2018 International Conference on Unmanned Aircraft Systems (ICUAS)
Dallas, TX, USA, June 12-15, 2018

A Fast Obstacle Collision Avoidance Algorithm for Fixed Wing UAS

Zijie Lin, Lina Castano, and Huan Xu *Member, IEEE*

***Abstract*—This paper presents a novel fast collision avoidance algorithm for navigation in 3D space of fixed-wing Unmanned Aerial Systems (UAS). This algorithm is aimed at increasing the ability of aircraft operations to complete mission goals by enabling fast collision avoidance of multiple obstacles. The new algorithm, named Flexible Geometric Algorithm (FGA), combines geometric avoidance of obstacles and selection of a critical avoidance start time based on kinematic considerations. FGA reduced computational time by 90% when compared to current waypoint generation methods for collision avoidance. The starting point for the avoidance time window is determined by collision likelihood. Using this algorithm, the (Unmanned Air Vehicle) UAV is able to avoid static and dynamic obstacles while still being able to recover its original trajectory after successful collision avoidance. Simulations for different mission scenarios show that this method is much more efficient at avoiding multiple obstacles than other methods. Algorithm effectiveness validation is provided with Monte Carlo simulations and parametric results. In addition, this algorithm does not have specific requirements on the sensor data types and can be applied to cooperative and non-cooperative intruders.**

***Index Terms*—fast avoidance, avoidance efficiency, geometric algorithms, flexible avoidance time window**

I. Introduction

UNMANNED Aerial Systems (UAS) have been utilized in a number of civil and military applications, including important public missions such as disaster relief, firefighting, search and rescue, as well as other applications in agriculture, infrastructure and scientific research. UAS are a fast growing industry with extensive economic implications and will eventually be integrated into the national airspace system (NAS), which will require UAS to safely interact with other air vehicles. Standards for safe integration must be in place before a UAS can share the same airspace as manned aircraft, or when they operate in areas where they may inadvertently deviate into airspace designated to manned air vehicles. A UAS consists of an unmanned air vehicle (UAV) plus a ground control, dedicated software, sensors, telemetry, etc., all of which need research and development, testing and evaluation for safe integration with the national airspace.

In recent years, the Department of Transportation and the FAA released a proposed set of regulations for small UAS (under 55 pounds) to enter the mainstream of U.S. civil aviation. One of the imperative technological requirements for safe integration of UAS into NAS is sense and avoid (SAA) capabilities. The purpose of SAA features is to prevent collisions with other traffic, natural fliers, population on the ground, or collateral damage to property. The SAA function needs to be able to provide a 'self separation' service, which would engage before a collision avoidance maneuver is needed and could therefore consist of gentler maneuvers [1]. Research on ground and airborne SAA sensor performance, data communication and algorithms are to provide solutions to avert possible collisions [2].

In this work, we consider navigation scenarios where self-separation was either not possible or precluded by occurrence of emergent events and therefore obstacle avoidance was necessary. In addition, FGA was designed such that selective start times would allow for shorter paths and overall separation from obstacles, so that a UAV would be able to get through busy airspace in a more efficient, less costly and more time effective way. FGA simulations entail avoidance of collisions with static obstacles such as buildings of different heights, and multiple dynamic intruders, as seen in Figure 1.

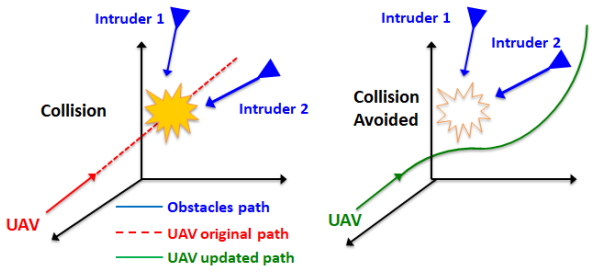


Fig. 1. UAV collision avoidance schematic: UAV avoids two dynamic obstacles by using FGA.

One major difficulty for widespread use of UAS is autonomous obstacle avoidance. Obstacles, e.g. buildings, birds and aircraft in the environment, will compromise the safety of the UAV's flight mission as well as of any obstacles encountered along its path. UAVs that stray from safe flight zones and into unpermitted airspace are

This work was supported in part by the Maryland Industrial Partnerships (MIPS) Program and by Millennium Engineering and Integration, Co.

Zijie Lin is with the Department of Electrical and Computer Engineering, University of Maryland, College Park, MD 20742 USA e-mail: zlin1@terpmail.umd.edu.

L. Castano is with the Aerospace Engineering Department, University of Maryland, College Park, MD 20742 USA e-mail: linacs@umd.edu.

H. Xu is with the Aerospace Engineering Department and Institute for Systems Research, University of Maryland, College Park, MD 20742 USA e-mail: mumu@umd.edu.

also hazardous. According to a report [3] released by the Federal Aviation Administration, between Aug, 2015 and Jan. 31, 2016, about 600 UAVs were in close proximity to manned aircraft or close to airports. To guarantee operational safety of UAS, new detection and avoidance algorithms need to be developed and tested [4], [1]. These new algorithms need to be compatible with current and future manned aircraft collision avoidance systems such as TCASII/ACAS X [5], as well as surveillance systems such as automatic dependent surveillance-broadcast (ADS-B) and air traffic control (ATC) separation management procedures and tools [6].

Obstacle avoidance algorithms have different properties depending on the amount of information available. When sufficient information is available about the obstacle set and a single vehicle is present then global planning methods may be used. When only local information is available, sensor based methods are used, with reactive methods being a subset of the latter one [7]. There are several widely used algorithms such as: force field approach [8], optimal approach [12], and see and avoid approach [8].

The force field approach [9] is based on the concept of potential fields; it calculates the electrical/magnetic potential or the gradient of particular variables between the UAV and obstacles. These algorithms assume that obstacles cause an attraction and repulsion force, and find a trajectory to avoid collisions. However, local minima in these methods is a challenging issue. While they have been studied extensively, they require the UAV to have an extremely high sensor sensitivity [10].

Sampling-based path planning algorithms [11], optimal path planning algorithms and waypoint generation methods [12], [13] have also been studied. They test all available points around a cube to choose a sub-waypoint [12] [13], then develop a path between the current waypoint and the sub-waypoint, and repeat this step until an optimal obstacle free path is found [14], [15]. Though these algorithms are successful at generating an obstacle-free path for a UAV in the situation of multiple obstacles, they are particularly costly in computation time [15] so that are not practical for real-time avoidance [16] [17].

There have been a number of algorithms developed in the area of sense, detect, avoid (SDA) [18]. Some work in the literature concentrates on the sensing and detection aspect, including improvement of sensor sensitivity [19], [20]. Sensor technologies used by SAA include active interrogation transponders [1], electro-optic [21], laser/LIDAR [6], onboard or ground based radar, ADS-B [22] and acoustic technologies [1].

Another widely used approach is found in geometric based algorithms [1], [23], [24], which can be thought of as a type of see and avoid algorithm. Geometric algorithms use the geometric relationship between a UAV and its obstacles to calculate an updated path for the UAV, requiring less processing power and are therefore more suitable for calculations onboard UAVs due to their size, weight and power constraints. Geometry based avoidance algorithms can be roughly classified into two kinds, [1], those which produce heading angle changes using information of the vehicle motion/location and those which use speed variation [24] to avoid the intruders instead of changing its direction. This paper develops a geometric obstacle avoidance algorithm, which is able to classify obstacles, perform the avoidance operations and allow the vehicle to return to its original navigation course.

This paper is organized as follows: Section II provides the problem formulation, avoidance geometry and model assumptions of FGA. Section III describes the collision scenarios, obstacle classification and working mechanism of the FGA algorithm. Section IV shows simulation results and analysis. Section V presents conclusions and discusses future work.

II. PROBLEM FORMULATION

A. Overview of FGA

The Flexible Geometric Algorithm (FGA) builds on concepts presented by [25] and [1]. It is constructed using differential geometry, which utilizes spatial location, relative velocities, angular displacements, and collision cone concepts [26] to find probable conflicts [27].

The algorithm assumes that the fixed-wing vehicle is flying at a constant speed. Data about position and velocity of the obstacles is assumed to be given and accessible to the UAV. FGA first scans the nearby environment using an onboard sensor (which for the purposes of this paper is assumed to be a perfect sensor) and continuously determines if there are any obstacles that could potentially threaten the UAV and are closer than the separation distance. After a potential obstacle is detected, two 2D collision cones, one in horizontal and another one in vertical, in the body frame of the aircraft, are projected on the encountered potential obstacle's protected disk. It then determines which particular avoidance procedure to apply (static or dynamic, single or multiple obstacles). Depending on the identified obstacle type and associated information, FGA then automatically calculates the optimal time function for the UAV to start the avoidance operation. All the variable values needed for the avoidance are calculated and locally stored.

When the UAV reaches the critical avoidance time, FGA then triggers the avoidance operation and allows it to operate during a time window of flexible duration until a safe distance from the obstacle is achieved. After that, the algorithm searches for the forwarded waypoints located on the initial path and lets the UAV return to the original intended path immediately. This minimizes the length of the path that is to be recalculated to avoid the obstacles, reducing computation time in comparison to waypoint generation or optimal path planning methods, which typically recalculate full path lengths. Different algorithms were compared in simulation to FGA, and results showed that FGA is faster and yields shorter avoidance paths.

The distinct features of the algorithm can be summarized as a combination of: (1) obstacle classification after

detection (2) short computation time (90% faster than an optimal path planning method), (3) least deviations from original path (67% shorter than optimal path planning method), (4) constant UAV speed, and (5) the ability to deal with complex environments such as multiple moving obstacles.

B. Obstacle Avoidance Geometry

A UAV with onboard FGA detects a potential obstacle if the surveillance distance R is violated. A few additional cylindrical and spherical layers can be added for additional surveillance and depending on the segmentation of the airspace. After the potential obstacle is detected, FGA will consider the obstacle a possible threat and will begin to calculate collision cones in horizontal or vertical directions to ascertain whether the obstacle is actually a threat. Figure 2 shows a UAV flying towards a dynamic obstacle in 3D space. The safe separation distance d_m is set around obstacle A and defines a protected sphere. The resulting relative velocity V_{ua} determines whether the UAV will run into the obstacle's protected sphere of radius d_m. In this work, the closest allowable point of approach is defined as any point on the edge of the protected sphere around the obstacle. In Figure 2, the point of closest approach is within the protected sphere and therefore is a scenario for a potential collision. The 3D collision cone has its apex at the location of the UAV and axis defined along $\overline{UA}$, with base radius $r_{min} \leq d_m$. and opening angle ν, as shown in:

$$\nu = tan^{-1}\left(\frac{r_{min}}{\overline{UA}}\right). \quad (1)$$

A collision is likely if the relative velocity is inside the collision cone of base radius d_m. Table I shows the definitions of the variables that are involved in FGA.

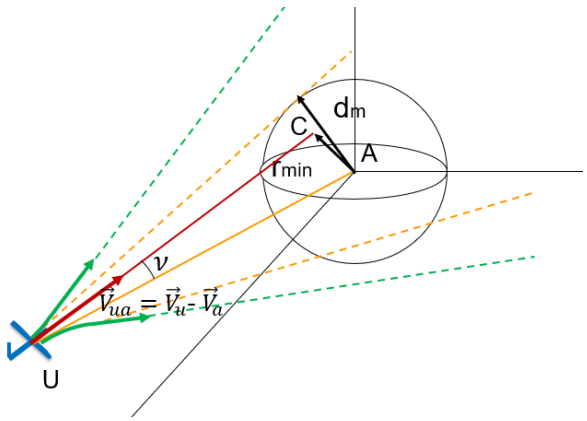


Fig. 2. Geometry of UAV and a potential collision

FGA executes its avoidance operation in 2D planes of 3D space. The 2D planes correspond to horizontal and vertical planes attached to the body frame of the UAV. A 2D cut of Figure 2 is shown in Figures 3 and 4. Figure 3 shows that the relative velocity is inside the 2D collision cone and therefore will not be at the desired separation distance from the obstacle. In this case, FGA calculates a critical start time based on velocity and time to closest approach. Figure 4 shows that it is outside the collision cone and in this case the UAV doesn't need to activate its collision avoidance mechanism.

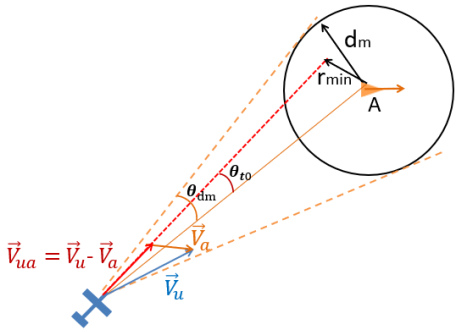


Fig. 3. Geometry of FGA dynamic obstacle avoidance for relative velocity inside the 2D collision cone

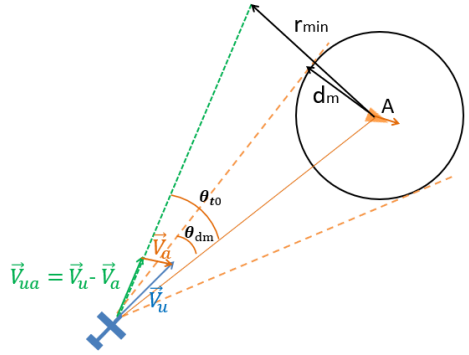


Fig. 4. Geometry of FGA dynamic obstacle avoidance for relative velocity outside the 2D collision cone

TABLE I: Main Variables in FGA

$U = U_o$	**Location of UAV when obstacles are detected**
$A_i = (A_x, A_y, A_z)$	**Obstacles' location when they are detected**
V_u	**Velocity of the UAV**
V_a	**Velocity of the obstacle**
V_{ua}	**Relative Velocity**
C	**PCA, point of closest approach**
ρ	**Minimum turning radius**
R	**Potentially dangerous obstacle, surveillance radius**
r_c	**Critical relative distance upon which the avoidance operation must engage immediately**
d_m	**Minimum safe separation radius. Note that** $R > r_c > d_m$
r_i	**Distance between i-th obstacle and UAV when detected**

C. Model Assumptions

The osbtacle avoidance simulations in this paper are based on the following assumptions:

(1) The algorithm is applicable for both 2D and 3D. However the avoidance operation is 2D. The UAV can either change its yaw or pitch angle.

(2) For the fixed-wing UAV model we are using UMD's RMRC Anaconda aircraft. Its velocity ranges from 13 m/s to 19 m/s and is set to be constant during the simulation [1]. The minimum separation radius d_m is set to be equal to the minimum turning radius. The turn radius ranges from 30-40 m/s at 30 degrees bank angle. The rate of turn ranges from 17 deg/s to 25 deg/s. The vehicle turn rate kinematics and climb speed assume constant velocity. The vertical distance traveled at climb rate V_c is:

$$z = V_c t_z = \frac{x V_c}{V_u - V_a}, \tag{2}$$

where x is the forward distance traveled. Assuming a coordinated turn in x-y, the turn radius ρ (in feet) of the fixed-wing UAV, in terms of airspeed of UAV and bank angle ϕ is:

$$\rho = \frac{V_u^2}{11.26 tan(\phi)}, \tag{3}$$

with a rate of turn (deg/s) of:

$$\dot{\psi} = \frac{1091 tan(\phi)}{V_u}, \tag{4}$$

and V_u the true airspeed in knots.

(3) This paper does not have any requirements on a specific type of onboard sensor. Any sensors that can be installed in the UAV and provide accurate location and speed data are applicable to the FGA algorithm, similar to [22].

(4) The search radius of the sensor onboard the UAV in this paper is set to be 50 times the minimum separation radius d_m. The potential danger radius R is set to be 10 times the minimum separation radius d_m. These parameters can be changed according to the different requirements.

(5) In the simulation, to guarantee general properties, all of the obstacles are randomly generated by an obstacle generation module. The module generates 10-20 static or dynamic obstacles for each case and about $20-30\%$ of them will be true hazards for the UAV. The data the obstacles generate includes the velocity, location, direction and other necessary variables.

III. AUTOMATIC FAST WINDOW GEOMETRIC AVOIDANCE ALGORITHM (FGA)

Once an obstacle is determined to be a threat, the collision avoidance mechanism is activated. The FGA algorithm is described in Algorithm 1 below. Obstacles classification is a first step in FGA, after which it will choose the appropriate avoidance procedure. The information about obstacles is saved onboard and is later used in the avoidance operation.

The FGA includes three major processes. First, the automatic sub-module selection (Section IIIA) is designed to distinguish different types of obstacles (static or dynamic, single or multiple). Second, the algorithm then evaluates the cost of changing the vehicle's pitch angle or heading angle (section IIIB1), as well as maximum possible angles for both, and chooses, for instance, a change in heading angle as the avoidance maneuver. The third process, one that decides the optimal avoidance starting point (section IIIB2) to shorten the path, recalculating time and path length (section IIIB3). When the avoidance operation stops, the algorithm returns the UAV to the next feasible waypoint.

FGA aims to decrease computation time, minimizing changes in the UAV's initial path and keeping the cruise speed of the UAV constant during the avoidance operation process. Therefore, the UAV will change its path selectively to avoid the obstacles when it is at a specific distance from them, not immediately after obstacle detection, which means that the avoidance operation will trigger only when the UAV reaches a critical point. This distance tolerance can be increased depending on application requirements. In summary, FGA has the following three major advantages which translate into reduced computation time: (1) It is approximately 90% faster than current waypoint-generation methods applied in multiple obstacle avoidance scenarios, as will be shown in Section IV, (2) It maintains constant UAV velocity instead of slowing down the UAV, and (3) It is efficient at avoidance of multiple obstacles simultaneously or in sequence using the different critical avoidance times assigned to each corresponding obstacle. We next describe the processes of the FGA in further detail.

A. Automatic Avoidance Module Selection based on Obstacle Classification

An automatic sub-module selection wil be chosen after the algorithm classifies obstacles into single or multiple and static or dynamic:

- Static obstacles: When the UAV is flying at lower altitudes, it may encounter static obstacles such as buildings, electrical towers and trees.
- Dynamic obstacles: At higher altitudes, the UAV may encounter other moving aircraft. If the distance between the UAV and other aircraft are smaller than the minimum separation of either UAV or aircraft, the collision is considered to be imminent. Furthermore, the UAV could encounter multiple moving obstacles, such as entering an area occupied by a group of aircraft; which means that the UAV needs to avoid several obstacles simultaneously or in sequence. In these cases, the control system onboard the UAV should consider additional factors such as the relative distance, size, relative velocity, and potential level of damage of the obstacles, and then calculate a new collision-free path.

B. Flexible Geometry Avoidance Process

The following steps apply to both static and dynamic obstacles.

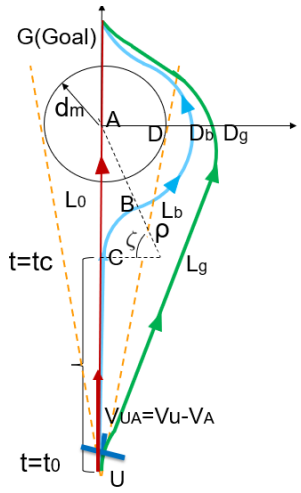


Fig. 5. FGA diagram for single obstacle

1) Avoidance maneuver and protected distance: In Figure 5, an aircraft is found to be a single dynamic obstacle A with coordinates (A_x, A_y, A_z). In this scenario, the FGA will utilize the single dynamic avoidance sub-module.

The algorithm evaluates the cost of changing the vehicle's pitch angle or heading angle for the 2D avoidance maneuver. It also calculates the maximum possible angles for both possible maneuvers, and chooses, for instance, a change in heading angle (Figure 5).

The minimum safe separation d_m is then defined as a vertical separation distance $d_{m,v}$ or a horizontal separation distance $d_{m,h}$ according to the type of avoidance maneuver chosen. In Figure 5, $d_m = d_{m,h}$, as the chosen avoidance entails a change in heading. These separation distances are application dependent and determined a priori.

For executing the avoidance maneuver, vehicle kinematics constraints described in section IIC are applied. The minimum turn radius ρ, as described in Equation 3, will be set by the protected distance d_m such that:

$$\rho_{min}(V_u) = \rho_{min} = d_m \tag{5}$$

as this is the minimum radius of turn that the UAV can use without entering the protected distance d_m.

In the following step, the FGA calculates the critical avoidance starting time t_c.

2) Critical avoidance start time: Here, we introduce a parameter t_c, the critical avoidance starting time. After t_c, the UAV is unable to avoid entering the safety area of radius d_m of the obstacles. This is applicable to cases where path deviations are undesirable or during emergency procedures that require more agile vehicle maneuvers.

Recall Figure 3, $\overrightarrow{V_a}$ is the velocity of the obstacle ($||\overrightarrow{V_a}|| = 0$ for the static obstacle) and $\overrightarrow{V_u}$ is the velocity of the controlled UAV. The resulting relative velocity is $\overrightarrow{V_r} = \overrightarrow{V_{ua}} = \overrightarrow{V_a} - \overrightarrow{V_u}$ with initial direction θ_{vr0}. The relative distance vector between the UAV and the obstacle, is r with initial value $r_0 = \overline{UA}$ and initial direction θ_{r0}.

The direction difference between relative distance vector $\overrightarrow{r}$ and relative velocity vector $\overrightarrow{V_r}$ is defined as:

$$\theta_{t0} = \theta_{vr0} - \theta_{r0}. \tag{6}$$

The critical avoidance time is calculated as:

$$t_c = \frac{r_0 cos\theta_{t0} - \rho_{min} - (d_m - r_0 sin\theta_{t0})}{||V_r||}. \tag{7}$$

In addition, due to physical constraints, the start time for avoidance operation needs to anticipate system response, such that $t_k = t_c - t_{res}$ ($t_{res} = 0$ for ideal vehicles). Therefore the critical avoidance time considering vehicle response can be calculated as:

$$t_k = t_c - t_{res} = \frac{r_0 cos\theta_{t0} - 2d_m + r_0 sin\theta_{t0}}{||V_r||} - t_{res} \tag{8}$$

The avoidance operation will be successful if the UAV starts before avoidance time t_k.

For multiple obstacle cases, each moving obstacle has a t_{ci} associated with it, such that t_{ci} corresponds to the i-th obstacle. Hence, for n multiple obstacles, the overall critical t_c is the minimum of (t_{ci}):

$$t_c = min(t_{ci}), i = 1,2,3...n; \tag{9}$$
$$t_k = t_c - t_{res};$$

3) Path length calculation: Figure 5 shows a simplified description of the mechanism of the FGA algorithm. This section will describe the different path lengths generated by the different mechanisms. Note the meaning of the different line segments in Figure 5:

Red path: without any collision avoidance operation. The length of the total path is L_0.

Green path: collision avoidance from the time the obstacle is detected. The length of the total path is L_g.

Blue path: collision avoidance from critical starting time t_c. The length of total path is L_b.

Lemma 3.1: The recalculated path using FGA is shorter than the one without applying FGA.

Proof.

In Figure 5, For blue line (applying FGA): $\overline{DD_b} = O(d_m)$

$$L_{b1} = \overline{UC}, L_{b2} = \widehat{CB} + \widehat{BD_b}, L_{b3} = \widehat{D_bG} \tag{10}$$

Where "$-$" represents a line segment and "$\frown$" represents a curved segment. Let the turning radius of UAV to be ρ such that

$$cos(\zeta) = \frac{\rho}{\rho + d_m + O(d_m)}, \tag{11}$$

$$\begin{aligned} L_{b2} &= \rho\zeta + (d_m + O(d_m))\zeta = (\rho + d_m + O(d_m))\zeta, \\ &= (\rho + d_m + O(d_m))acos(\frac{d_m}{\rho + d_m + O(d)}), \end{aligned} \tag{12}$$

Equation 12 is an increasing function of ρ, since

$$\rho \geq \rho_{min} = d_m, \tag{13}$$

$$L_{b2min} = (2d_m + O(d_m)) * acos(\frac{d_m}{2d_m + O(d)}), \\ = \frac{2\pi d_m}{3} d_m = 2.09 d_m, \tag{14}$$

$$L_b = L_{b1} + L_{b2} + L_{b3}. \tag{15}$$

For the green line (without applying FGA):

$$L_{g1} = \widehat{UD_g}, L_{g2} = \widehat{D_g G}, \tag{16}$$

$$L_g = L_{g1} + L_{g2}, \tag{17}$$

since in obstacles generation, $r \geq 5d_m$,

$$L_{g1} > \sqrt{r^2 + d_m^2} = d_m \sqrt{1 + r^2/d_m^2} \geq d_m \sqrt{26} = 5.1 d_m, \tag{18}$$

Thus, to let

$$L_{b2} < L_{g1} \tag{19}$$

exist, the following Equation 20 need to be satisfied:

$$L_{b2} = (\rho + d_m + O(d_m)) acos(\frac{d_m}{\rho + d_m + O(d)}) < 5.1 d_m, \tag{20}$$

which requires:

$$\rho \leq 2.8 d_m. \tag{21}$$

The only constraint for ρ is Equation 13, $\rho \geq \rho_{min} = d_m$. Thus, Equation 21 can be satisfied, which means Equation 20 exists and Equation 19 is proved.

After reaching the safe separation distance, FGA(blue) and the one without FGA (green) will head to the nearest waypoint.

If FGA uses the minimum turning radius ρ_{min}, and we assume the green line also uses ρ_{min}, then according to Dubins path planning theory, the shortest path for an air vehicle to travel between two points uses minimum turning radius if circle turning is necessary. Thus

$$L_{b3} \approx L_{g2}, \tag{22}$$

Hence,

$$L_{b2} + L_{b3} < L_{g1} + L_{g2} = L_g. \tag{23}$$

Note that for FGA, the re-calculated path length is $L_{b2} + L_{b3}$, not $L_{b1} + L_{b2} + L_{b3}$.

Therefore, it is proved that after using FGA, the path that needs to be updated is shorter. □

For multiple obstacles, the proof is much more involved. To illustrate this, let's take one case for instance. From Figure 6 we have that,

$$L_{b2} + L_{b3} \approx (d_m + \Delta d)\pi, \tag{24}$$

$$L_g = L_{g1} + L_{g2} \geq (2d_m + \delta d)\pi, \tag{25}$$

Therefore,

$$L_g > L_b. \tag{26}$$

Since FGA calculates shorter paths, it will lead to a shorter computation time, assuming that the same computation speed is being used by the algorithm.

Hence, after applying FGA the path that needs to be re-calculated and the collision avoidance duration time are shortened significantly.

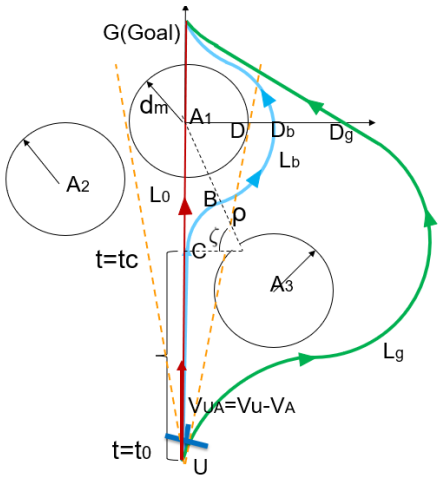


Fig. 6. FGA diagram for multiple obstacles

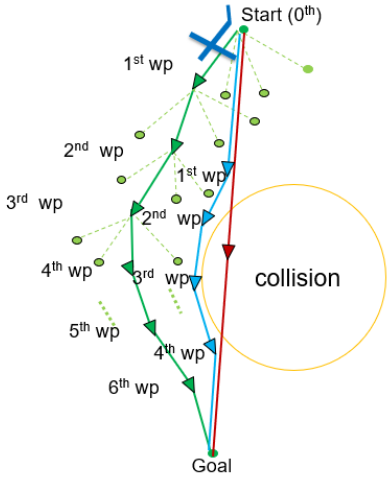


Fig. 7. Diagram of FGA applied in waypoint generation algorithm

4) Reduction of Path Calculation Example: (Figure 7): We illustrate the FGA reduction of path calculations in a waypoint generation method example. Suppose that if the obstacle avoidance operation starts when the obstacle is detected, the number of re-calculated points is m_g. Now assume that if the obstacle avoidance operation is initiated using FGA, the number of re-calculated points is m_b.

From Lemma 1, the re-calculated path using FGA is shorter than the one that does not apply FGA; that is $L_b \leq L_g$. The length between a waypoint and its sub-waypoints is a constant, therefore we have:

$$m_b \leq m_g \tag{27}$$

According to optimal path/waypoint generation path planning methods [12], to generate m waypoints with each waypoint, it has a sub-waypoints to be checked and the total calculated step is:

$$(a + a^2 + a^2(m-1)) = a + a^2 m, \tag{28}$$

substitute m_g and m_b

$$(a + a^2 + a^2(m_g - 1)) = a + a^2 m_g, \quad (29)$$

$$(a + a^2 + a^2(m_b - 1)) = a + a^2 m_b. \quad (30)$$

The ratio of steps generated will be:

$$\eta_1 = (a + a^2 m_b)/(a + a^2 m_g) = (m_b + \frac{1}{a})/(m_g + \frac{1}{a}), \quad (31)$$

and the ratio of steps saved will be:

$$\eta_2 = 1 - \eta_1 = 1 - (m_b + \frac{1}{a})/(m_g + \frac{1}{a}). \quad (32)$$

For example, if $a = 5, m_g = 100, m_b = 20,$ then

$$\eta_2 = 1 - (20 + \frac{1}{5})/(100 + \frac{1}{5}) = 0.8 = 80\%. \quad (33)$$

Algorithm 1 FGA algorithm

1: Obstacle classification: single static obstacle, single dynamic obstacle, multiple static obstacles and multiple dynamic obstacles.
2: For single static or dynamic obstacles: calculate the critical avoidance starting time t_c using Equation 7.
3: For multiple static or dynamic obstacles, calculate critical avoidance starting time t_{ci} for each obstacle, 1,2,... n. Then calculate the global critical avoidance starting time $t_c = min(t_{ci})$ for the UAV.
4: If $t_c \geq 0$, at time t_c, trigger the avoidance operation. If $t_c < 0$, it means the obstacle density is higher than the threshold so that the obstacles need to be avoided in sequence instead of simultaneously, go to step 5.
5: Divide obstacles into two categories, (1) urgent, saved to matrix M_1, (2) not urgent, saved to M_2. Avoid the obstacles in M_1 first. Then check the obstacles in M_2 and update the ones which are a threat to M_1, then repeat 1 to 5.
6: Use geometry relationship and cost estimation to adjust the direction of the UAV, after which the relative velocity between the UAV and the obstacle will change:

$$\vec{V_r} = \vec{V_a} - \vec{V_u}$$

7: Calculate the angle between the relative velocity vector $\vec{V_r}$ and relative distance vector $\vec{r}$ using Equation 6:

$$\theta_t = \theta_{vr} - \theta_r$$

8: The desired direction difference is $\theta_{dm} = d_m/r$. Repeat step 7 until $|\theta_t - \theta_{dm}| < 0.01$.
9: Check the closest approach to obstacles every dt. Stop when $r_{min} > d_m$. Search the nearest reachable waypoint on original path and fly to it.

IV. SIMULATION RESULTS AND ANALYSIS

A. Simulation Results

Monte Carlo simulations were conducted using FGA for collision avoidance between a UAV and multiple static and dynamic obstacles. The simulated UAV was given values presented in Table II. Obstacles were randomly generated by an obstacle generator module we designed and the UAV chose different avoidance operations automatically according to the different situations encountered.

TABLE II: Vehicle Variables for Monte Carlo simulations

V_a, **velocity of the obstacle**	**Generated between 0 and 25 m/s**
V_u, **velocity of the UAV**	**15 m/s**
ρ, **min turning radius**	**30 m**
R, **potential danger radius**	**300 m**
$r_c(t_c)$, **critical avoidance starting distance (time)**	**Dependent on different situations**
d_m, **safe separation radius**	**30m, note that** $R > r_c > d_m$

Tables III and IV summarize the Monte Carlo simulation results for a UAV avoiding static or dynamic obstacles using FGA. In these simulations we have set the minimum vertical and horizontal required separation distance to be d_m, v =30 m and d_m, h =30m, respectively. From Tables III and IV, the overall success rate is 98.9% for static obstacles and 95% for dynamic obstacles, which is more efficient as compared to other algorithms such as [21], which has an overall success rate for static/dynamic obstacles of 88%.

TABLE III: FGA Avoids Static Obstacles Effectively

Static Obstacles			
Num. of obst.	**Num. of cases**	**Success num.**	**Success rate %**
0	1091	1091	100
1	1756	1756	100
2	1298	1295	99
3	571	553	97
> 3	284	272	95
Summary	5000	4957	98.9
Total time cost for 5000 cases: 28.1 s, average time cost: 0.0056 s			

TABLE IV: FGA Avoids Dynamic Obstacles Effectively

Dynamic Obstacles			
Num. of obst.	**Num. of cases**	**Success num.**	**Success rate %**
0	960	960	100
1	1694	1694	100
2	1413	1333	94
3	680	602	88
> 3	253	198	78
Summary	5000	4787	95
Total time cost for 5000 cases: 30.2 s, average time cost: 0.006 s			

An example of what each of these Monte Carlo runs looks like is shown in Figures 8 and 9. Figure 8 provides the trajectories of two obstacles (blue paths) and the trajectory of the UAV before (red path) and after (green path) applying the FGA avoidance algorithm. The UAV starts the avoidance operation at $t_c = 3.2s$ after the obstacle has been detected. It manages to achieve a 31 meter separation (minimum safe separation $d_m = 30m$) from the obstacles, while the separation without applying FGA is 25 meters.

Figure 9 depicts another simulation case. Here, the UAV avoids three dynamic obstacles that threaten to break its minimum separation distance requirements. In this figure, Figure 9(a) shows the 3D paths of the UAV and its obstacles. Figure 9(b) has a comparison of relative

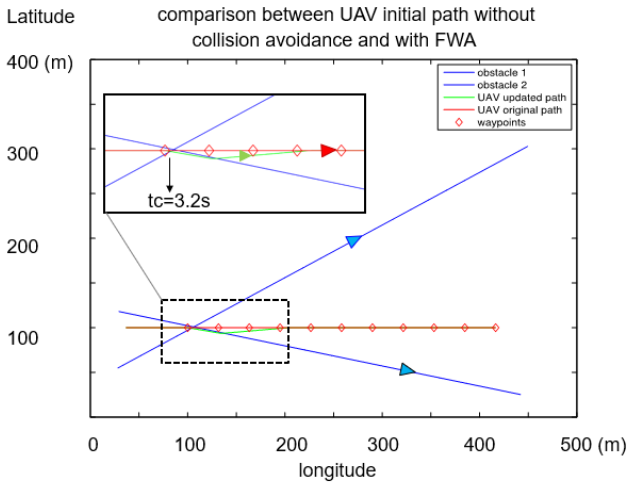


Fig. 8. 2D view of simulation for Case 7 (from Table VIII)

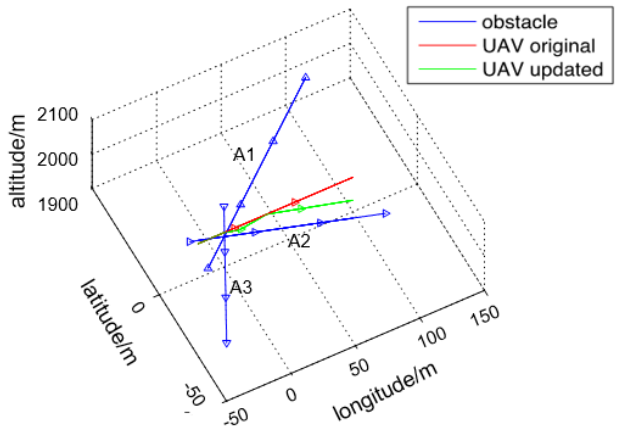
distance changes before and after applying FGA. Three different types of red/green lines represent the relative distance between the three different obstacles and the UAV before/after applying FGA. Similar to the previous case, in this situation, the UAV starts the avoidance operation at $t_c = 0.9s$ instead of starting it at the time when the obstacles are detected. It manages to achieve a 32 meter separation (minimum safe separation $d_m = 30m$) from the obstacles while the separation without applying FGA is 28 meters.

The failure rate of FGA is small, as shown in Tables III and IV. Figures 10 and 11 provide an analysis of the factors that influence the failure rate of FGA in the Monte Carlo simulations. Figure 10 shows results of failure rates in avoiding static obstacles. In Figure 10(a), failure rate 1 corresponds to cases where the initial altitude of the UAV is higher than all of the static obstacles. Failure rate 2 is when the UAV's initial altitude is between the altitude of the highest and the lowest obstacle. From results in Figures 10(a) and 10(b), flying at higher altitudes and larger detection distances will decrease the FGA failure rate.

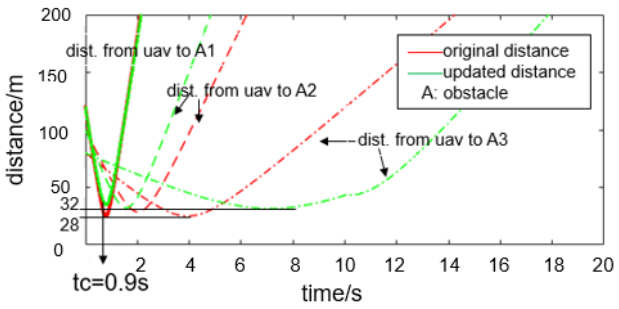
In Figure 11, results of failure rates when avoiding dynamic obstacles are presented. In Figure 11(a), failure rate 1 indicates cases where the initial altitude of the UAV is higher or lower than the altitude of all of the dynamic obstacles. Failure rate 2 is when the initial UAV altitude is between the highest and lowest obstacle. It also shows failure rates per number of obstacles. From these results, fewer number of obstacles, larger altitude difference, and larger initial distance when the UAV detects the obstacles, will decrease the failure rate. A smaller ratio of average obstacles' speed to UAV speed, will also decrease the failure rate.

B. Comparison of FGA and other Collision Avoidance Algorithms

After validating FGA using Monte Carlo simulations, we now compare FGA with a traditional geometric algorithm

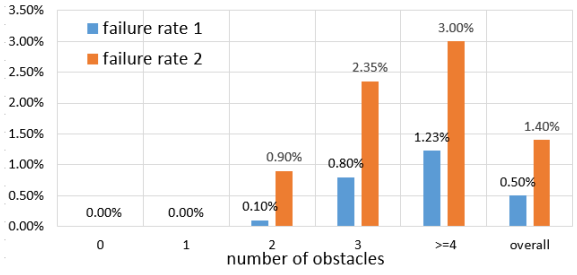


(a) 3D paths for UAV and obstacles

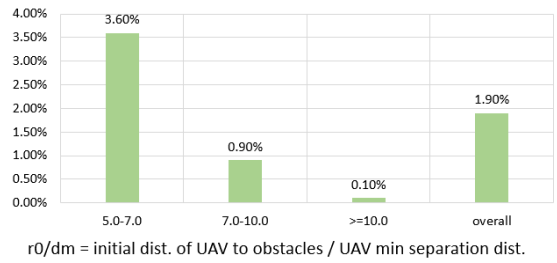


(b) Relative distance between UAV and obstacles

Fig. 9. 3D view of simulation for Case 9 (from Table VIII)

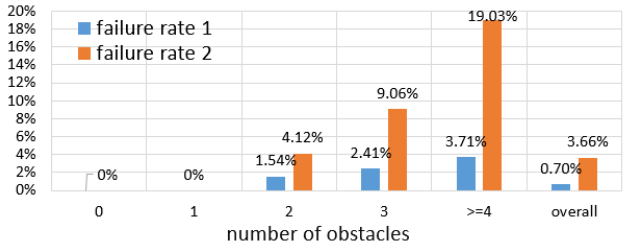


(a) Failure rate-altitude and number of obstacles

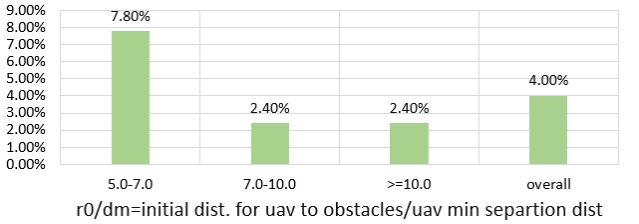


(b) Failure rate-initial distance between obstacles and UAV

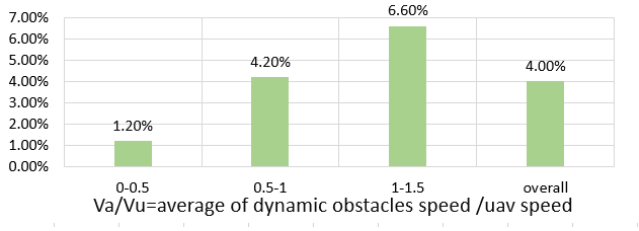
Fig. 10. FGA failure rate for avoiding static obstacles



(a) Failure rate-altitude and number of obstacles



(b) Failure rate-initial distance between obstacles and UAV



(c) Failure rate-dynamic obstacles' speed/UAV speed

Fig. 11. FGA failure rate for avoiding dynamic obstacles

[1] and an optimal algorithm [12]. Table V provides a qualitative comparison of the main properties of these algorithms along with their advantages and disadvantages. This table shows that FGA has a combination of advantageous characteristics when compared to the other two methods.

TABLE V: Comparison of Properties of FGA, Optimal, and Traditional Geometric Algorithms for Collision Avoidance

Properties	Trad Geo.	Opt.	FGA.
Optimal avoidance starting time	N	N	Y
Consider UAV velocity	Y	Y	Y
Consider obstacle velocity	Y	Y	Y
Consider minimum turning rate	Y	N	Y
Stopping point	N	N	Y
Resume original path	N	N	Y
Flexible avoidance operation time	N	N	Y
Manage single static/dyn. obstacle	Y	Y	Y
Manage multi. static/dyn. obstacles	Y	Y	Y
High speed computation	Y	N	Y

A comparison of average time cost of FGA and major collision avoidance methods such as potential field and optimal algorithms is presented in Table VI. This table was constructed by using values reported in the literature [14], [28], [28], [16] and [12], and correspond to either static obstacles or dynamic obstacles or mixed cases. The average time cost calculated for FGA is based on 5000 case runs, including the avoidance of both static and dynamic obstacles. This comparison indicates that the time consumed by FGA is much less than the ones reported for the potential field method and optimal method.

TABLE VI: Time Cost of Major Collision Avoidance Methods

Methods	Average time cost each case	Time saved by FGA	Evaluation
FGA	0.0056s	NA	fast
Geometry [14]	0.0096s	41.67%	fast
Potential field 1 [28]	0.082s	92.7%	medium
Optimal 1 [16]	0.18s	96.89%	slow
Optimal 2 [12]	0.3s	98%	slow

Another set of comparisons were made between FGA and optimal methods [12], one in terms of the path length generated (Table VII) and one in terms of time cost (Table VIII). Four case runs are shown in Table VII. Results from this table show that the recalculated path using FGA is much less than the one produced by the optimal algorithm. Table VIII provides detailed information of five cases chosen from the Monte Carlo runs. Results shown in this table indicate that overall, FGA has a much higher efficiency.

TABLE VII: Comparison of The Path Length between Optimal Algorithm and FGA Algorithm

Avoidance Cases	Case1	Case2	Case3	Case4
initial path length of UAV	200	200	200	200
Updated path length of UAV using opt. method	252	241	270	305
Updated path length of UAV using FGA	70.5	65.3	82.4	91.7
Path saved by FGA	76%	72.9%	70.4%	70%

TABLE VIII: Comparison of Optimal and FGA Algorithms Performance in Five Cases of Multiple Dynamic Obstacles

Case number	Case 5	Case 6	Case 7	Case 8	Case 9
Obstacles/pos relative to UAV	2/below	2/above	2/behind	2/ahead	3/2 behind, 1 ahead
Dist. when detected m	90-120	< 90	90-120	100-150	100-150
Obstacle speed m/s	12-18	12-15	12-20	10-12	10-17
Orig. min distance m	3 V	5 V	25 H	20 H	20 H
Avoidance subroutine	Pitch up	Pitch up	Steer left	Steer left	Steer right
Updated min dist.	50 V	50 V	31 H	35 H	32 H
Avoid. start time, FGA	1.82	1.52	2.35	0.92	0.36
Time cost, opt. method	0.26	0.3	0.5	0.5	0.8
Time cost, FGA	0.0056	0.0056	0.0060	0.0062	0.0065
Time saved by FGA	97.85%	98.13%	98.8%	98.76%	99.19%

In summary, we have shown that FGA is a fast and effective collision avoidance algorithm via results from Monte Carlo simulations. We have also presented qualitative and quantitative results that show that FGA has a shorter computation time and path length when compared to an

optimal algorithm. Other currently used collision avoidance algorithms such as optimal path algorithms, do not determine a starting point for the avoidance operation or how long it should last and where to stop according to mission objectives and to get shorter paths. FGA checks for these and it also checks for future waypoints after successful avoidance. The optimal method searches all available points to pick up the next waypoint and repeats the same operation at each iteration, but it requires a lot of memory and computation time. The optimal method also requires a fast heading change and a small turning radius, which might be limited by hardware constraints of the UAV. Therefore, even after calculation of the optimal path, the UAV may not be able to follow this path, as it may require that the UAV uses a turning radius which is smaller than its minimum turning radius. In contrast, FGA overcomes these shortcomings, as it is tailored to the vehicle kinematics. In addition, performance of FGA shows a clear advantage in computation cost and path length as compared to the optimal method.

V. CONCLUSION AND FUTURE WORK

From the above simulation results, FGA proves its efficiency for a fixed-wing UAV at avoiding different types of obstacles, especially multiple moving obstacles. The short response time and light computation load make FGA a practical algorithm for real-time applications.

As part of future work, FGA can be extended into more complicated environments. For instance, in a scenario that includes a larger number of static and dynamic obstacles. In this case, FGA will calculate different critical avoidance times corresponding to the different obstacles. FGA has the ability to utilize these different times to calculate a collision avoidance sequence for the UAV.

ACKNOWLEDGMENT

The authors would like to thank the Maryland Industrial Partnerships (MIPS) Program and Millennium Engineering and Integration, Co. for their support of this project.

REFERENCES

[1] P. Angelov, *Sense and avoid in UAS: research and applications*. John Wiley & Sons, 2012.

[2] X. Yu and Y. Zhang, "Sense and avoid technologies with applications to unmanned aircraft systems: Review and prospects," *Progress in Aerospace Sciences*, vol. 74, pp. 152–166, 2015.

[3] I. Karamouzas, B. Skinner, and S. J. Guy, "Universal power law governing pedestrian interactions," *Physical review letters*, vol. 113, no. 23, p. 238701, 2014.

[4] R. J. Kephart and M. S. Braasch, "See-and-avoid comparison of performance in manned and remotely piloted aircraft," *IEEE Aerospace and Electronic Systems Magazine*, vol. 25, no. 5, pp. 36–42, 2010.

[5] C. von Essen and D. Giannakopoulou, "Analyzing the next generation airborne collision avoidance system," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2014, pp. 620–635.

[6] M. Huerta, "Integration of civil unmanned aircraft systems (UAS) in the national airspace system (NAS) roadmap," *Federal Aviation Administration, Retrieved Dec*, vol. 19, p. 2013, 2013.

[7] M. Hoy, A. S. Matveev, and A. V. Savkin, "Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey," *Robotica*, vol. 33, no. 3, pp. 463–497, 2015.

[8] J. Seo, Y. Kim, and A. Tsourdos, "Differential geometry based collision avoidance guidance for multiple UAVs," *IFAC Proceedings Volumes*, vol. 46, no. 19, pp. 113–118, 2013.

[9] K. Sigurd and J. How, "UAV trajectory design using total field collision avoidance," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2003, p. 5728.

[10] H. Chao, Y. Gu, and M. Napolitano, "A survey of optical flow techniques for UAV navigation applications," in *Unmanned Aircraft Systems (ICUAS), 2013 International Conference on*. IEEE, 2013, pp. 710–716.

[11] Y. Lin and S. Saripalli, "Path planning using 3D dubins curve for unmanned aerial vehicles," in *Unmanned Aircraft Systems (ICUAS), 2014 International Conference on*. IEEE, 2014, pp. 296–304.

[12] T. Yu, J. Tang, L. Bai, and S. Lao, "Collision avoidance for cooperative UAVs with rolling optimization algorithm based on predictive state space," *Applied Sciences*, vol. 7, no. 4, p. 329, 2017.

[13] P. Yang, K. Tang, J. A. Lozano, and X. Cao, "Path planning for single unmanned aerial vehicle by separately evolving waypoints," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1130–1146, 2015.

[14] D. Bozhinoski, A. Bucchiarone, I. Malavolta, A. Marconi, and P. Pelliccione, "Leveraging collective run-time adaptation for UAV-based systems," in *Software Engineering and Advanced Applications (SEAA), 2016 42th Euromicro Conference on*. IEEE, 2016, pp. 214–221.

[15] D.-S. Jang, H.-J. Chae, and H.-L. Choi, "Optimal control-based UAV path planning with dynamically-constrained TSP with neighborhoods," pp. 373–378, 2017.

[16] H. Jing-Lin, S. Xiu-Xia, L. Ri, D. Xiong-Feng, and L. Mao-Long, "UAV real-time route planning based on multi-optimized rrt algorithm," in *Control And Decision Conference (CCDC), 2017 29th Chinese*. IEEE, 2017, pp. 837–842.

[17] D. Pascarella, S. Venticinque, and R. Aversa, "Autonomic agents for real time UAV mission planning," in *Ubiquitous Intelligence and Computing, 2013 IEEE 10th International Conference on and 10th International Conference on Autonomic and Trusted Computing (UIC/ATC)*. IEEE, 2013, pp. 410–415.

[18] H. Pham, S. A. Smolka, S. D. Stoller, D. Phan, and J. Yang, "A survey on unmanned aerial vehicle collision avoidance systems," 2015.

[19] J. Haugen and L. Imsland, "Monitoring moving objects using aerial mobile sensors," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 2, pp. 475–486, 2016.

[20] C. Luo, S. I. McClean, G. Parr, L. Teacy, and R. De Nardi, "UAV position estimation and collision avoidance using the extended kalman filter," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 6, pp. 2749–2762, 2013.

[21] S. Saha, A. Natraj, and S. Waharte, "A real-time monocular vision-based frontal obstacle detection and avoidance for low cost UAVs in GPS denied environment," in *Aerospace Electronics and Remote Sensing Technology (ICARES), 2014 IEEE International Conference on*. IEEE, 2014, pp. 189–195.

[22] Y. Kwang and Y. Kwang, "Performance simulation of radar sensor based obstacle detection and collision avoidance for smart UAV," in *Digital Avionics Systems Conference, 2005. DASC 2005. The 24th*, vol. 2. IEEE, 2005, pp. 10–pp.

[23] J.-W. Park, H.-D. Oh, and M.-J. Tahk, "UAV conflict detection and resolution based on geometric approach," *International Journal of Aeronautical and Space Sciences*, vol. 10, no. 1, pp. 37–45, 2009.

[24] S. J. Guy, J. Van Den Berg, M. C. Lin, and D. Manocha, "Geometric methods for multi-agent collision avoidance," in *Proceedings of the twenty-sixth annual symposium on Computational geometry*. ACM, 2010, pp. 115–116.

[25] H.-S. Shin, A. Tsourdos, and B. White, "UAS conflict detection and resolution using differential geometry concepts," *Sense and avoid in UAS: Research and applications*, vol. 62, 2012.

[26] J. Goss, R. Rajvanshi, and K. Subbarao, "Aircraft conflict detection and resolution using mixed geometric and collision cone approaches," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2004, p. 4879.

[27] Z. Daniels, L. Wright, J. Holt, and S. Biaz, "Collision avoidance of multiple UAS using a collision cone-based cost function," *Computer Science and Software Engineering Department, Auburn University, Tech. Rep. CSSE12-07*, 2012.

[28] Q. Wang and J. Zhang, "MPC and TGFC for uav real-time route planning," in *Control Conference (CCC), 2017 36th Chinese*. IEEE, 2017, pp. 6847–6850.