are then presented in Section 7.

2 Problem Statements

The purpose of the multi-UAV system is to complete the missions safely without colliding with any static or dynamic obstacles in the airspace. Our concern here is to design a three-dimensional collision avoidance algorithm to drive the vehicles safely while they are performing the missions.

2.1 Original Three-Dimensional Velocity Obstacle

In [24, 25], the authors present a 3-D velocity obstacle (**VO**) method. Each dynamic obstacle is depicted as a point source with a spherical protected zone, S_{pz}, i.e., we treat each UAV as a particle with a radius, a spherical ball. A three-dimensional situation between an avoiding vehicle at position $\overrightarrow{\boldsymbol{p}_A}$ and moving obstacle at position $\overrightarrow{\boldsymbol{p}_B}$ is depicted in Fig. 1. Their velocities corresponding to the vehicle A and B are $\overrightarrow{V}_{\mathrm{A}}$ and $\overrightarrow{V}_{\mathrm{B}}$ respectively. By collecting all relative velocity extension $\overrightarrow{V}_{\mathrm{R}}$ ($\overrightarrow{V}_{\mathrm{R}} = \overrightarrow{V}_{\mathrm{A}} - \overrightarrow{V}_{\mathrm{B}}$) that cuts through the S_{pz}, we form a 3-D collision cone **CC** (see Fig. 1a), where ψ_{az} is the azimuth angle of the collision cone, θ_{el} is the elevation angle of the collision cone, d_{oi} is the distance of the avoiding vehicle from the moving obstacle, r_{pz} is the radius of the spherical protected zone, and B_{CC} is the base circle of the collision cone. It should be noticed that r_{pz} is a safe distance which is given by users. For example, according to our experience, r_{pz} may be selected as $\rho \times V_{Max}$ with constant ρ ($\rho \geq 1$),where V_{Max} is the maximum speed.

The definition of the velocity obstacle implies that if $\overrightarrow{V}_{\mathrm{R}} \in$**CC** then vehicles A and B will collide in the future time. Conversely, if $\overrightarrow{V}_{\mathrm{R}} \notin$**CC**, vehicles A and B are guaranteed to be collision-free. It is useful to build an equivalent condition using absolute velocity of vehicle A. This can be done by adding the velocity of B, $\overrightarrow{V}_{\mathrm{B}}$ to **CC**, i.e., translating the **CC** along the obstacle's velocity $\overrightarrow{V}_{\mathrm{B}}$ to form a three-dimensional velocity obstacle **VO** cone (see Fig. 1b), where A_{vo} is the apex of the VO cone, d_{vo} is the length of the VO cone, C_{vo} is the center of the base of the VO cone, r_{vo} is the radius of the base circle of the VO cone, and B_{vo} is the base cirlce of the VO cone. Collision between the avoiding vehicle and the dynamic obstacle will occur if the avoiding vehicle's velocity, $\overrightarrow{V}_{\mathrm{A}}$, is within the **VO** cone ($\overrightarrow{V}_{\mathrm{A}} \in \boldsymbol{VO}$).

2.1.1 Avoidance Plane

Since it is rather difficult to design avoidance strategy for 3D space, a strategy called the avoidance plane, P_{ϕ}, is introduced in literature [24, 25]. The avoidance plane is a tool used to describe the three-dimensional case into separate two-dimensional planes to resolve for avoidance strategy. Three avoidance planes at 0°, 40° and 90° are depicted in Fig. 2. Each avoidance plane intersects the three-dimensional **VO** cone. The intersection produces a conic-section on each of the avoidance planes. In order to describe the conic-section on each avoidance plane, the **VO** cone must first be mathematically described. It is mathematically represented by its apex $\mathbf{A}_{vo}$, right-base circle, $\mathbf{B}_{vo}$, and the center of the base, $\mathbf{C}_{vo}$. Equations 3, 4 and 5 describe the $\mathbf{A}_{vo}$, $\mathbf{B}_{vo}$ and $\mathbf{C}_{vo}$ respectively.

The avoidance maneuver is any velocity chosen outside of the conic-section on each avoidance plane, P_{ϕ}. The derivation of the conic-section is presented in literature [24, 25].

After describing the three-dimensional case into separate two-dimensional setups, other techniques in two-dimensional can be applied. In this paper, the avoidance plane is define

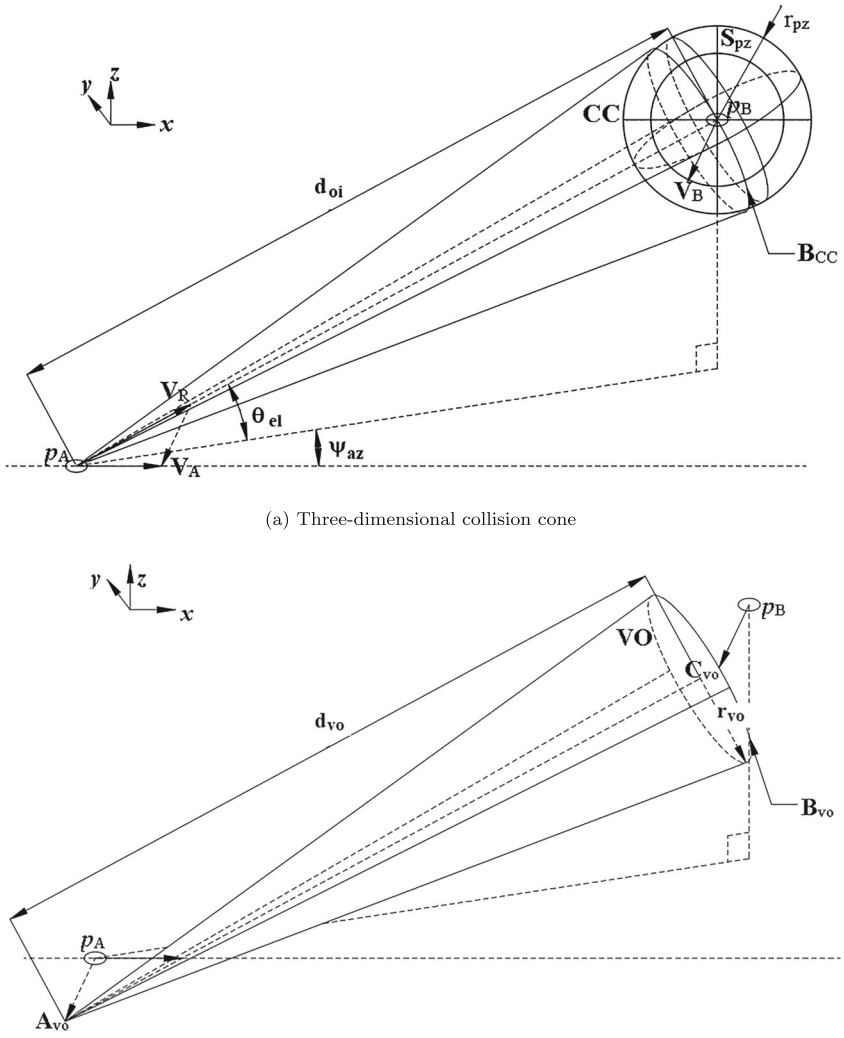


Fig. 1 Original three-dimensional velocity obstacle **VO**

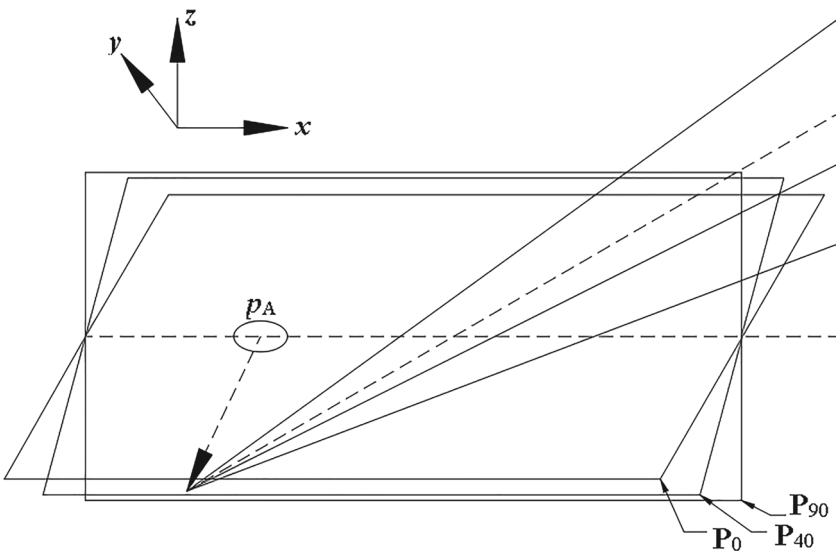


Fig. 2 Three avoidance plane P_0, P_{40} and P_{90}

as any plane which lies the avoiding vehicle's position ,$\overrightarrow{\boldsymbol{p}}_{\mathrm{A}}$. The 0° avoidance plane, P_0, is the xy plane. Vehicle A is defined to be at the origin of every avoidance plane. The range of the avoidance plane is from P_{-90} to P_{90}. In the avoiding plane, the proposed algorithm can find feasible velocity control to get outside of the VO cone.

$$\begin{aligned} d_{oi} &= \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2 + (z_B - z_A)^2}, \\ \theta_{el} &= tan^{-1}\left(\frac{z_B - z_A}{\sqrt{(x_B - x_A)^2 + (y_B - y_A)^2}}\right), \\ \psi_{az} &= tan^{-1}\left(\frac{y_B - y_A}{x_B - x_A}\right) \end{aligned} \tag{1}$$

$$\begin{aligned} d_{vo} &= \frac{{d_{oi}}^2 - {r_{pz}}^2}{d_{oi}}, \\ r_{vo} &= r_{pz}\frac{\sqrt{{d_{oi}}^2 - {r_{pz}}^2}}{d_{oi}} \end{aligned} \tag{2}$$

$$\boldsymbol{A}_{vo} = \begin{bmatrix} x_A \\ y_A \\ z_A \end{bmatrix} = \overrightarrow{\boldsymbol{V}_B} \tag{3}$$

$$\boldsymbol{B}_{vo} = \begin{bmatrix} x_{Bvo} \\ y_{Bvo} \\ z_{Bvo} \end{bmatrix} = \boldsymbol{R}_{\theta_{el}|\psi_{az}} \begin{bmatrix} d_{vo} \\ r_{vo}cost \\ r_{vo}sint \end{bmatrix} + \overrightarrow{\boldsymbol{V}_B}, \quad 0 \le t \le 2\pi \tag{4}$$

$$\boldsymbol{C}_{vo} = \begin{bmatrix} x_{Cvo} \\ y_{Cvo} \\ z_{Cvo} \end{bmatrix} = \boldsymbol{R}_{\theta_{el}|\psi_{az}} \begin{bmatrix} d_{vo} \\ 0 \\ 0 \end{bmatrix} + \overrightarrow{\boldsymbol{V}_B} \tag{5}$$

$$\boldsymbol{R}_{\theta_{el}|\psi_{az}} = \begin{bmatrix} cos\psi_{az}cos\theta_{el} & -sin\psi_{az} & -cos\psi_{az}sin\theta_{el} \\ sin\psi_{az}cos\theta_{el} & cos\psi_{az} & -sin\psi_{az}sin\theta_{el} \\ sin\theta_{el} & 0 & cos\theta_{el} \end{bmatrix} \tag{6}$$

3 Enhancing the Original Three-Dimensional Velocity Obstacle VO

Investigation on the original three-dimensional velocity obstacle **VO** method has discovered limitations. This section presents the limitations and proposed improvement to the equations and algorithms involved.

3.1 Proposed Improvement

The limitations and solutions are presented in this section. The solutions have been implemented and tested.

3.1.1 Hyperbolic Conic-Section

In literature [24, 25], the authors present a hyperbolic shape produced on the avoidance plane. However, it uses a condition ${z_{Bvo}}^{\phi} \neq {z_A}^{\phi}$ instead of $0 \le t_g = -\frac{{z_A}^{\phi}}{{z_{Bvo}}^{\phi} - {z_A}^{\phi}} \le 1$ (we will explain t_g later). This results in an error side of the hyperbolic shape due to another additional VO cone produced when t_g is negative, where the hyperbolic shape represents the two VO cones (one at positive t_g and one at negative t_g) intersecting the avoidance plane. Therefore, it produces the hyperbolic shape as shown in Fig. 3. The error results in the algorithm ignoring some avoidance velocities.

The equations that represent the conic-section have been modified to remove the error side of the hyperbolic. In the rotation plane with the rotation angle ϕ, the apex $\boldsymbol{A}_{vo}$ as shown in Eq. 3 is translated into

$$\boldsymbol{A}_{vo}{}^{\phi} = \begin{bmatrix} {x_A}^{\phi} \\ {y_A}^{\phi} \\ {z_A}^{\phi} \end{bmatrix} = \boldsymbol{R}_{P_\phi}\boldsymbol{A}_{vo}, \tag{7}$$

where the translation matrix is given by

$$\boldsymbol{R}_{P_\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\phi & sin\phi \\ 0 & -sin\phi & cos\phi \end{bmatrix}, \tag{8}$$

while the base $\boldsymbol{B}_{vo}$ as shown in Eq. 4 is translated into

$$\boldsymbol{B}_{vo}{}^{\phi} = \begin{bmatrix} {x_{Bvo}}^{\phi} \\ {y_{Bvo}}^{\phi} \\ {z_{Bvo}}^{\phi} \end{bmatrix} = \boldsymbol{R}_{P_\phi}\boldsymbol{B}_{vo}. \tag{9}$$

The line of the VO cone that connects the apex and the base is expressed by

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = (B_{vo}^{\phi} - A_{vo}^{\phi})t_g + A_{vo}^{\phi} \tag{10}$$

where t_g is a parameter. By considering the conic-section on the xy-body axis,i.e., $z = 0$, the parameter t_g can be derived

$$\left(z_{Bvo}^{\phi} - z_A^{\phi}\right)t_g + z_A^{\phi} = 0 \implies t_g = -\frac{{z_A}^{\phi}}{{z_{Bvo}}^{\phi} - {z_A}^{\phi}} \tag{11}$$

Thus, the equation for avoidance plane is given by

$$\begin{aligned} VO_{P_\phi} &= \begin{bmatrix} x_{vo} \\ y_{vo} \end{bmatrix} \\ &= \begin{bmatrix} -\left({x_{Bvo}}^{\phi} - {x_A}^{\phi}\right) \bullet \frac{{z_A}^{\phi}}{{z_{Bvo}}^{\phi} - {z_A}^{\phi}} + {x_A}^{\phi} \\ -\left({y_{Bvo}}^{\phi} - {y_A}^{\phi}\right) \bullet \frac{{z_A}^{\phi}}{{z_{Bvo}}^{\phi} - {z_A}^{\phi}} + {y_A}^{\phi} \end{bmatrix}, \\ 0 \le & -\frac{{z_A}^{\phi}}{{z_{Bvo}}^{\phi} - {z_A}^{\phi}} \le 1 \end{aligned} \tag{12}$$

A condition, $0 \le -\frac{{z_A}^{\phi}}{{z_{Bvo}}^{\phi} - {z_A}^{\phi}} \le 1$, is imposed on Eq. 12 to remove the error side of the hyperbolic conic-section.

3.1.2 VO Cone'S Symmetry Line

In literature [24, 25], the authors present an equation to represent the point where the **VO** cone's symmetry line intersects

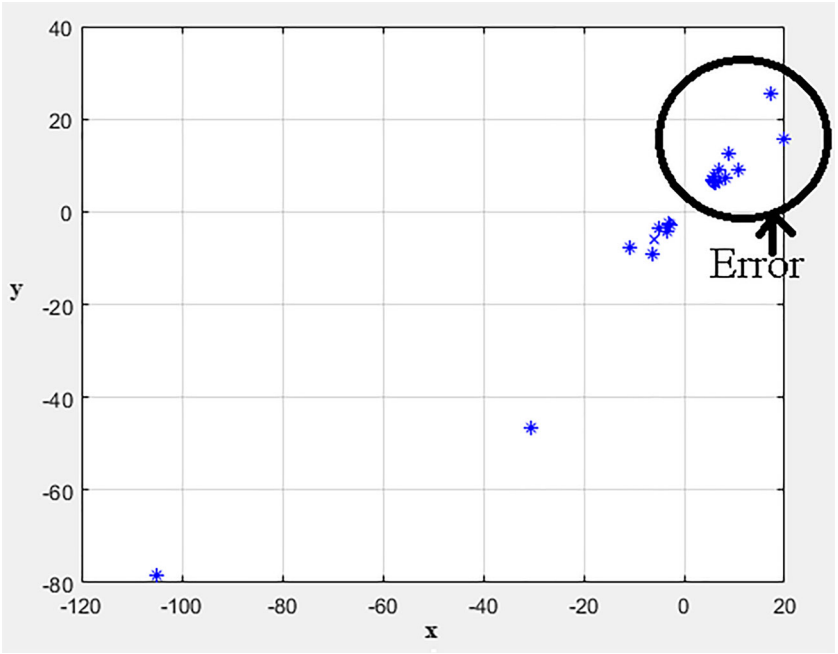


Fig. 3 Hyperbolic conic-section: one branch of the hyperbola represents the curve when t_g is negative; this branch is marked by a circle

the avoidance plane. The equation did not consider instance where the avoidance plane intersects the **VO** cone but did not intersect the **VO** cone's symmetry line. Moreover, the equation generates an error side similar to the hyperbolic conic-section. The symmetry line is useful in other strategies such as dividing the avoidance velocities into two groups.

Modifications have been made to the original equation. The point where the symmetry line intersects the avoidance plane is represented by the following equation

$$H_{P_\phi} = \begin{bmatrix} x_h \\ y_h \end{bmatrix} = \begin{bmatrix} \left({x_{Cvo}}^{\phi} - {x_A}^{\phi}\right) \bullet \left|\frac{-{z_A}^{\phi}}{{z_{Cvo}}^{\phi} - {z_A}^{\phi}}\right| + {x_A}^{\phi} \\ \left({y_{Cvo}}^{\phi} - {y_A}^{\phi}\right) \bullet \left|\frac{-{z_A}^{\phi}}{{z_{Cvo}}^{\phi} - {z_A}^{\phi}}\right| + {y_A}^{\phi} \end{bmatrix}, \quad {z_{Cvo}}^{\phi} \neq {z_A}^{\phi} \tag{13}$$

Absolute condition is imposed on $t_h = \left|\frac{-{z_A}^{\phi}}{{z_{Cvo}}^{\phi} - {z_A}^{\phi}}\right|$. This condition extents the symmetry line as an infinitely long line. Moreover, it removes the error side due to the additional **VO** cone produced at negative t_h.

3.1.3 Two-Dimensional Situations

A two-dimensional situation is formed when the **VO** cone's apex, $\mathbf{A}_{vo}$, vehicle A position, $\overrightarrow{p_{\mathrm{A}}}$, and obstacle B position, $\overrightarrow{p_{\mathrm{B}}}$, are all on one avoidance plane. When the $\mathbf{A}_{vo}$ is on the avoidance plane, it means that ${z_A}^{\phi} = 0$. This results in Eq. 12 reducing to a single point, ${\mathbf{A}_{vo}}^{\phi}$. A triangle conic-section represents a two-dimensional case. However, the original conic-section equation is reduced to a single point, thereby giving a false information to the entire algorithm.

Collision avoidance algorithm should be capable of handling both two-dimensional and three-dimensional situations for the multi-UAV system to operate safely. Two points of the triangle shape are obtained by the **VO** cone's base circle, $\mathbf{B}_{vo}$, intersecting with the avoidance plane (${z_{Bvo}}^{\phi} = 0$). Equations 14 and 15 determine the time t in Eq. 4 when the intersection occurs, and they are given by

$$t_1 = cos^{-1}\left(\frac{A}{\sqrt{B^2 + C^2}}\right) + tan^{-1}1\left(\frac{C}{B}\right), \; t_1 \epsilon R \tag{14}$$

$$t_2 = -cos^{-1}\left(-\frac{A}{\sqrt{B^2 + C^2}}\right) + tan^{-1}1\left(\frac{C}{B}\right), \; t_2 \epsilon R \tag{15}$$

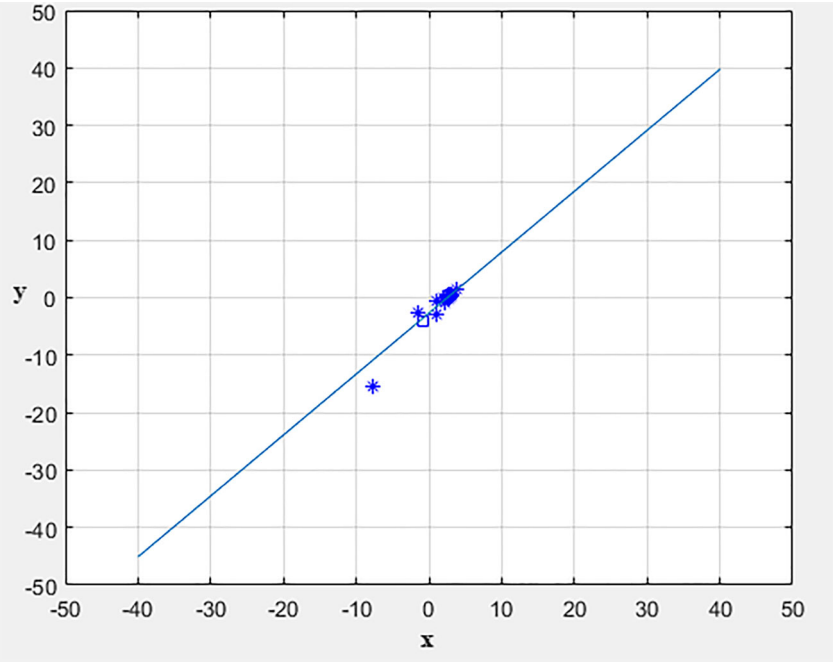
where

$$\begin{aligned} A &= d_{vo}\left(cos\phi sin\theta_{el} - sin\phi sin\psi_{az} cos\theta_{el}\right) + {z_A}^{\phi}, \\ B &= r_{vo} sin\phi cos\psi_{az}, \\ C &= r_{vo}(sin\phi sin\theta_{el} sin\psi_{az} + cos\phi cos\theta_{el}), \\ & t_1 < t_2, \; 0 \leq t \leq 2\pi \end{aligned} \tag{16}$$

Equation 17 defines two points of the triangle conic-section. When both the conditions $t_1 \epsilon R$ and $t_2 \epsilon R$ are true, it indicates that the $\mathbf{B}_{vo}$ is intercepting the avoidance plane. Equations 12 and 17 work together to complete the three points conic-section (triangle). The expression of Eq. 17 is given by

$$\begin{aligned} VO_{P_\phi} &= \begin{bmatrix} {x_{Bvo}}^{\phi}(t_1) \\ {y_{Bvo}}^{\phi}(t_1) \end{bmatrix}, t_1 \epsilon R, \\ VO_{P_\phi} &= \begin{bmatrix} {x_{Bvo}}^{\phi}(t_2) \\ {y_{Bvo}}^{\phi}(t_2) \end{bmatrix}, t_2 \epsilon R \end{aligned} \tag{17}$$

Fig. 4 Incomplete parabolic



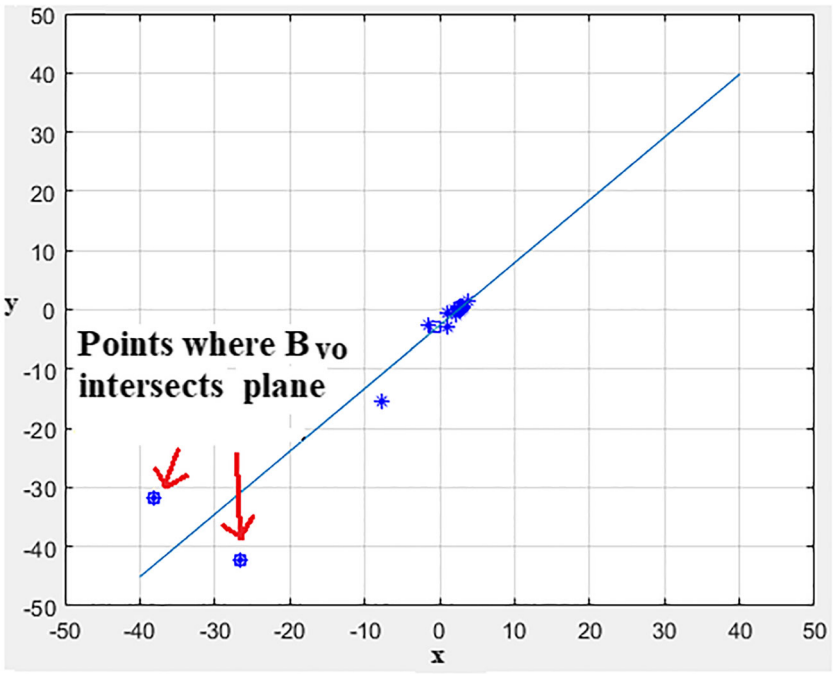
3.1.4 Incomplete Parabolic and Hyperbolic Shape

The previous section introduced a way to remove the error side of the hyperbolic. This section, presents another limitation on the original conic-section equation.

The original equation is derived to only consider the intersection of the **VO** cone's generating lines (lines from apex to base circle) with the avoidance plane. It does not consider the base circle, $\mathbf{B}_{vo}$, intersecting with the avoidance plane. As a result, Eq. 12 produces incomplete parabolic and hyperbolic shape. Figure 4 shows the incomplete parabolic shape. The algorithm may choose avoidance velocity inside the conic-section.

The parabolic and hyperbolic can be completed by finding the two points where the $\mathbf{B}_{vo}$ intersects the avoidance plane. This can be achieved using Eq. 17. Figure 5 shows the completed parabolic shape.

Fig. 5 Complete parabolic



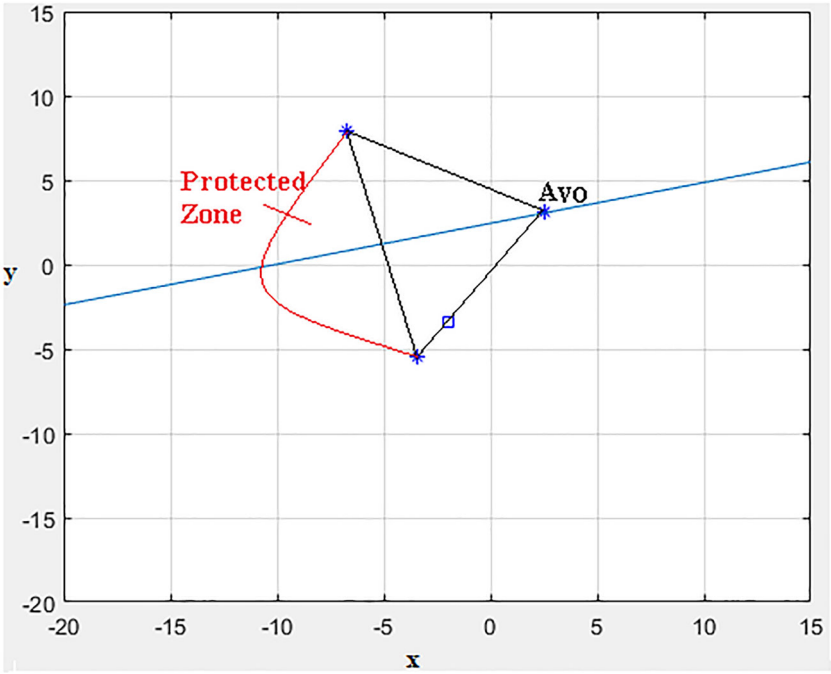


Fig. 6 When obstacle is too close, wrong avoidance velocity is chosen

3.1.5 Simplifying Spherical Protected Zone

As shown in Fig. 1b, the **VO** cone is defined after removing approximately half of the spherical protected zone. This simplifies the mathematics involved in defining the **VO** cone. However, this may result in choosing the wrong avoidance velocity as shown in Fig. 6. The proposed algorithm in [25] may choose an avoidance velocity inside part of the protected zone that has been removed. Eventually, this leads the occurrence of a collision.

This error occurs on the parabolic, hyperbolic and triangle shape. Figure 7 shows two lines, where the right side line (**VOlineR**) and the left side line (**VOlineL**) are defined to resolve this issue. The avoidance velocity is now chosen outside of the conic-section and the boundary between the two lines. The two lines are defined using $\mathbf{A}_{vo}{}^{\phi}$ and Eq. 17.

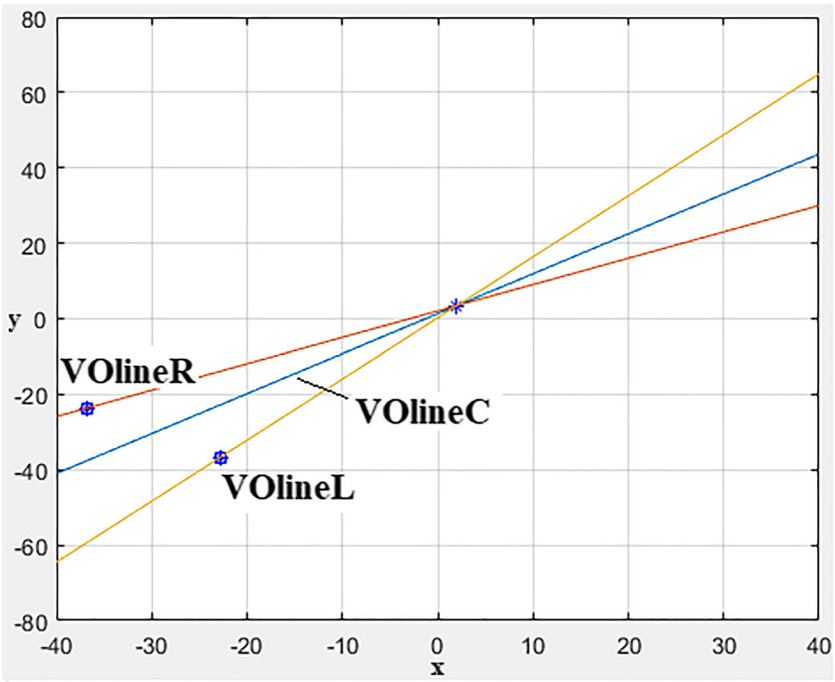


Fig. 7 Two lines for parabolic, hyperbolic and triangle shape

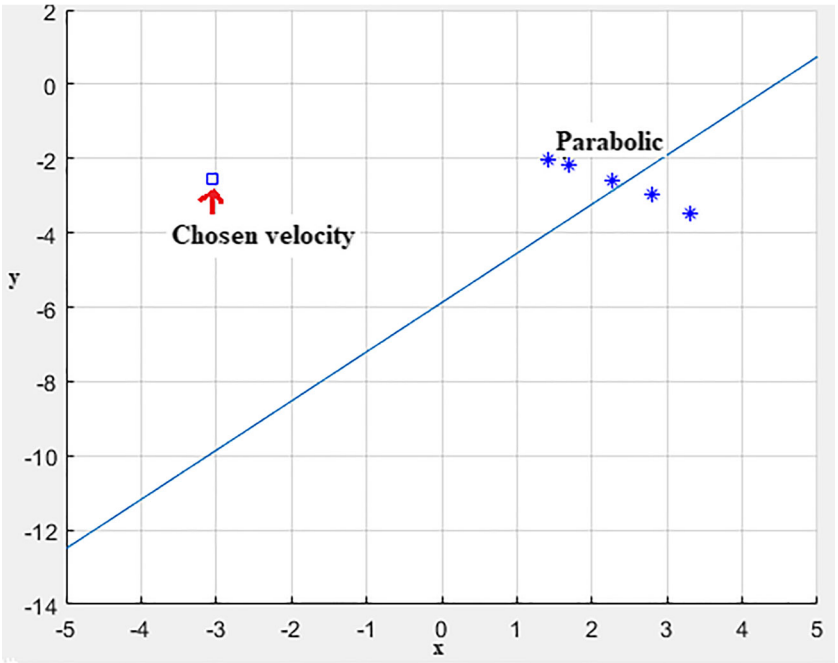


Fig. 8 Choosing wrong avoidance velocity in a parabolic shape

3.1.6 Problem in Parabolic and Hyperbolic Shape

When the avoiding vehicle is close to the obstacles protected zone, it results in the parabolic or hyperbolic shape shown in Fig. 8. The parabolic curve is smaller and may result in the collision avoidance strategy choosing the wrong avoidance velocity.

The length of the parabolic and hyperbolic should be increased to prevent choosing the wrong avoidance velocity. The approach is to create an virtual sphere which is an expansion of the original obstacle. As shown in Fig. 9, the own vehicle is located at P_A, while the obstacle is referred to a sphere S_{pz} with a radius of r_{pz} located at P_B. The virtual sphere $S_{pz}^{'}$ is located at $P_B^{'}$ and its radius is $\tau \times r_{pz}$, where the time to prevent the parabolic issue is defined as $\tau \geq 1.5$. The distance between the own vehicle P_A to the virtual obstacle $P_B^{'}$ is given by

$$|P_B^{'} P_A| = \tau |P_B - P_A|, \tag{18}$$

where is obtained from the relation equation $\frac{|P_B - P_A|}{|P_B^{'} P_A|} = \frac{r_{pz}}{\tau r_{pz}}$. Finding a new protected zone further away from the avoiding vehicle will resolve this issue completely.

This section introduces the different type of shapes on the avoidance plane. Implementing the original **VO** method and the proposed improvement shows eight types of conic-section. They are the circle, eclipse, parabolic, hyperbolic, single point, single line, triangle and blank. The blank conic-section means that the avoidance plane did not intersect the three-dimensional **VO** cone. All velocities on the avoidance plane can be selected as avoidance maneuver.

4 Avoiding Static Obstacles Using Pyramid Cone

It is essential for the collision avoidance algorithm to achieve avoidance with both dynamic and static obstacles. Different shapes and sizes of static obstacles may be presented in the

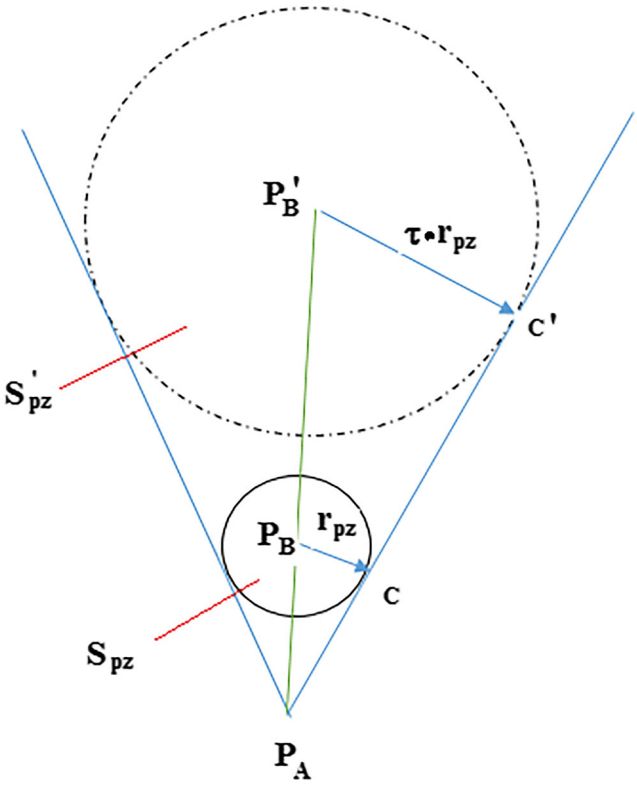
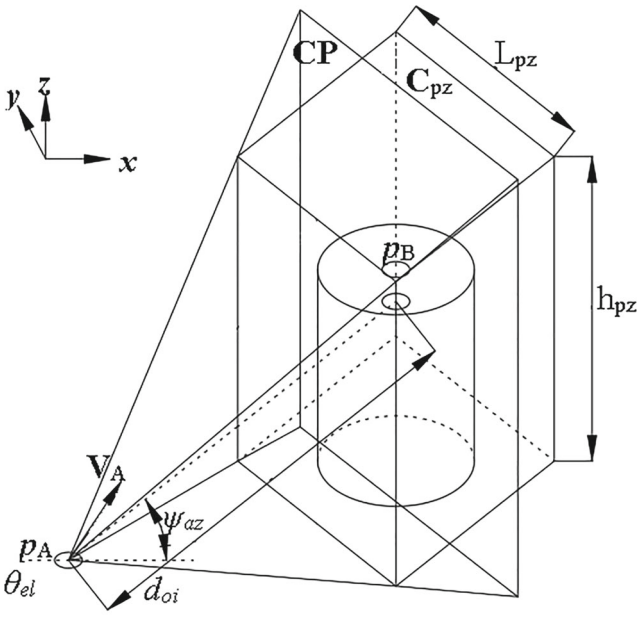
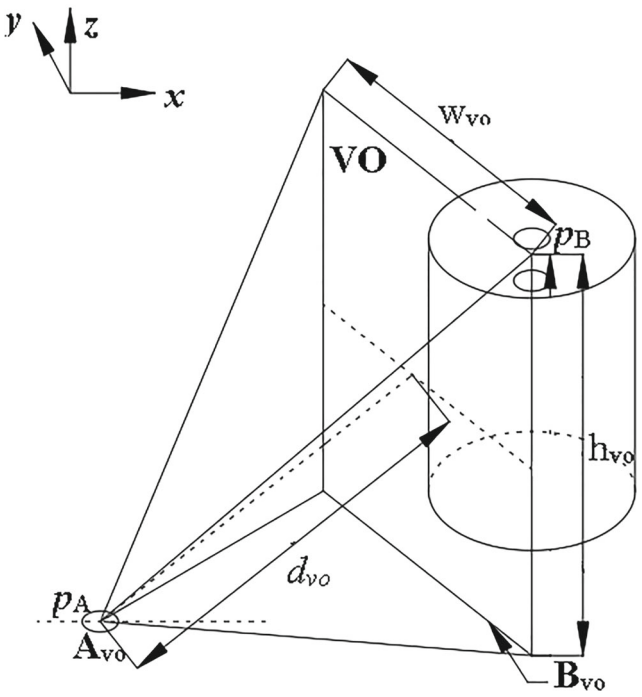


Fig. 9 Expanding S_{pz}

3D environment, such as tall buildings or floating obstacles. The collision avoidance algorithm must be capable of handling all sorts of obstacles to be considered ready for use in urban, civilian or warfare environment. The previous section enhances the original three-dimensional velocity obstacle. However, it can only be applied on moving obstacles. This section builds on the original three-dimensional **VO** method [24, 25] to attain collision avoidance with static obstacles.



(a) Three-dimensional collision **VO** pyramid



(b) Three-dimensional velocity obstacle **VO** pyramid

Fig. 10 Pyramid cone of dealing with cylinder obstacle

4.1 Velocity Obstacle Pyramid

The work presented in literature [24, 25], is unable to handle variety in shape of static obstacles (such as tall buildings). The only shape possible for obstacles is spherical. This paper presents a unique approach to solve this problem. The basic idea is to use a tall square shape, C_{pz}, to establish the protected zone in the airspace, where a cylinder obstacle is considered. The C_{pz} square base connects to the ground level. This idea offers the means to avoid tall static obstacles. At the same time, it also prevents the vehicle from travelling under any floating obstacles. With the rectangular side of the C_{pz} facing the vehicle, the orientation of the C_{pz} changes with respect to the position of the avoiding vehicle.

Figure 10 depicts the encounter between an avoiding vehicle at position p_A and a tall cylinder shape static obstacle at position p_B. The C_{pz} is defined around the static obstacle. The square base is defined at equal distance around the center of the static obstacle. The length of the square base, L_{pz}, and the height of the C_{pz}, h_{pz}, are predefined parameters. Here, h_{pz} must be greater than the height of the static obstacle, while the L_{pz} must be greater than the largest length of any obstacles for the C_{pz} to be sufficiently large to ignore the orientation of the any types of obstacle. The collision pyramid **CP** is formed by accumulating the relative velocity extension between the avoiding vehicle and static obstacle. With $\overrightarrow{V_B} = 0$, the velocity obstacle **VO** pyramid is identical to the **CP**. The **VO** pyramid is shown in Fig. 10b to illustrate a clearer view of the pyramid.

4.2 VO Pyramid Cone on Avoidance Plane

The avoidance plane can be used on the **VO** pyramid for finding avoidance velocity. It is any plane in which lies the avoiding vehicle. The intersection between the avoidance plane and the **VO** pyramid produces the conic-section. The avoidance velocity must be outside the conic-section. Therefore, we have to give the expression of the coinic-section mathematically.

In order to express the conic-section on each avoidance plane, the **VO** pyramid must first be described in the three-dimensional space. As shown in Fig. 11, the **VO** pyramid is represented by its apex $\mathbf{A}_{vo}$, base rectangle, $\mathbf{B}_{vo}$, and center on base rectangle, $\mathbf{C}_{vo}$. It should be noticed that the rectangle form of the base $\mathbf{B}_{vo}$ is further described by the four lines (L_1, L_2, L_3, and L_4).

The mathematical expressions of $\mathbf{A}_{vo}$, B_{vo} and $\mathbf{C}_{vo}$ are presented by

$$\boldsymbol{A}_{vo} = VO_{P_\phi} = \begin{bmatrix} x_A \\ y_A \\ z_A \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{19}$$

$$B_{vo} = \begin{bmatrix} x_{B_{vo}} \\ y_{B_{vo}} \\ z_{B_{vo}} \end{bmatrix} = \boldsymbol{R}_{\theta_{el}|\psi_{az}} \begin{bmatrix} x_o \\ y_o \\ z_o \end{bmatrix} \tag{20}$$

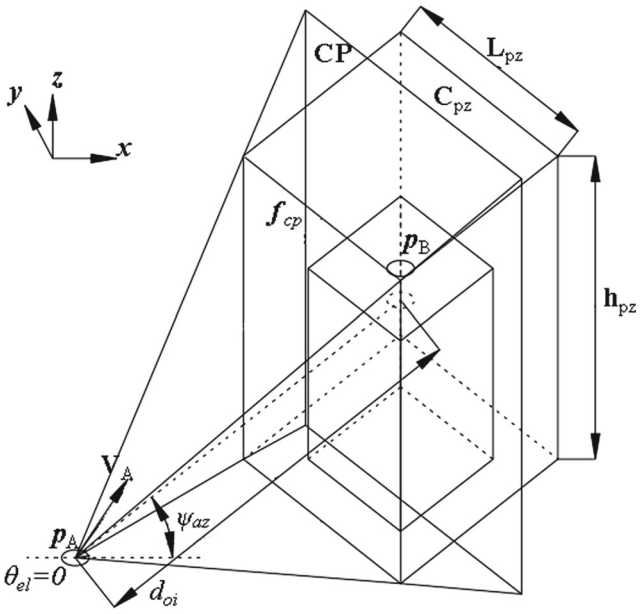
$$\boldsymbol{C}_{vo} = \boldsymbol{R}_{\theta_{el}|\psi_{az}} \begin{bmatrix} d_{vo} \\ 0 \\ 0 \end{bmatrix} \tag{21}$$

where the translation matrix $\boldsymbol{R}_{\theta_{el}|\psi_{az}}$ is given by Eq. 6.

Before translating the vector $[x_o \quad y_o \quad z_o]'$, it is given based on the UAV body as the frame reference, that is

$$\begin{bmatrix} x_o \\ y_o \\ z_o \end{bmatrix} = \begin{cases} \begin{bmatrix} d_{vo} \\ -\frac{Length\ of\ Line1}{2} \le y_o \le \frac{Length\ of\ Line1}{2} \\ \frac{Length\ of\ Line2}{2} \end{bmatrix} : Line1 \\ \begin{bmatrix} d_{vo} \\ -\frac{Length\ of\ Line1}{2} \\ -\frac{Length\ of\ Line2}{2} \le z_o \le \frac{Length\ of\ Line2}{2} \end{bmatrix} : Line2 \\ \begin{bmatrix} d_{vo} \\ -\frac{Length\ of\ Line1}{2} \le y_o \le \frac{Length\ of\ Line1}{2} \\ -\frac{Length\ of\ Line2}{2} \end{bmatrix} : Line3 \\ \begin{bmatrix} d_{vo} \\ \frac{Length\ of\ Line1}{2} \\ -\frac{Length\ of\ Line2}{2} \le z_o \le \frac{Length\ of\ Line2}{2} \end{bmatrix} : Line4 \end{cases} \tag{22}$$

Using the mathematical description of the **VO** pyramid, the conic-section can be determined. One alternative is to virtually rotate the **VO** pyramid about the x-axis. In this case, virtual rotation of only the $\mathbf{B}_{vo}$ will assist



(a) Three-dimensional collision **VO** pyramid

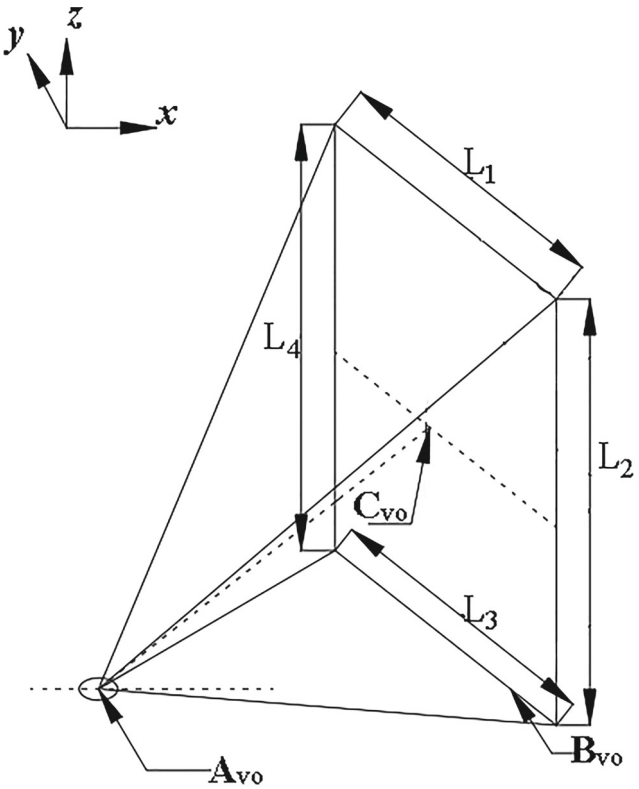
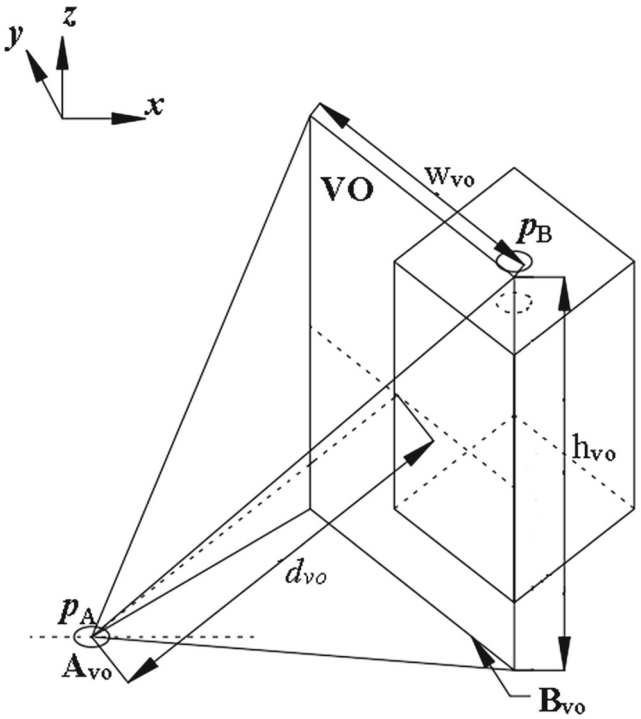


Fig. 11 Simplified **VO** pyramid



(b) Three-dimensional velocity obstacle **VO** pyramid

Fig. 12 Pyramid cone of dealing with square obstacle

in simplifying the expression for the conic-section. The rotation is described by

$$\boldsymbol{A}_{vo}{}^{\phi} = \begin{bmatrix} x_{A_{vo}}{}^{\phi} \\ y_{A_{vo}}{}^{\phi} \\ z_{A_{vo}}{}^{\phi} \end{bmatrix} = \boldsymbol{R}_{P_\phi} \boldsymbol{A}_{vo},$$

$$\boldsymbol{B}_{vo}{}^{\phi} = \begin{bmatrix} x_{B_{vo}}{}^{\phi} \\ y_{B_{vo}}{}^{\phi} \\ z_{B_{vo}}{}^{\phi} \end{bmatrix} = \boldsymbol{R}_{P_\phi} \boldsymbol{B}_{vo} \tag{23}$$

where,

$$\boldsymbol{R}_{P_\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\phi & sin\phi \\ 0 & -sin\phi & cos\phi \end{bmatrix}. \tag{24}$$

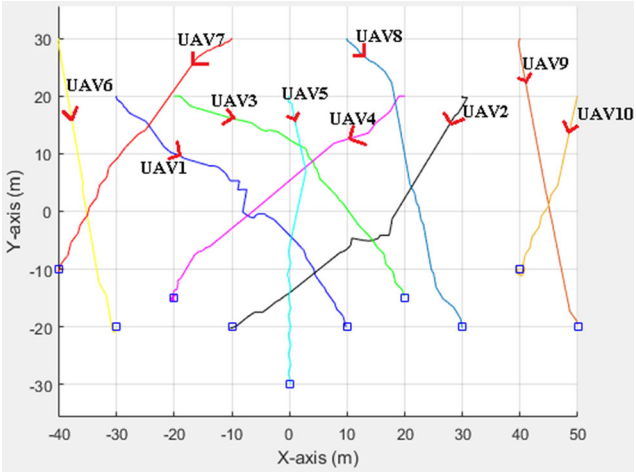
The conic-section will then be the point where the $\mathbf{B}_{vo}{}^{\phi}$ intersects with the xy plane (P_0). The intersection is described by assigning $z_{B_{vo}}{}^{\phi} = 0$ and the solution of (x_{vo}, y_{vo}) is obtained. The designed velocity must be outside the intersection points.

4.3 Extension of Velocity Obstacle Pyramid

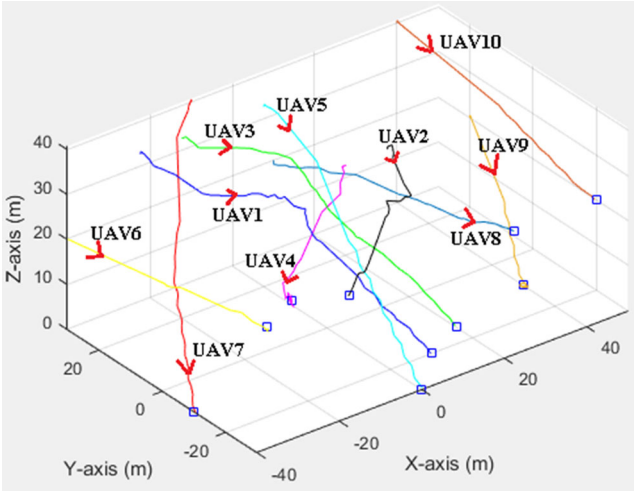
The previous work is focused on an approach to handle cylinder obstacles. If the obstacles are different, for example square obstacles, the proposed approach can still work well. Consider a square shape of obstacles. Taking a similar idea as in Section 4.1, a tall cube, C_{pz}, is used to protect the square obstacle. The C_{pz} base sits to the ground. Figure 12 shows a pyramid generation which is located at between an avoiding vehicle at position p_A and a square shape obstacle at position p_B. The cube C_{pz} is an expansion to the square obstacle. The cube base is defined at equal distance around the center of the square obstacle. The length of the cube base, L_{pz}, and the height of the C_{pz}, h_{pz}, are predefined parameters. It should be noticed that h_{pz} must be chosen to be a number greater than the height of the square obstacle, while the L_{pz} must be chosen to a number greater than the largest diagonal length of the square obstacle for the C_{pz} to be sufficiently large to ignore the orientation of the obstacle.

Table 1 Simulation parameters

Parameter	Value	Unit
Max distance of other vehicle for performing collision consideration	55	m
Max distance to nearest point of obstacle for performing collision consideration	55	m
Range of velocity	0 to 4	m/s
UAV lowest height	0	m
UAV highest height	40	m



(a)



(b)

Fig. 13 Flight paths of ten UAVs crossing encounter. **a** Two-dimensional view, **b** Three-dimensional view

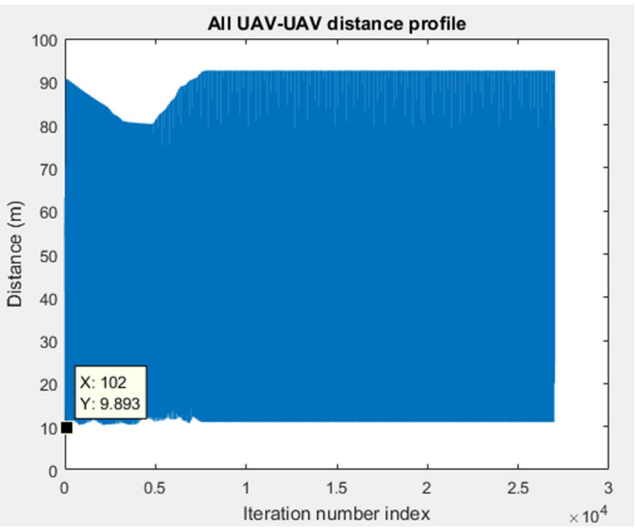


Fig. 14 Distance between 10 UAVs in crossing encounter

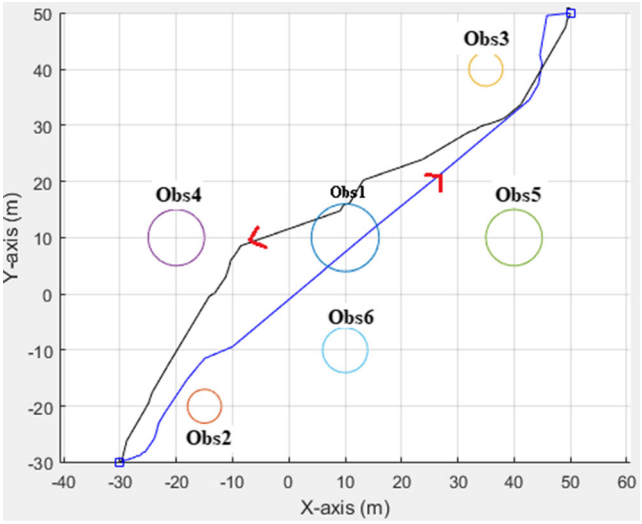
The collision pyramid **CP** is formed by accumulating the relative velocity extension between the avoiding vehicle and static obstacle. Since the velocity of the static obstacle is zero, the velocity obstacle **VO** pyramid is identical to the **CP**. Figure 12b gives a further illustration to the **VO** pyramid.

After the VO pyramid is obtained, the same VO pyramid cone on avoidance plane can be generated as in Section 4.2.

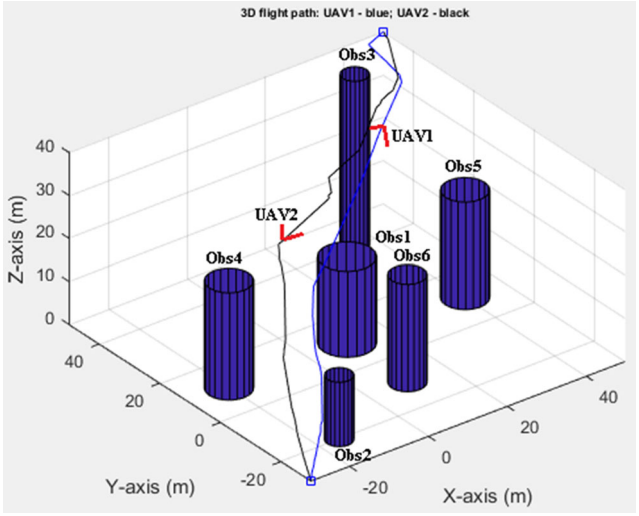
5 Simulation Results

To evaluate the capability of the method in this paper, simulation has been conducted. The tested hardware is desktop computer,Acer M670G. The three-dimensional **VO** method developed in this paper is implemented in a MATLAB simulation. The simulation involves multiple moving and static obstacles.

Each UAV is assumed to be programmed with the proposed algorithm and it can obtain the other UAV information which includes position and speed. From the proposed algorithm, it should be noticed that each UAV independently produces avoiding velocity control based on its own detection of obstacles. Each vehicle is depicted as a point source in the simulation, moves at a constant time step. The radius of the protected zone for each vehicle is 10m. According to the proposed algorithm, the avoiding velocity control is updated at each time step. The simulation

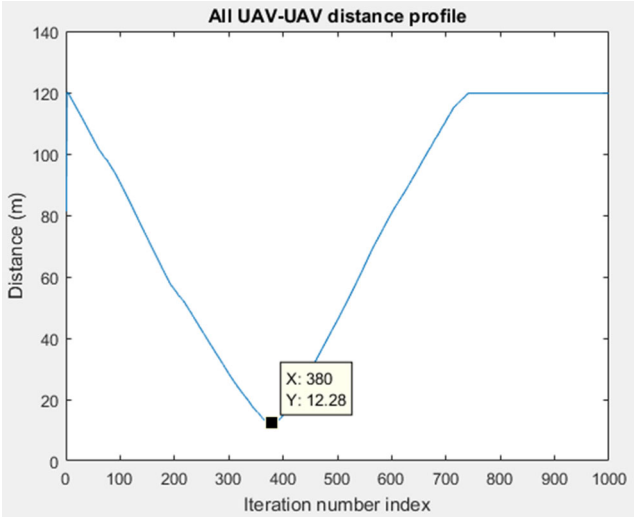


(a)

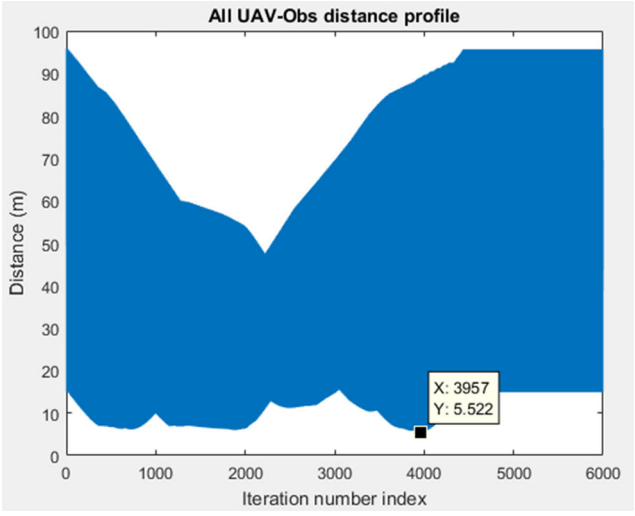


(b)

Fig. 15 Trajectory results of 2 UAVs crossing encounter with 6 static obstacles. **a** Two-dimensional view, **b** Three-dimensional view



(a)



(b)

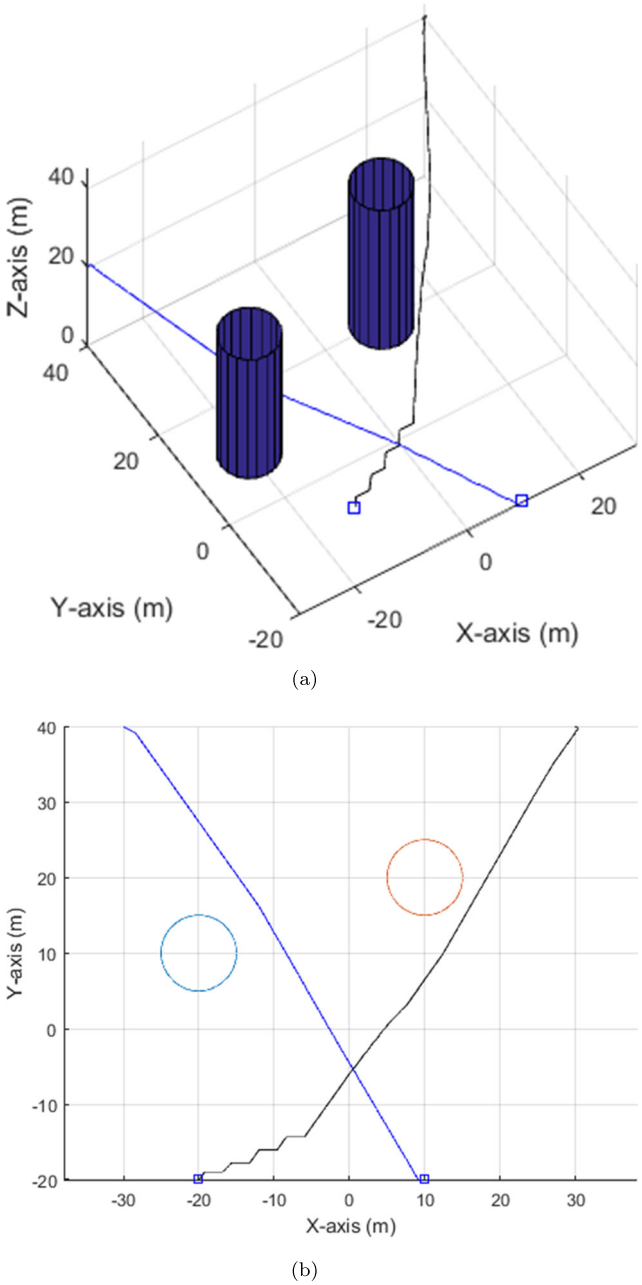
Fig. 16 Distance between 2 UAVs and 6 static obstacles in crossing encounter. **a** Vehicle-vehicle **b** Vehicle-obstacle

parameters are given in Table 1. In order to show the effectiveness of the proposed algorithm, several cases are simulated.

In the first case, a crossing encounter of ten UAVs is tested. This aims to evaluate the potential of the improved **VO** cone in dense environment. The flight path of each UAV is shown in Fig. 13. The distance profile between vehicle-vehicle is presented in Fig. 14. No collision is detected throughout the simulation. Each vehicle maintains a safe distance of 9.893 meters which is bigger than the radius 5m of the protected zone S_{pz}.

In the second case, a simulation involves two UAVs with multiple obstacles in their path. Its trajectory results are given in Fig. 15. Figure 16 presents both the distance between each vehicle and the distance between vehicle and static obstacles. The vehicles are able to perform collision avoidance with nearest distance of 12.28 meters between the vehicles. The nearest distance between vehicle and static obstacles is 5.522 meters.

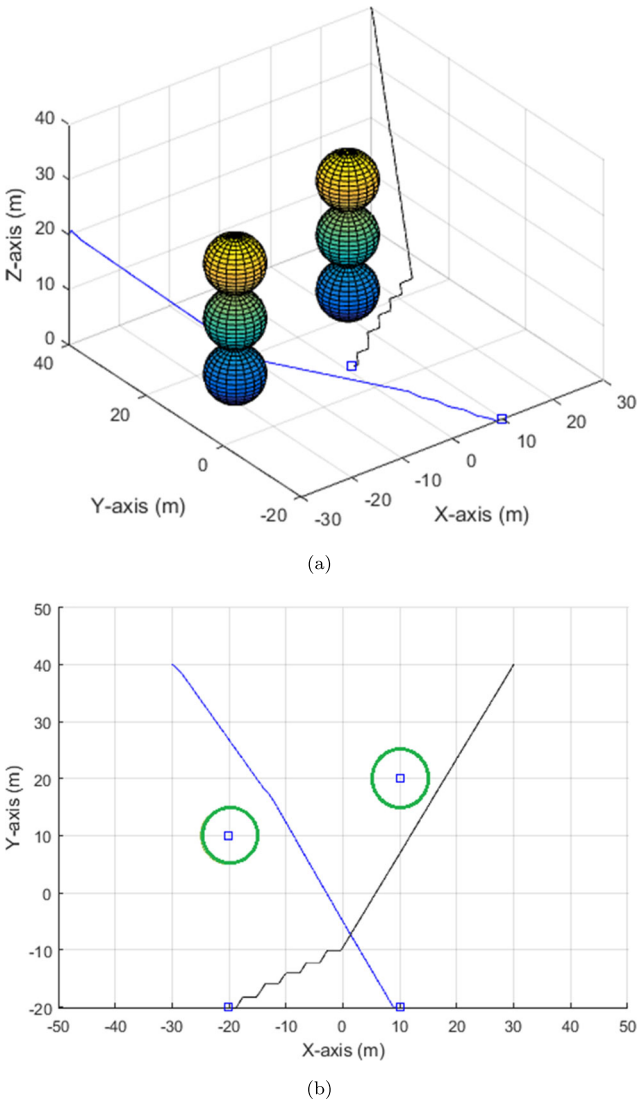
In the third case, we intent to compare the proposed algorithm with the result of [24, 25]. In this simulation, it involves



(a)

(b)

Fig. 17 Trajectory results of 2 UAVs crossing encounter with two static obstacles using our algorithm. **a** Three-dimensional veiw, **b** Two-dimensional view

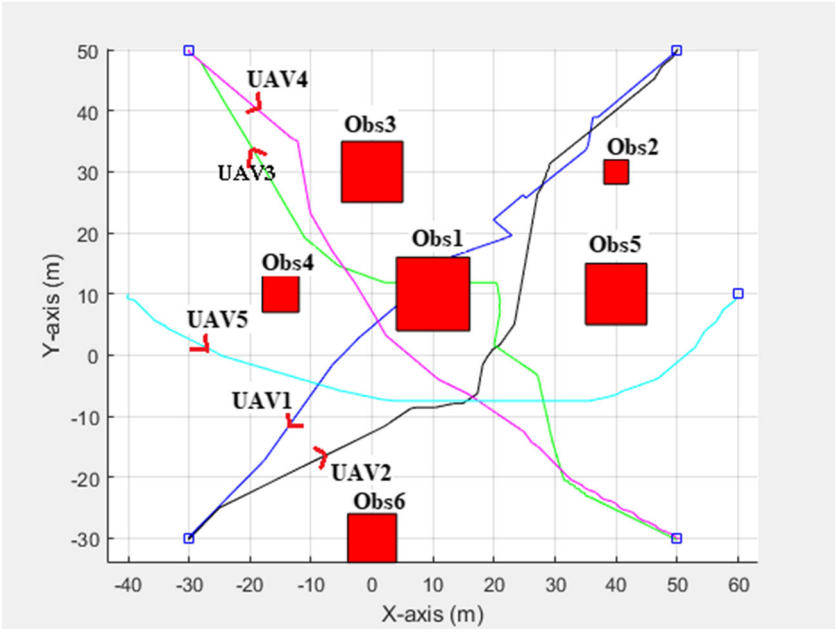


(a)

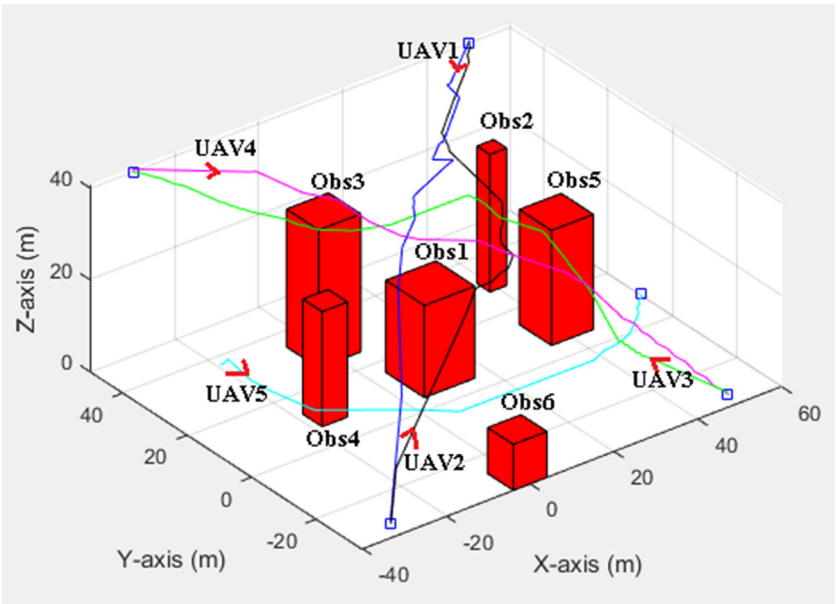
(b)

Fig. 18 Trajectory results of 2 UAVs crossing encounter with two static obstacles using the proposed algorithm [24, 25]. **a** Three-dimensional view, **b** Two-dimensional view

two UAVs with two obstacles in their path. In this scenario, the proposed algorithm can be used directly to avoid both obstacles and other moving UAVs. The results are given in Fig. 17. It is observed that the vehicles are able to perform collision avoidance and the running time in MATLAB is 42.486s. However, when using the algorithm proposed in [24, 25], it only handles the spherical ball and cannot deal with the cylinder form (obstacle) directly. To overcome this drawback, one way is to use multi-spheres to approximate the cylinder form. In this way, each cylinder is approximated by three spheres as shown in Fig. 18, and thus the proposed algorithm in [24, 25] can be used. The results show that the vehicles can avoid both moving vehicle and static obstacles, as shown in Fig. 18. However, the running time in MATLAB is 203.424s. It should be noticed that both simulations use the same iteration number of 140. This implies that the



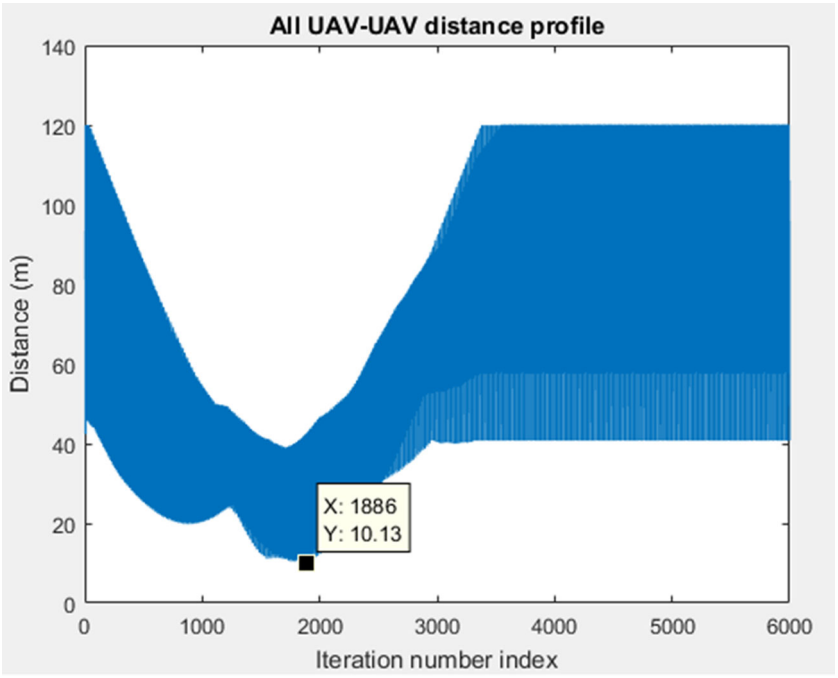
(a)



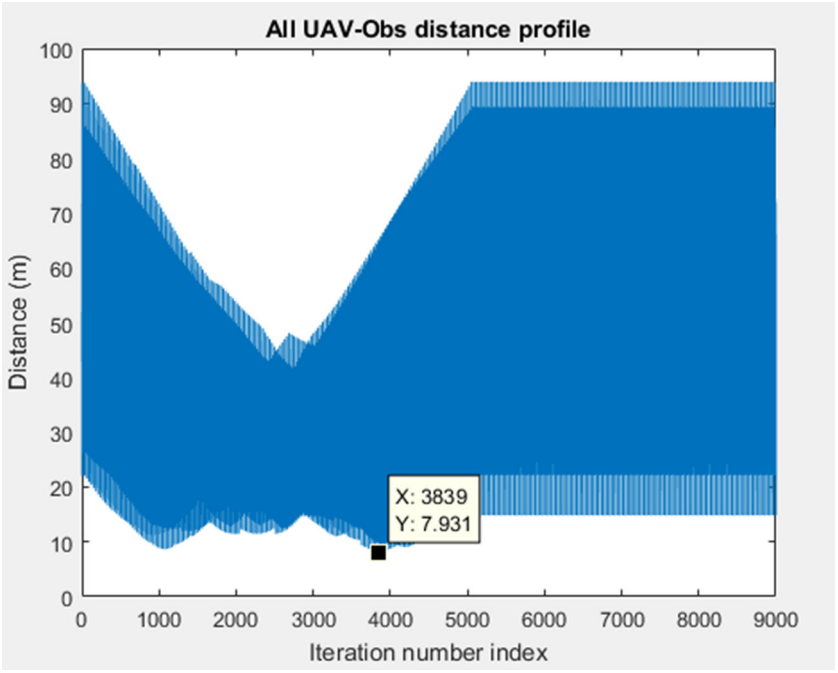
(b)

Fig. 19 Trajectory results of five UAVs head on encounter with six tall static obstacles. **a** Two-dimensional view, **b** Three-dimensional view

proposed approach on average takes 0.3035s per iteration, while the approach used in [24, 25] on average takes 1.4530s per iteration. Obviously, by comparison with our algorithm, the proposed algorithm in [24, 25] takes a longer time for collision avoidance and is not suitable for dealing with the static obstacles, especially for higher buildings (obstacles). This is why in this paper we extend the result of [24, 25] to cover the case where static obstacles are not spherical.



(a)



(a) Vehicle-vehicle (b) Vehicle-obstacle

Fig. 20 Distance between five UAVs and six tall static obstacles in head on encounter. **a** Vehicle-vehicle, **b** Vehicle-obstacle

In the fourth case, we intent to show that the proposed algorithm can handle the case where obstacles are square. There are five UAVs in a dense environment of six tall static obstacles. This aims to test the potential of the **VO** cone and **VO** pyramid concept working together in a different type of obstacles. The trajectory of each vehicle is presented in Fig. 19. The distance recorded throughout the simulation is shown in Fig. 20. No collision is detected. The closest distance between vehicles is 10.13 meters. The closest distance between the vehicle and six tall static obstacles is 7.931 meters.

6 Real-Life Flight Test

This section will present the actual flight test on what has been presented in this paper. The collision avoidance algorithm is implemented in C++ code which is connected to robot operating system (ROS).

6.1 Experimental Setup

The real-life flight test is carried out in a $6m \times 6m \times 2m$ room as shown in Fig. 21. Figure 22 shows the components of the UAV. The collision avoidance algorithm is downloaded into the payload which is connected to the aircraft control board.

The ground control station (GCS) is used to communicate with the UAV. It consists of one notebook and one route with antenna as shown in Fig. 23. The operation is shown in Fig. 24. The ground control station (GCS) sends parameters and commands to the UAV. The on-board laser sensor is used to measure the position of the UAV, updating the information into the collision avoidance algorithm at a frequency of 5Hz. Then, the UAV uses the information to perform collision avoidance.

Figure 25 shows a simple illustration of the key parts in ROS node graph. The real ROS node graph is far more complex. The key idea is that node 1 published the laser information to topic 1. Node 2 which has subscribed to topic 1 will receive the data. The collision avoidance algorithm will calculate the avoidance position of the UAV and publisher to topic 2. The controller uses the information from topic 2 to drive the UAV.

Figure 26 shows a simple flowchart of the c++ program developed for the collision avoidance algorithm presented in this paper. The parameters of the real flight test are given in Table 2. The maximum height of the UAV is set at 1m.

6.2 Flight Test

The real-life flight test will examine the efficiency of the three-dimensional velocity obstacle method in avoiding static obstacles. The mission of the UAV is to fly to its destination without colliding with a static obstacle. The test scenario is shown in Fig. 27. The UAV first takes off and flies to the desired location. It has to avoid the static obstacle during its path. After avoiding the obstacle, it will land at the desired destination.

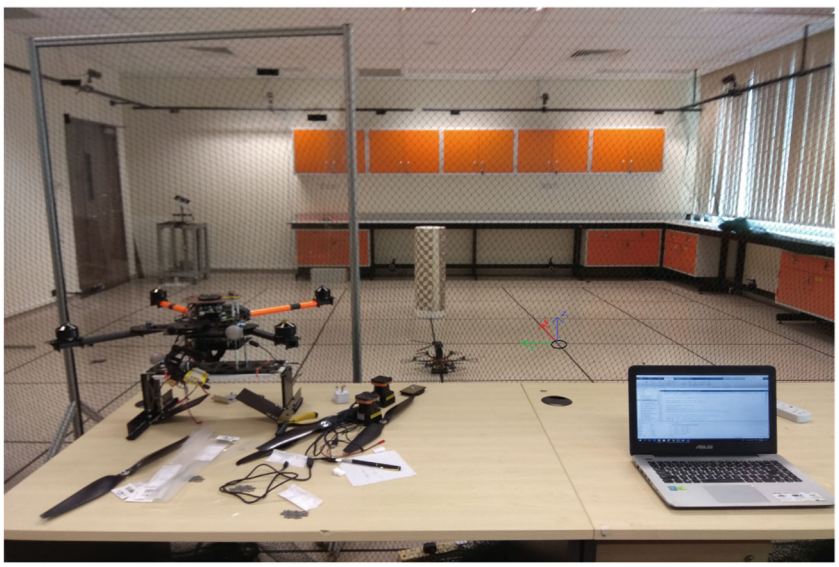
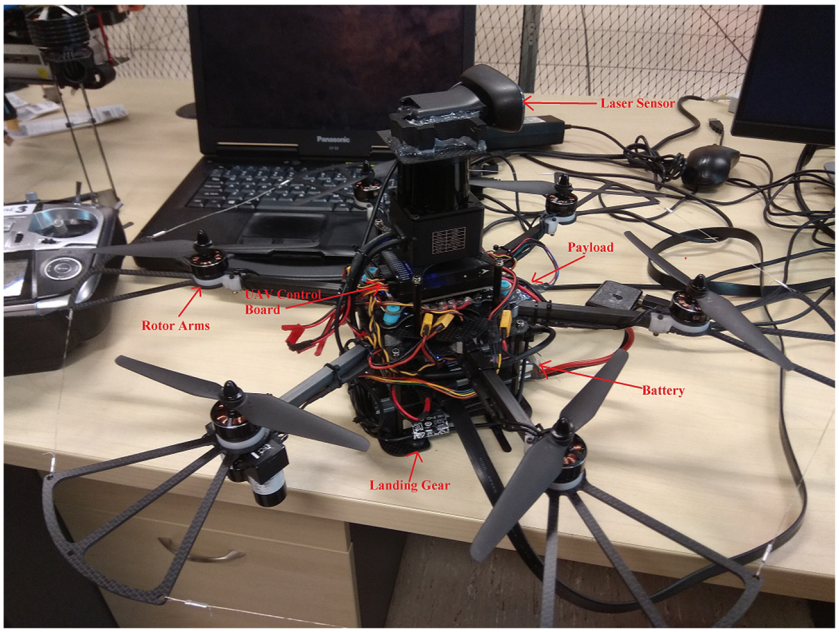
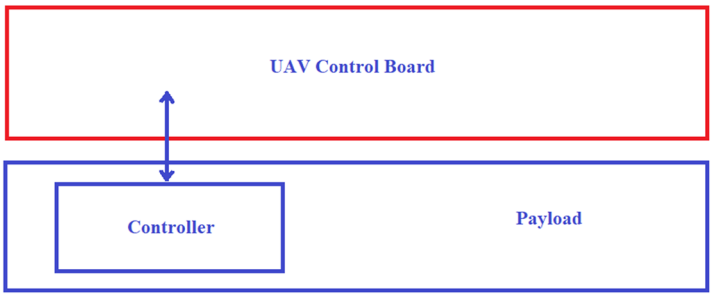


Fig. 21 Flight Test room

Fig. 22 UAV Testbed





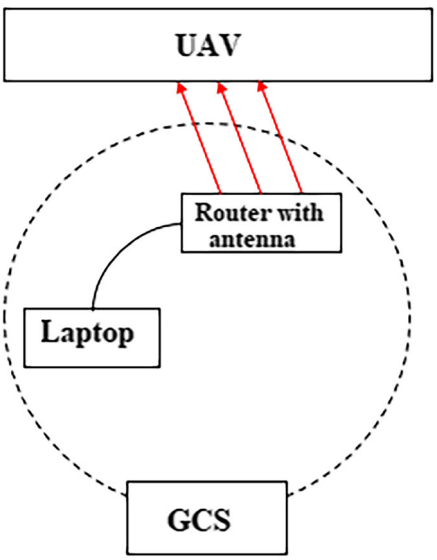


Fig. 23 Ground Control Station

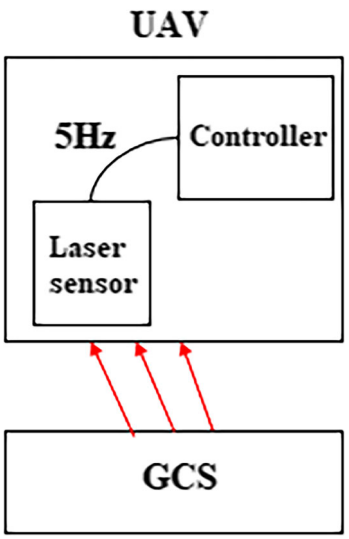


Fig. 24 Operation Concept

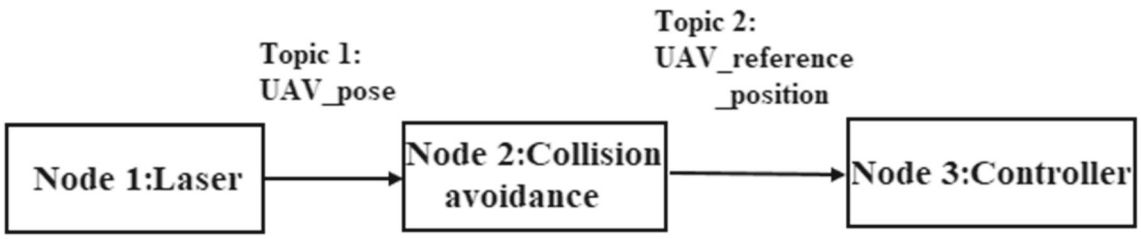


Fig. 25 Simple illustration of ROS node graph

Fig. 26 Flowchart of c++ program

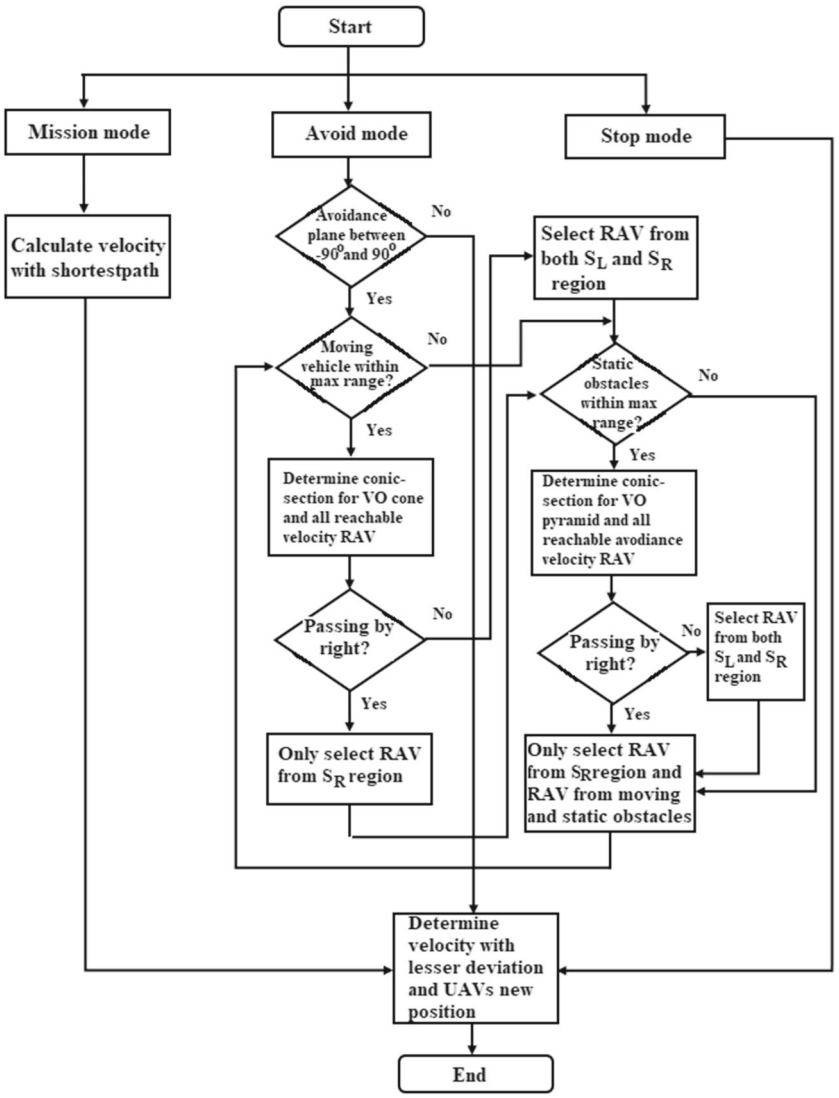


Table 2 Parameters used

Parameter	Value	Unit
Max distance to nearest point of obstacle for performing collision consideration	5	m
Max value of v to evaluate	1 to 1.2	m/s
Radius of vehicle	1	m
Max speed	1.2	m/s
Radius of protected zone for static obstacle	1	m

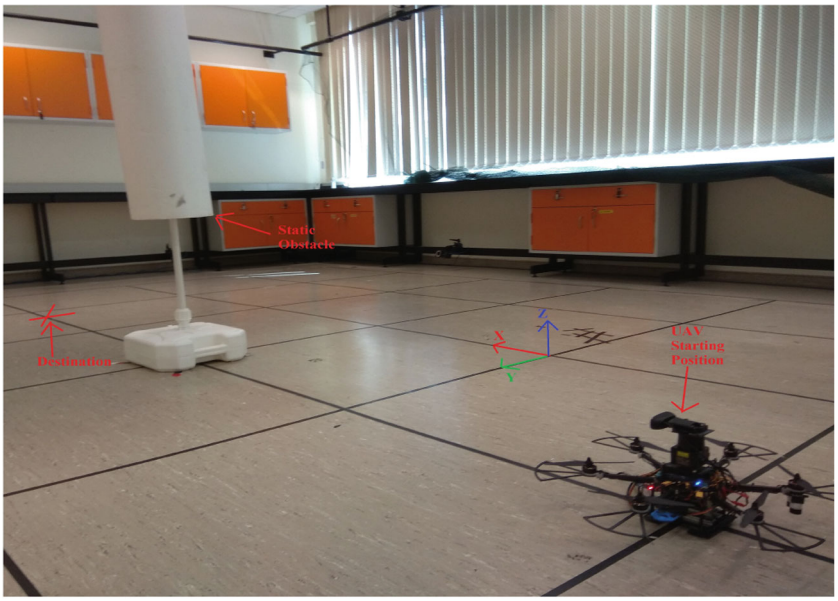
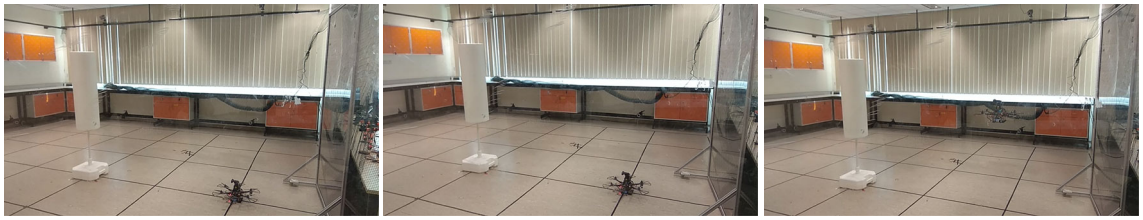


Fig. 27 Flight test setup with static obstacle

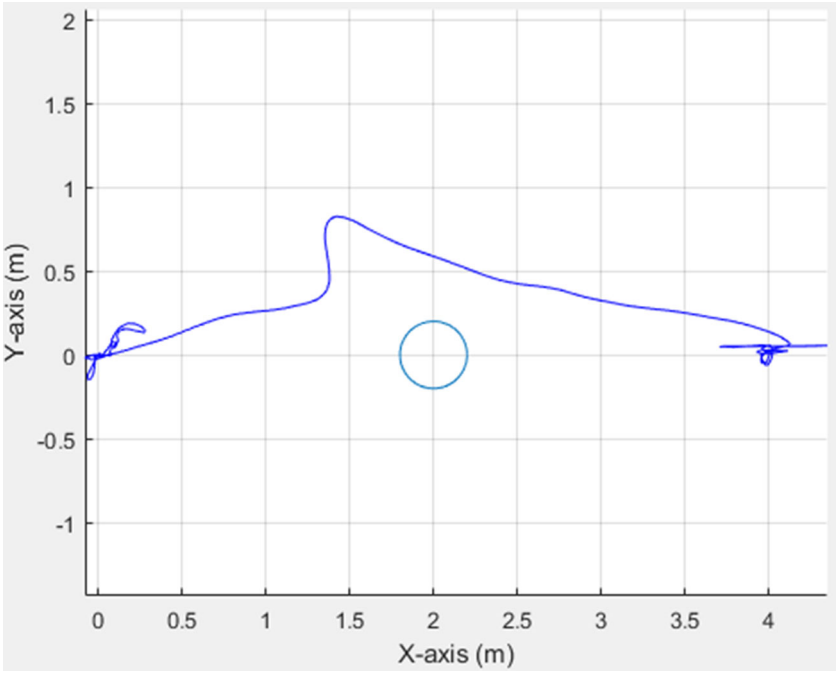


(a) 0s (b) 10s (c) 20s

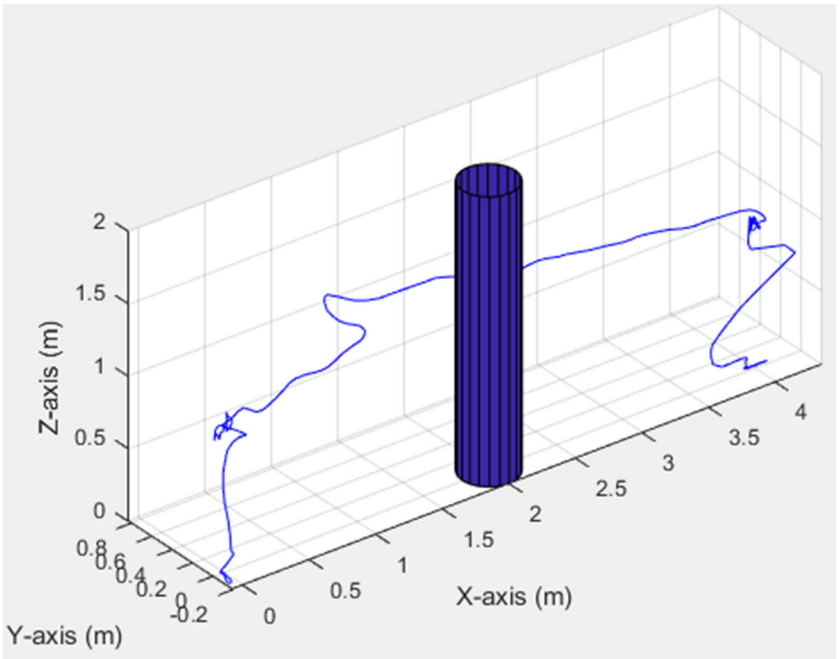
(d) 30s (e) 34s (f) 40s

(d) 50s (e) 55s (f) 60s

Fig. 28 Position of UAV at specific time



(a)



(b)

Fig. 29 Trajectory results of UAV. **a** 2-D flight path **b** 3-D flight path

Figure 28 shows the position of the UAV at specific time. The UAV was able to execute its mission and avoid the static obstacle placed in its path. The collision is prevented and it performs its mission successfully. Figure 29 shows the trajectory result of the UAV. The distance profile between the obstacle and UAV is shown in Fig. 30. The trajectory result shows that the UAV was able to avoid the static obstacle. The closest distance between the UAV and the obstacle is approximately 0.3521 meter.

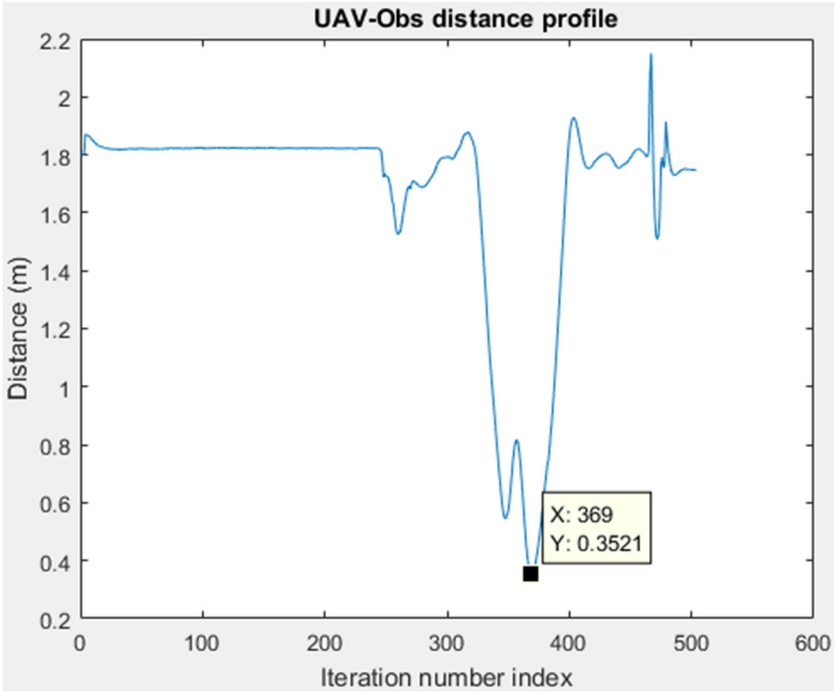


Fig. 30 Distance between obstacle and UAV

7 Conclusion

This paper has presented a 3-D velocity obstacle method to attain collision avoidance with multiple dynamic and static obstacles. The improvement to the existing **VO** cone has been proposed. A tall cylinder shape protected zone which can be used to approximate all kinds of static obstacles, is introduced for each static obstacles. The velocity obstacle pyramid is presented in dealing with 3-D obstacles.

Simulation has been conducted to evaluate the potential of the 3-D **VO** algorithm proposed in this paper. The proposed method is effective in collision avoidance with multiple dynamic and static obstacles. It is also successful in avoiding tall obstacles. Real-life flight test has been conducted to assess the algorithm capability in real-time situation. The algorithm has shown success in collision avoidance with static obstacle.

For future research, simulations on more realistic conditions will be considered (for example using 3D simulators such as Gazebo, V-REP or Webots, model real environment). Further real flight experiment is also required to determine the effectiveness and efficiency of the algorithm in avoiding multiple moving and static obstacles.

References

1. Huang, S., Teo, R., Liu, W., Dymkou, S.M.: Distributed cooperative UAV loss detection and auto-replacement protocol with guaranteed properties. J. Intell. Robot. Syst. **93**(1), 303–316 (2019)
2. Huang, S., Teo, R., Liu, W., Dymkou, S.M.: Agent model for multi-UAV control via protocol designs. Int. J. Intell. Comput. Cybern. **10**(4), 412–429 (2017)
3. Zhang, G., Shang, B., Chen, Y.Q., Moyes, H.: SmartCaveDrone: 3D cave mapping using UAVs as robotic co-archaeologists. In: 2017 International Conference on Unmanned Aircraft Systems, pp. 1052–1057. IEEE, Miami (2017)
4. Zhang, G., Chen, Y.Q., Moyes, H.: Optimal 3D reconstruction of caves using small unmanned aerial systems and GGB-D cameras. In: 2018 International Conference on Unmanned Aircraft Systems (ICUAS), pp. 410–415. IEEE, Dallas (2018)
5. Chevet, T., Maniu, C.S., Vlad, C., Zhang, Y.: Voronoi-based UAVs formation deployment and reconfiguration using MPC techniques. In: 2018 International Conference on Unmanned Aircraft Systems (ICUAS), pp. 9–14. IEEE, Dallas (2018)
6. Yu, X., Zhang, Y.: Sense and avoid technologies with applications to unmanned aircraft systems: Review and prospects. Progress Aeros. Sci. **74**, 152–166 (2015)
7. Yu, X., Zhou, X., Zhang, Y.: Collision-free trajectory generation and tracking for UAVs using Markov decision process in a cluttered environment. J. Intell. Robot. Syst. **93**(1–2), 17–32 (2019)
8. Zhou, X., Yu, X., Peng, X.: UAV collision avoidance based on varying cells strategy. IEEE Transactions on Aerospace and Electronic Systems, to be published, https://doi.org/10.1109/TAES.2018.2875556
9. Preiss, J.A., Hönig, W., Ayanian, N., Sukhatme, G.S.: Downwash-aware trajectory planning for large quadrotor teams, arXiv:1704.04852v2 [cs.RO] (2017)
10. Manathara, J.G., Ghose D.: Reactive collision avoidance of multiple realistic UAVs. Aircraft Eng. Aerosp. Technol. **83**(6), 388–396 (2011)
11. Lalish, E., Morgansen, K.A.: Distributed reactive collision avoidance. Auton. Robots **32**(3), 207–226 (2012)
12. Lao, M., Tang, J.: Cooperative multi-UAV collision avoidance based on distributed dynamic optimization and causal analysis. Appl. Sci. **7**(1), 83 (2017)

13. Migliaccio, G., Mengali, G., Galatolo, R.: Conflict detection and resolution algorithms for UAVs collision avoidance. Aeronaut. J. **118**(1205), 828–842 (2016)
14. Han, S.C., Bang, H., Yoo, C.S.: Proportional navigation-based collision avoidance for UAVs. Int. J. Control Autom. Syst. **7**(4), 553–565 (2009)
15. Vera, S., Cobano, J.A., Heredia, G., Ollero, A.: Collision avoidance for multiple UAVs using rolling-horizon policy. J. Intell. Robot. Syst. **84**(14), 387–396 (2016)
16. Budiyanto, A., Cahyadi, A., Adji, T.B., Wahyunggoro, O., Grafika, J.: UAV obstacle avoidance using potential field under dynamic environment. In: International Conference on Control, Electronics, Renewable Energy Communications, pp. 187–192 (2015)
17. Mac, T.T., Copot, C., Hernandez, A., De Keyser, R.: Improved potential field method for unknown obstacle avoidance using UAV in indoor environment. In: SAMI 2016 - IEEE 14th International Symposium Applied Machine Intelligence Informatics, pp. 345–350 (2016)
18. Huang, S., Swee, R., Teo, H., Liu, W., Dymkou, S.M.: Distributed cooperative collision avoidance control and implementation for multi-unmanned aerial vehicles. In: Proc. of 2017 11th Asian Control Conference (ASCC), pp. 222–227. Gold Coast (2017)
19. Tan, C.Y., Huang, S., Tan, K.K., Teo, S.H.R.: Three-dimensional cooperative collision avoidance design on unmanned aerial vehicle. In: 2018 International Conference on Unmanned Aircraft Systems, pp. 522–530. Dallas (2018)
20. Jenie, Y.I., Van Kampen, E.-J., de Visser, C.C., Ellerbroek, J., Hoekstra, J.M.: Selective velocity obstacle method for deconflicting maneuvers applied to unmanned aerial vehicles. J. Guid. Control Dyn., **38**(6) (2015)
21. Kuwata, Y., Wolf, M.T., Zarzhitsky, D., Huntsberger, T.L.: Safe maritime navigation with COLREGS using velocity obstacles, 4728–4734 (2011)
22. Fiorini, P., Shiller, Z.: Motion planning in dynamic environments using velocity obstacles. I. J. Robot. Res **17**, 760–772 (1998)
23. Chakravarthy, A., Ghose, D.: Obstacle avoidance in a dynamic environment: a collision cone approach. IEEE Trans. Syst. Man Cybern. Part A: Syst. Humans **28**(5), 562–574 (1998)
24. Jenie, Y.I., van Kampen, E., de Visser, C.C., Ellerbroek, J., Hoekstra, J.M.: Three-dimensional velocity obstacle method for UAV deconflicting maneuvers. AIAA Guidance, Navigation and Control Conference, 116 (2016)
25. Jenie, Y.I., van Kampen, E.-J., de Visser, C.C., Ellerbroek, J., Hoekstra, J.M.: Three-dimensional velocity obstacle method for uncoordinated avoidance maneuvers of unmanned aerial vehicles. J. Guid. Control Dyn. **39**(10), 2312–2323 (2016)

Chee Yong Tan received his B.Eng. in Electrical and Comupter Engineering (2018) in National University of Singapore. He is currently work on National Lab. Defence Science Organisation.

Sunan Huang received his Ph.D degree from Shanghai Jiao Tong University, 1994. From 1995 to 1997, he was a postdoctoral fellow in the Dept. of Electrical Engineering and Computer Sciences, University of California at Berkeley. During 1997-2012, he was a research fellow in the Dept. of Electrical and Computer Engineering, National University of Singapore. During 2013-2014, he was a visiting professor in Hangzhou Dianzi University. He is currently a senior research scientist in Temasek Laboratories, National University of Singapore.

Kok Kiong Tan received his B.Eng. in Electrical Engineering with honours in 1992 and PhD in 1995, all from the National University of Singapore. He is currently a Professor with the Department of Electrical and Computer Engineering, National University of Singapore. His current research interests are in the applications of advanced control techniques to industrial control systems and high precision mechatronic systems.

Rodney Swee Huat Teo received both his Ph.D. (2004) and M.S. (1998) degrees in Aeronautics Engineering from Stanford University and his B.Eng. (1990) in Mechanical Engineering from the National University of Singapore. He has held positions as Project Engineer (1990-1995) and Project Manager (1996-1997) on helicopter acquisition and system integration projects in the Defence Materiel Organisation of Singapore. He is currently a Senior Research Scientist of the Temasek Laboratories at the National University of Singapore. His current work is in research and development in the area of autonomy for mini unmanned aerial systems.