Robotica: page 1 of 15. © Cambridge University Press 2019
doi:10.1017/S0263574719000754

# Autonomous Area Exploration and Mapping in Underground Mine Environments by Unmanned Aerial Vehicles*

Hang Li§, Andrey V. Savkin§† and Branka Vucetic‡

§*School of Electrical Engineering and Telecommunications, University of New South Wales, Sydney, Australia*
‡*School of Electrical and Information Engineering, University of Sydney, Sydney, Australia*

(Accepted April 25, 2019)

**SUMMARY**
In this paper, we propose a method of using an autonomous flying robot to explore an underground tunnel environment and build a 3D map. The robot model we use is an extension of a 2D non-holonomic robot. The measurements and sensors we considered in the presented method are simple and valid in practical unmanned aerial vehicle (UAV) engineering. The proposed safe exploration algorithm belongs to a class of probabilistic area search, and with a mathematical proof, the performance of the algorithm is analysed. Based on the algorithm, we also propose a sliding control law to apply the algorithm to a real quadcopter in experiments. In the presented experiment, we use a DJI Guidance sensing system and an Intel depth camera to complete the localization, obstacle detection and 3D environment information capture. Furthermore, the simulations show that the algorithm can be implemented in sloping tunnels and with multiple UAVs.

KEYWORDS: Flying robot navigation; Unmanned aerial vehicle; Area exploration; 3D mapping; 3D navigation.

## 1. Introduction

Nowadays, due to the high achievements in the research field of unmanned aerial vehicle (UAV), flying robots are implemented in a lot of engineering applications, such as civil engineering,[1,2] agriculture engineering,[3] mining engineering,[4] smart factory[5] and other fields.[6,7] In this paper, we consider the application of UAVs in mining engineering for under ground mine mapping.

Mapping plays a significant role in underground mining. The aim of the 3D mapping in underground mining is to obtain the accurate data and model of the 3D area of the underground mine. In underground mining, the shape of the underground mine area is dynamic due to a lot of reasons, like the excavation of new tunnels or some natural factors. The accurate and up-to-date 3D model and data of the underground mine environment are, therefore, necessary for efficient and safe mining process.

With the development of robotics and sensor technologies, there is a lot of achievements in previous research works of flying robot map building. In the work,[8] the authors proposed a practical river area mapping method for a flying robot. In this work, stereo vision and global positioning system/inertial navigation system (GPS/INS) technology are integrated in the presented method for visual odometry and state estimation. A 3D laser scanner is then used for 3D modelling of the

*This work was supported by the Australian Research Council.

†Corresponding author. E-mail: a.savkin@unsw.edu.au

obstacles beside the river. In the work,[9] the authors proposed a cost-effective method to monitor landslides using structure by using motion and image correlation of multi-temporal UAV photograph. In the work,[10] the authors proposed a robust visual map building method for UAVs by using the adaptive Scale Invariant Features Transform (SIFT) algorithm and L-infinity norm-based feature matching. In another work,[11] the authors proposed a multi-volume occupancy grid-based approach to 3D mapping. Moreover, the 3D robot exploration has been utilized in the last few years. For example, in the work,[12] a novel path planning algorithm for autonomous UAV exploration of 3D environments was proposed. The algorithm is an implementation of random search tree method for 3D exploration with obstacle avoidance. In the work,[13] the authors used the next-best-view strategy combining with an online inspection algorithm to drive a flying robot to explore a 3D area and build the 3D model of obstacles. In the work,[14] a stochastic differential equation-based exploration algorithm was proposed. This algorithm involves a particle-based representation of unoccupied space and frontier detection.

In this paper, we consider the application of UAVs in underground mine mapping and propose a 3D tunnel system search and mapping algorithm. In the proposed algorithm, a UAV is modeled as an extension of 2D non-holonomic model.[15] The robot's horizontal angular velocity and vertical velocity are two control inputs and, thus, the horizontal motion control and altitude control are separated. The UAV is able to access the accurate measurements of its position and attitude. There are several approaches to achieve accurate self-positioning, such as refs..[16–18] In our navigation algorithm, the measurement of the minimum distance to obstacles and the measurement of the horizontal scan of obstacles in front of the UAV are used to navigate the robot searching the tunnel to avoid collisions with tunnel walls. The proposed method has some similarity with robot navigation methods based on visibility graphs or road maps. However, there are several important differences. Probabilistic roadmap (PRM)-type algorithms typically consist of two phases: a construction and a query phase. In the construction phase, a road map or a visibility graph is built. In the query phase, a robot path is obtained by the Dijkstra algorithm or other algorithms to find the optimal path. Typical examples of such approaches include navigation based on a priori building a visibility graph from the terrain elevation graph,[19] UAV navigation based on an a priori built cloud map of the environment,[20] navigation through a known static environment with moving obstacles with a known motion pattern,[21] navigation through a partially known environment with either norm bounded uncertainty[22] or uncertainty described by a stochastic model.[23,24] In the problem under consideration, we cannot have these two separate stages. We build a map of the underground mine and a collision-free path simultaneously. In our method, we assume that the underground mine environment is completely unknown a priori. Therefore, the UAV does not have any visibility graph or a road map a priori. The flying robot builds some picture of the environment in real-time simultaneously avoiding collisions with obstacles, walls, etc. In this process, the UAV starts with a picture of a small "local" part of the environment. Hence the proposed algorithm is closer in spirit to local collision avoidance algorithms than to global path planning algorithms.[25,26] The proposed algorithm belongs to the class of probabilistic algorithms. This property follows from the fact that at certain points of its trajectory, the robot with some probability $p > 0$ makes the decision either to continue to follow the boundary of an obstacle or switch to another path. However, the proposed approach is quite different from well-known approaches of probabilistic robotics such as the rapidly-exploring random tree (RRT) method[27] or the PRM method that involve optimization over quite large search spaces and quite large random trees. In our approach, each time the robot chooses over just two possible future trajectories so the algorithm is very fast and easily implementable in real time. The developed method can be modified for multiple flying robots.

The performance of the proposed algorithm was confirmed by computer simulations and experiments. In the experiments, we built a real flying robot equipped with an advanced sensing system and a 3D depth camera. The advanced sensing system consists of stereo vision cameras and sonar sensors and provides accurate localization and measurements of the minimum distance to surrounding obstacles. The 3D depth camera is used to capture the 3D samples of the tunnel environment for map building purpose.

Comparing with previous works, the main contribution of this paper is that we consider the use of a flying robot for autonomous tunnel environment exploration and 3D map building. The proposed navigation algorithm is relatively easily implementable with real flying robots and UAVs. The measurements we considered for the navigation are simple and easy to be obtained by many types of sensors, such as sonar sensor, stereo camera and 2D laser range finder. We also propose a sliding

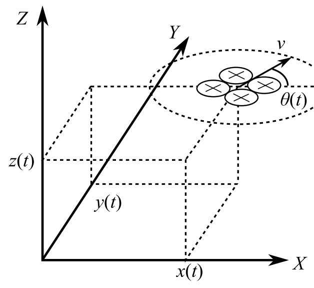


Fig. 1. The model of the flying robot.

mode control law to perform the navigation algorithm presented in this paper. The sliding mode control law is robust and makes the navigation possible for different types of UAVs. The proposed algorithm enables flying robots to build a complete map of a closed tunnel area in a finite time. Another feature of the proposed algorithm is that it can be implemented in sloping tunnels or with multiple drones.

The method presented in this paper can be used in various tunnel types including underground mine tunnels, traffic tunnels and cave tunnels. The data captured by the flying robot can be transmitted to a remote computer. The tunnel area search and map building is autonomous and an operator only needs to start or stop the map building in the remote computer.

The remaining parts of this paper are organized as follows. Section 2 describes the problem statement including some definitions and assumptions. It is followed by Section 3 where the area search and map building algorithm is proposed. Moreover, a computer simulation is carried out in Section 4 to confirm the performance of the algorithm. Furthermore, the proposed algorithm is assessed by an experiment with a real flying robot in Section 5. Finally, Section 6 presents a brief conclusion including the future work.

## 2. System Description

In this paper, we consider a kinematic model of a flying robot with six system states including position coordinates and attitude angles that is a 3D extension of a 2D Dubins model.[15] The robot moves with a constant speed $v$ in the heading direction $\theta$ (yaw) in horizontal. The horizontal angular velocity $u_\theta$, the vertical velocity $u_z$ and the speed $v(t)$ are three control inputs. This model is widely used to describe many flying robots and UAVs, see, for example, refs.[28–30] Here, we give the mathematical description of this model in a 3D environment as follows (see Fig. 1):

$$\begin{cases} \dot{x}(t) = v(t)\cos\theta(t) \\ \dot{y}(t) = v(t)\sin\theta(t) \\ \dot{z}(t) = u_z(t) \\ \dot{\theta}(t) = u_\theta(t) \end{cases}, \qquad \begin{matrix} x(0) = x_0 \\ y(0) = y_0 \\ z(0) = z_0 \\ \theta(0) = \theta_0 \end{matrix}. \tag{1}$$

In the robot model (1), $(x, y, z)$ are the Cartesian coordinates of the vehicle. The $z$-axis of the global coordinate system is in the opposite direction of the gravity. This 3D robot needs to pitch up and down to maneuver. It is achieved by the control input $u_z(t)$.

The angular velocity $u_\theta$ satisfies the following constraint:

$$|u_\theta(t)| \le u_M. \tag{2}$$

Furthermore, the robot speed $v(t)$ satisfies the constraint:

$$0 \le v(t) \le v_M. \tag{3}$$

Here $u_M > 0$ and $v_M > 0$ are given constants.

The flying robot travels within an unknown underground mine with some tunnels. The underground mine environment can be defined as follows.

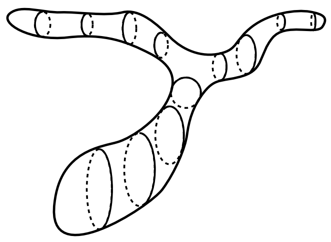


Fig. 2. An underground mine area $\mathcal{A}$.

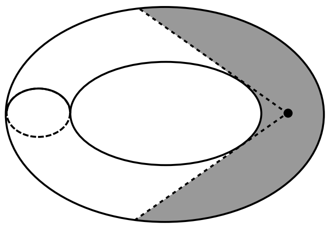


Fig. 3. The set $\mathcal{M}(t) \subset \mathcal{A}$ seen by the robot at time $t$.

**Definition 2.1.** *An underground mine environment is a closed, bounded and connected point set $\mathcal{A} \subset \mathbb{R}^3$. The boundary $\partial\mathcal{A}$ of $\mathcal{A}$ is smooth and consists of ground and walls of the tunnels. Other points of $\mathcal{A}$ are unoccupied (see Fig. 2).*

**Notation 2.1.** *For any point $p \in \mathbb{R}^3$ and any closed set $D \subset \mathbb{R}^3$, the minimum distance $\rho(p, D)$ from p to D is introduced as follows:*

$$\rho(p, D) := \min_{q \in D} \|p - q\|. \tag{4}$$

Let $d_s > 0$ be a given safety margin. The safety requirement is to drive the flying robot through the collision-free part of the area $\mathcal{A}$ while keeping the safety margin $d_s$ from the boundary $\partial\mathcal{A}$; see the following definition.

**Definition 2.2.** *Consider any time period $[0, t_f]$, where $t_f > 0$. Let $p(t) = (x(t), y(t), z(t))$, $d(t) = \rho(p(t), \partial\mathcal{A})$. If for any time $t \in [0, t_f]$, the inequality $d(t) \geq d_s$ holds, the robot's trajectory $p(t)$, $t \in [0, t_f]$ is called collision-free navigation of the flying robot.*

In our method, the flying robot is equipped with a self-positioning system and a 3D obstacle-sensing system. The self-positioning system can access the location $(x(t), y(t), z(t))$ and robot heading $\theta(t)$ at any time $t$, and the obstacle-sensing system combining with the self-positioning system obtains at any time $t$ a set $\mathcal{M}(t) \subset \mathcal{A}$, which is a point set consisting from all points of $\mathcal{A}$ such that the straight line segments from those points to the robot do not intersect with $\partial\mathcal{A}$. It means, $\mathcal{M}(t)$ indicates the part of the environment $\mathcal{A}$ which can be sensed straightly by the robot at time $t$ (see Fig. 3).

While the robot is navigated by the algorithm, the objective of the robot navigation is to search the underground environment and build a 3D map $\mathcal{V}$ of the unknown underground mine area $\mathcal{A}$. The 3D map $\mathcal{V}$ of the environment obtained by the robot a time period $[0, t_f]$ is as follows:

$$\mathcal{V} = \bigcup_{t=0}^{t_f} \mathcal{M}(t). \tag{5}$$

**Definition 2.3.** *For a given $t_f > 0$. If $\mathcal{A}$ and $\mathcal{V}$ are congruent, the time $t_f$ is called a map building completing time and $\mathcal{V}$ is called a complete 3D map.*

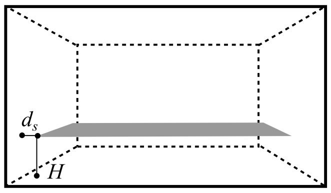


Fig. 4. The traveling surface $\mathcal{S}$ within $\mathcal{A}$.

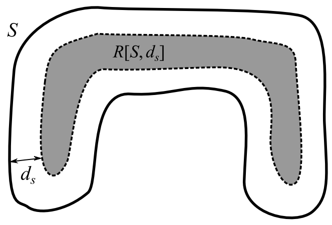


Fig. 5. The $d_s$-reduction of $\mathcal{S}$.

To guarantee the success of the safe navigation, some assumptions are presented as follows:

**Assumption 2.1.** *The initial position of the robot is $3d_s$ far away from the tunnels' walls and ground.*

**Assumption 2.2.** *The safety margin $d_s$ satisfies $d_s > \frac{v_M}{u_M}$.*

## 3. Navigation Algorithm

In this section, we propose a 3D safe area search algorithm for the robot model (1). The proposed algorithm is an extension of a 2D area search and map building algorithm for ground mobile robot.[31] Firstly, we consider the robot flying with a given constant height $H \geq d_s$ above the ground and introduce the following definition.

**Definition 3.1.** *A traveling surface $\mathcal{S}$ of the robot consists of all the points $p \in \mathcal{A}$ such that the minimum distance from $p$ to $\partial\mathcal{A}$ in the direction of gravity is equal to $H \geq d_s$ and $\rho(p, \partial\mathcal{A}) \geq d_s$ (see Fig. 4).*

Then, we introduce some assumptions and propose the navigation algorithm based on these assumptions.

**Assumption 3.1.** *The initial position $(x_0, y_0, z_0)$ belongs to the traveling surface $\mathcal{S}$.*

**Assumption 3.2.** *The traveling surface $\mathcal{S}$ is embedded in a horizontal plane.*

Assumption 3.2 is introduced to simplify the description and mathematical analysis of the proposed navigation algorithm. The algorithm is still valid in more general cases where Assumption 3.2 does not hold, such as any non-self-intersecting traveling surface $\mathcal{S}$ with gentle slope.

To describe the algorithm, the following definitions and assumptions are introduced in advance.

**Definition 3.2.** *There are two circles with the radius $R := \frac{v_0}{u_M}$ on the plane $\mathcal{S}$ that cross the initial robot position $(x_0, y_0, z_0)$ and tangent to the robot initial heading $\theta_0$. Here $v_0$ is some constant speed $0 < v_0 \leq v_M$. The two circles are called initial circles.*

**Definition 3.3.** *A $d_s$-reduction of $\mathcal{S}$ is defined as the set $\mathcal{R}[\mathcal{S}, d_s] := \{p \in \mathcal{S} : \rho(p, \partial\mathcal{S}) \geq d_s\}$. The boundary of $\mathcal{R}[\mathcal{S}, d_s]$ is denoted by $\partial\mathcal{S}(d_s)$ which consists of one or more non-self-intersecting and closed curves (see Fig. 5).*

**Definition 3.4.** *A tangent segment set $\mathcal{L}$ is defined as a set of all the common tangent line segments, each of which is tangent to one or two curves in $\partial\mathcal{S}(d_s)$, or tangent to an initial circle and a curve in $\partial\mathcal{S}(d_s)$ at its two different end points, respectively, and does not intersect with $\partial\mathcal{S}$. The two end points are called tangent points.*

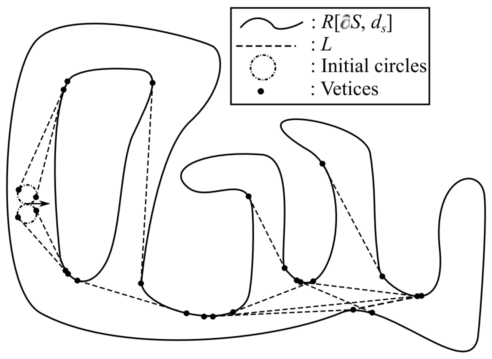


Fig. 6. The graph $\mathcal{G}$.

**Definition 3.5.** *A navigation graph $\mathcal{G}$ is introduced that its vertices are the end points of the tangent segments in $\mathcal{L}$ and its edges are the tangent segments belonging to $\mathcal{L}$, the arcs of the initial circles and the segments of the curves belonging to $\partial\mathcal{S}(d_s)$ (see Fig. 6).*

**Definition 3.6.** *z On the graph $\mathcal{G}$, if the robot is at a vertex, which is an end point of a tangent segment, and the robot's heading coincides with the direction from the end point where the robot is to another end point of the tangent segment, the robot's direction is called exit direction.*

Now we are here to propose the safe navigation algorithm as follows.

**A1:** Initially, the robot selects a initial circle randomly and starts traveling along the initial circle with the constant speed $v_0$.

**A2:** When the robot reaches a tangent point on the initial circle with an exit direction, the robot starts traveling along the corresponding tangent segment with some arbitrary speed $v(t)$.

**A3:** When the robot reaches another tangent point belonging to the tangent segment the robot is traveling along, the robot starts traveling along $\partial\mathcal{S}(d_s)$ in the tangent direction with some arbitrary speed $v(t)$.

**A4:** When the robot reaches a vertex on $\partial\mathcal{S}(d_s)$ with an exit direction, the robot, with a probability $p$, $0 < p < 1$, starts traveling along the corresponding tangent segment, and with a probability $1 - p$, the robot continues traveling along $\partial\mathcal{S}(d_s)$ with some arbitrary speed $v(t)$. Then the robot travels by following the rules **A3** and **A4** cyclically.

The robot is traveling on some traveling 2D surface most of the time. However, it builds a 3D map of the unknown environment which makes the problem a 3D problem.

To introduce the mathematical analysis, the following assumptions are presented in advance.

**Assumption 3.3.** *For any two vertices a and b, and any valid direction $\vec{a}$ at vertex a on the graph $\mathcal{G}$, there exists a smooth path connecting a and b and its direction at vertex coincides with the direction $\vec{a}$.*

**Assumption 3.4.** *For any point $p \in \mathcal{A}$, there exists a point $q \in \partial\mathcal{S}(d_s)$ such that the straight line segment between p and q does not intersect with $\partial\mathcal{A}$.*

**Assumption 3.5.** *There exists at least one tangent segment connecting any of two initial circles and $\partial\mathcal{S}(d_s)$ with a valid exit direction the robot can reach while traveling along the initial circle.*

Assumptions 3.3–3.5 are introduced for rigorous mathematical analysis of the proposed navigation algorithm. The algorithm is still valid in practical implementations if these assumptions do not hold. Furthermore, it should be pointed out that it follows from the standard graph theory and Definition 3.6 that if the traveling surface is convex then the Assumptions 3.3–3.5 automatically hold.

Then the main theoretical result is as follows.

**Theorem 1.** *Suppose Assumptions 2.1, 2.2 and 3.1–3.5 hold, and the robot (1) is navigated by Algorithms* **A1**–**A4** *with a probability p. Then, for any initial position and heading, with probability 1, there exists a map building completing time $t_f \geq 0$ such that the robot's trajectory during period $[0, t_f]$ is collision-free navigation with a complete 3D map $\mathcal{V}$.*

**Proof of Theorem 1:** According to Assumptions 2.1, 2.2, 3.1 and 3.2, the robot's trajectory is collision-free navigation while traveling along any of two initial circles in Algorithm **A1**. According to Assumption 3.5, Algorithms **A2** and **A3** are able to be switched to successively from Algorithm **A1** and reach $\partial\mathcal{S}(d_s)$. Then, according to Assumption 2.2, it can be seen that the robot (1) is able to travel along $\partial\mathcal{S}(d_s)$ with the non-holonomic constraint. It follows from Definitions 3.3 and 3.4 that the robot is always at a distance greater than the safety margin $d_s$ while traveling along the tangent segment or $\partial\mathcal{S}(d_s)$ in Algorithms **A2**, **A3** and **A4**. Hence, the robot's trajectory is collision-free navigation. Moreover, according to Assumption 3.3 and the above analysis, any point $p \in \partial\mathcal{S}(d_s)$ can be reached by the robot with a non-zero probability under the navigation Algorithms **A1**–**A4** from any initial position and heading. Finally, the robot switches from some navigation mode to another with some probability $0 < p < 1$; hence, with probability 1, the robot will eventually move along any tangent segment or boundary segment. Therefore, Assumption 3.4 and above analysis of the collision-free navigation imply that, with probability 1, there exists a map building completing time $t_f \geq 0$ such that the robot's trajectory during period $[0, t_f]$ is collision-free navigation with a complete 3D map $\mathcal{V}$.

Based on the proposed navigation algorithm, we design a corresponding motion controller to drive the flying robot. Firstly, we extract three types of measurements from the total measurement $\mathcal{M}(t)$ by the following definitions.

**Definition 3.7.** *For any $t$, the distance from the robot's position $(x(t), x(t), z(t))$ to the total measurement $\mathcal{M}(t)$ in the direction of the gravity is called flying height, denoted by $h(t)$.*

**Definition 3.8.** *For any $t$, let $U$ be a horizontal plane through the robot's position $(x(t), x(t), z(t))$. The set $\mathcal{N}(t) = U \cap \mathcal{M}(t)$ is called horizontal measurement.*

The horizontal measurement $\mathcal{N}(t)$ should be extracted as finite pairs of polar coordinates $(\alpha_i, r_i)$, $i = 1, 2, \ldots, n$ in the robot's body frame, that is, the reference direction of the polar coordinate system coincides with the robot's heading $\theta(t)$. For any $i > 1$, we assume $\alpha_i > \alpha_{i-1}$.

With the above measurements, we consider a simple proportional controller to drive the robot flying with the given height $H$ above the ground. The proportional controller is proposed as follows:

$$u_z(t) = g(H - h(t)), \tag{6}$$

where $g > 0$ is a tunable constant. This controller can also be replaced by other more complex controllers in practical implementations.

For the motion control in the horizontal plane described in Algorithms **A1**–**A4**, we developed a control law based on a control law presented in ref. 32. The control law includes three motion control strategies. The first is an initial circle traveling control that the flying robot moves with the maximum angular velocity on horizontal plane. The second is a boundary following control that the robot travels along a $d_s$-reduced boundary; see ref.[33] The third one is a target tracking navigation; see for example ref.[34] Then, our sliding mode navigation law can be expressed as follows:

$$u_\theta(t) = \begin{cases} \pm u_M & R1 \\ \mathrm{sgn}\,[\phi(t)]\,u_M & R2 \\ \Gamma \mathrm{sgn}\left[\dot{d}(t) + X(d(t) - d_s)\right] u_M & R3 \end{cases}, \tag{7}$$

where the function $\mathrm{sgn}(x)$ is defined as follows:

$$\mathrm{sgn}(x) = \begin{cases} 1 & x > 0 \\ 0 & x = 0 \\ -1 & x < 0 \end{cases}, \tag{8}$$

and the saturation function $X(x)$ is defined as follows:

$$X(x) = \begin{cases} lx & |x| < k \\ lk\, sgn(x) & \text{otherwise} \end{cases}, \tag{9}$$

where $l$ and $k$ are tunable constants.

Mode $R1$ describes the motion of traveling along an initial circle with maximum angular velocity. Mode $R2$ describes the traveling along a tangent segment, where $\phi(t)$ is defined as the direction of the intermediate target $T_m$ relative to the robot's body frame. Mode $R3$ describes the boundary following navigation, where the calculation is based on the minimum obstacle distance $d(t)$.

This navigation law defined three control strategies corresponding to three separate modes $R1 - R3$. There are three rules for switching the mode between three modes $R1 - R3$. Initially, mode $R1$ is active, and transitions to other modes are determined as follows:

1. $R1 \to R2$: it occurs when the robot reaches a tangent point on one of two initial circles with a valid exit direction, that is, in the horizontal measurement $\mathcal{N}(t)$, there exist two points $(\alpha_i, r_i)$ and $(\alpha_{i+1}, r_{i+1})$ such that the following inequalities $|\alpha_i| < \bar{\alpha}$, $|\alpha_{i+1}| < \bar{\alpha}$, $|r_i - r_{i+1}| < \bar{r}$ hold, where $\bar{\alpha} > 0$ and $\bar{r} > 0$ are two thresholds. Then, an intermediate target in the robot's local polar coordinate system is calculated as follows:[35]
$$T_m = \begin{bmatrix} \frac{\alpha_i+\alpha_{i+1}}{2} + \arctan\left(\frac{\operatorname{sgn}(r_{i+1}-r_i)d_s}{\min\{r_i, r_{i+1}\}}\right) \\ \sqrt{d_s^2 + \min\{r_i, r_{i+1}\}^2} \end{bmatrix}. \qquad (10)$$
Furthermore, $T_m$ is transformed into the global Cartesian coordinate system for the following navigation in mode $R2$ and the variable $\Gamma = \operatorname{sgn}(r_i - r_{i+1})$ is calculated.
2. $R2 \to R3$: it occurs when the robot finishes a traveling along a tangent segment and reaches the intermediate target $T_m$, that is, $|T_m - p(t)| < \bar{e}$, where $p(t)$ is the robot's current position and $\bar{e} > 0$ is a threshold.
3. $R3 \to R2$: with the given probability $p$, it occurs when the robot reaches a tangent point on $\partial\mathcal{S}(d_s)$ with a valid exit direction. Then the intermediate target $T_m$ and the variable $\Gamma$ are calculated as above.

Because of the chattering in the robot's heading caused by the sliding mode control, once it is decided to not pursue a tangent line, there is a short pause that tangent following mode cannot be engaged again during the pause. It means, during this short pause, the mode $R3$ will not switch to $R2$ even if the robot's heading coincides with a tangent line again. In other words, if there are two tangents points on the robot's trajectory that are very close to each other, the robot ignores the second one. Such situations are very rare.

With the control laws (6) and (7) and the mode transition rules, the flying robot (1) can be navigated as the description of the proposed Algorithms **A1–A4**.

## 4. Computer Simulations

To confirm the performance of our algorithm in the tunnels of the underground mine, we carry out some computer simulations in V-REP simulation environment with a quadcopter robot model. In the first simulation, a complex tunnel environment with branches and cyclical routes is built in a flat ground. The tunnels are 3 m high and the parameters used in the simulation are indicated in Table I. The simulation result is shown in Fig. 7. It can be seen that the quadcopter starts traveling from a random initial position and through the tunnels with several times of the mode transition. The quadcopter avoids the collisions with the tunnel walls and builds a complete 3D map.

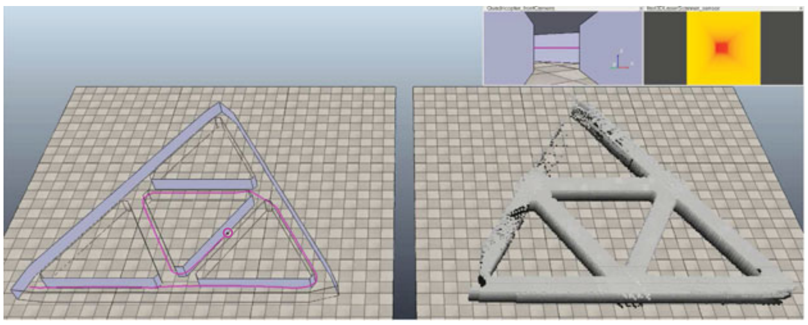
We also carry out the second simulation to indicate the performance of our algorithm in a sloping tunnel environment indicated by Remark 3.1. In this simulation, a tunnel environment with a branch is built and some of the tunnel segments are 15° sloping. The parameters used for the control laws are the same as the first simulation in Table I. Figure 8 shows the simulation result. It can be seen that the quadcopter travels from an initial position and through the flat and sloping tunnels without any collision. While traveling in the environment, the robot was building a complete 3D map presented by an octree.

Moreover, we study the case of using several UAVs in underground mine environments. The use of several UAVs can accelerate the area search and map building since the proposed navigation algorithm belongs to the class of random area exploration algorithm. Firstly, to drive distributed drones simultaneously in a tunnel area without collision, an additional control strategy is proposed here to develop the proposed navigation algorithm for multiple UAVs collision-free navigation. In the

Table I. The parameters used in the simulation.

| | |
|---|---|
| $v$ | 0.51 m/s |
| $u_M$ | 0.3 rad/s |
| $H$ | 1.5 m |
| $d_s$ | 1 m |
| $\bar{\alpha}$ | 0.05 rad |
| $\bar{r}$ | 2 m |
| $\bar{e}$ | 0.1 m |
| $P$ | 0.3 |
| $g$ | 0.5 |
| $l$ | 0.3 |
| $k$ | 1 |
| Sampling period | 0.05 s |

(a)



(b)

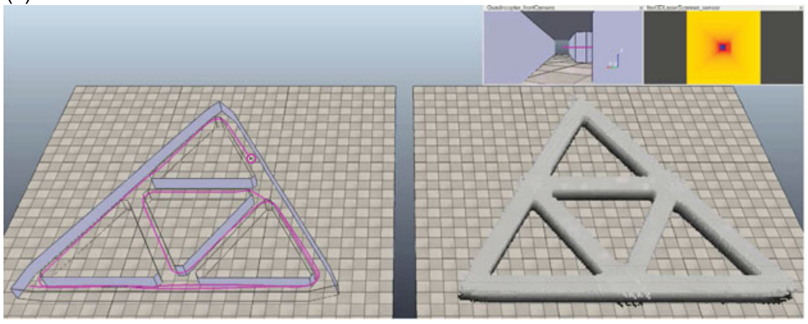


Fig. 7. The simulation result in the complex tunnel environment. The left half shows the tunnels and the robots' trajectory. The right half shows the complete 3D map presented by an octree. The gray scale of each voxel in the octree indicates its height. The right-top pictures are the colourful image and depth image in the front face.

additional control strategy, we modify the constant height $H$ to a variable $H(t)$ satisfying inequality $d_s \leq H(t) \leq d_M - d_s$, where $d_M > d_s$ is the minimum distance from the ground to the top of the tunnel areas. Then, the multiple drones are labeled with ID numbers 1, 2, .... Then we define a range $d_r > d_s$ so that any two drones between which the distance is smaller than $d_r$ can exchange information by wireless communication and may collide. Then, the drones can be divided into several groups, in each of which the drones build a local communication network to share their locations associated with their IDs. Generally, due to the randomness involved in the proposed navigation algorithm, we assume that the number of drones $n > 0$ in any group is always smaller than an integer $N > 0$ and $d_M \geq (N + 1)d_s$. Now, for any drone $i$, we propose the following strategy to avoid the collision between drones in a group:

$$H(t) = m\frac{d_M}{n+1}, \tag{11}$$

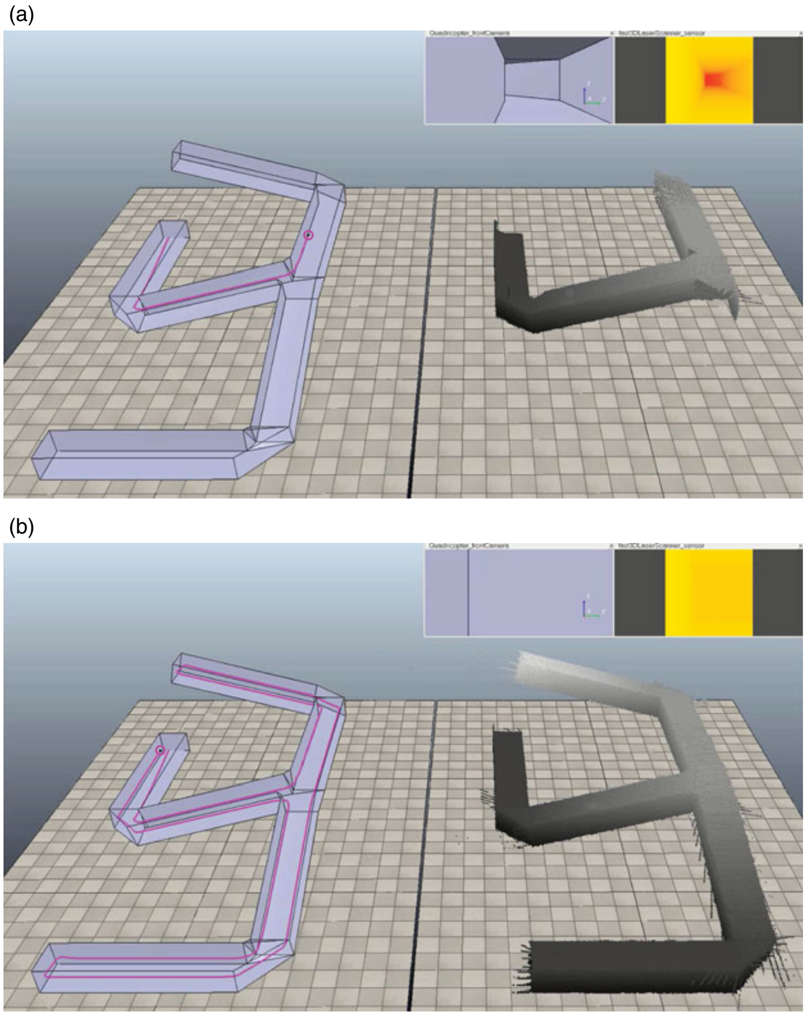


Fig. 8. The simulation result in the sloping tunnel environment.

where $m$ is the number of drones labeled with the IDs larger than drone $i$ in a group. With the control strategy (11), the drones can autonomously select the safe flying height during traveling simultaneously with other drones and avoid any possible collision when they are closer than $d_r$.

To confirm the proposed additional control strategy, we carry out the third simulation with three drones in the same tunnel environment used in the first simulation. In this simulation, $d_r = 4d_s$ and other parameters are indicated in Table I. Figure 9 shows the simulation result that shows the trajectories of three drones and indicates the effect of the control strategy (11). We can see that when any two drones encounters, the altitude of those drones will changes to different levels to avoid the collision.

## 5. Experiments with a Real Flying Robot

In this section, we carry out an experiment with a real flying robot to confirm the performance of our navigation algorithm in practical implementations. In the experiment, a DJI Matrice 100 quadcopter equipped with a DJI Guidance sensing system,[18] a DJI Manifold onboard computer, an Intel D435 depth camera and a LattePanda onboard computer is used as the flying robot to perform the autonomous area search and 3D map building in an indoor environment. Figure 10 shows a picture of our flying robot with annotations. The diagonal wheelbase of the quadcopter is 650 mm and the total weight is approximately 3.1 kg.

The DJI Matrice 100 quadcopter includes four motors and inertial measurement unit connected with a flight controller, which provides the lower-level motion control of the quadcopter, such as 3-axis speed control and yaw control. The DJI Guidance sensing system is an advanced obstacle

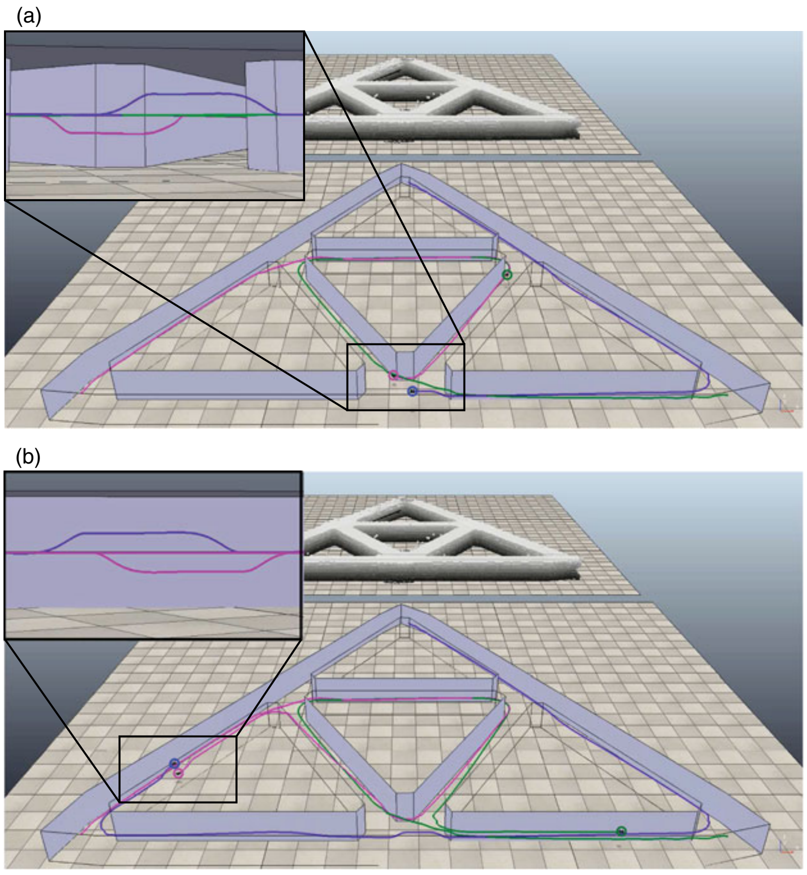


Fig. 9. The simulation result with multiple drones.

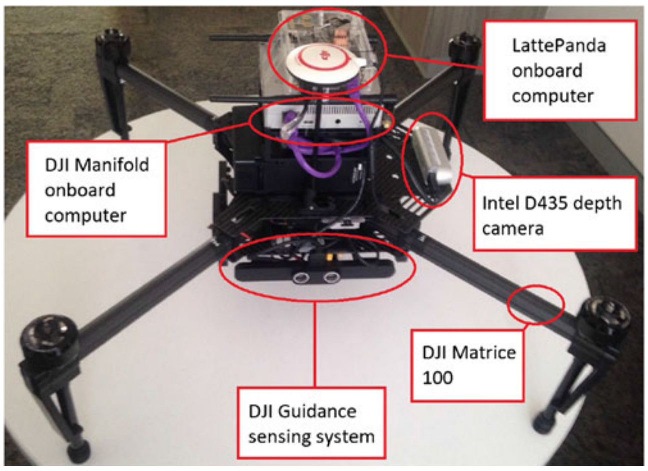

LattePanda onboard computer
DJI Manifold onboard computer
Intel D435 depth camera
DJI Matrice 100
DJI Guidance sensing system

Fig. 10. The flying robot used in the experiment.

sensing system integrating stereo visual cameras and ultrasonic sensors. With an individual vision process module, it provides an accurate visual odometry and the minimum obstacle distance in five directions (bottom, front, left, back and right). The Intel D435 depth camera faces the front direction of the quadcopter and provides the real-time 3D measurements of the environment in a field of view of $86^\circ \times 57^\circ \times 94^\circ$ (Horizontal × Vertical × Diagonal). The LattePanda onboard computer is connected by cable with the Intel D435 depth camera and drives the Intel D435 depth camera individually. The LattePanda onboard computer is used to receive the real-time position coordinates and attitude angles and provide the coordinates of the real-time 3D point cloud in the global coordinate

Table II. The parameters used in the experiment

| | |
|---|---|
| $v_c$ | 0.1 m/s |
| $u_M$ | 0.3 rad/s |
| $H$ | 0.7 m |
| $d_s$ | 1.3 m |
| $\bar{\alpha}$ | 0.2 rad |
| $\bar{r}$ | 0.5 m |
| $\bar{e}$ | 0.2 m |
| $d_c$ | 0.9 m |
| $P$ | 0.7 |
| $g$ | 1 |
| $l$ | 2 |
| $k$ | 1 |
| Sampling period | 0.1 s |

system. The DJI Manifold onboard computer is the core component of the robot. It is connected by cables with the flight controller of the DJI Matrice 100 quadcopter, the DJI Guidance sensing system and the LattePanda onboard computer.

For any time $t$, the DJI Manifold onboard computer can retrieve the position coordinates $(x(t), y(t), z(t))$, the attitude angles including the yaw $\theta(t)$, the flying height $h(t)$ and the minimum obstacle distance $d(t)$ from the DJI Guidance sensing system. Furthermore, for any time $t$, the DJI Manifold onboard computer can also retrieve the real-time 3D point cloud $\mathcal{M}(t)$ in the global Cartesian coordinate system and the horizontal measurement $\mathcal{N}(t)$ in the robot's local polar coordinate system. Then, with the above measurements, the DJI Manifold onboard computer is programmed with the proposed control laws (6) and (7) and the mode transition rules. It is also able to communicate with a ground computer by WiFi to receive the start and stop commands, and send the real-time measurement $\mathcal{M}(t)$. In the ground computer, the real-time measurements $\mathcal{M}(t)$ are received and integrated into the total 3D map $\mathcal{V}$.

The experiment was conducted in a meeting room. In the meeting room, a closed experiment area is enclosed by some boards to construct a segment of a tunnel with terminals. Moreover, considering the real quadcopter model and the safety problems caused by unknown disturbance in practical implementations, we added the following control law as the speed $v$ is controllable for a real quadcopter:

$$v(t) = \begin{cases} 0 & R1 \text{ or } (R3 \text{ and } d_f(t) < d_c) \\ v_c & \text{otherwise} \end{cases}, \tag{12}$$

where $d_f(t)$ is the minimum obstacle distance only in the front, left and right faces, $d_c < d_s$ is a critical safety distance and $v_c$ is a given constant speed. Furthermore, in this experiment, considering the height of the boards used to enclose the experiment area, we calculate the minimum obstacle distance $d(t)$ by only using the measurements in front, left, back and right faces. Then, the main parameters used in this experiments are listed in Table II. The safety margin is determined according to the size of the experimental environment. Then, the speed and maximum angular velocity are chosen to make Assumption 2.2 hold and have a safe experimental process in a small workspace. For real underground mine environment, the size of the workspace is much larger than the experimental environment; therefore, the speed can be faster than what we choose in the experiment, and the angular acceleration should be satisfactory for practical implementations.

The experiment results are shown in Figs. 11 and 12. In Fig. 11, it is seen that, at the beginning, the robot was turning around with initial mode $R1$. Then the robot detected a tangent line coinciding with the robot's heading and mode transitions to $R2$. With the mode $R2$, the robot was traveling along the tangent line and reach the tangent point. Next, the mode transitions to $R3$ and the robot was traveling along the boundary of the tunnel walls. While the robot was traveling through the area, the real-time 3D point cloud was integrated in the ground computer to generate the 3D map of the environment (see Fig. 12). The experiment confirms the performance of our navigation algorithm in the practical implementations of a real quadcopter. With the analysis of the 3D map shown in Fig. 12, some incremental errors resulted by the error of the pose estimation can be found. These map errors

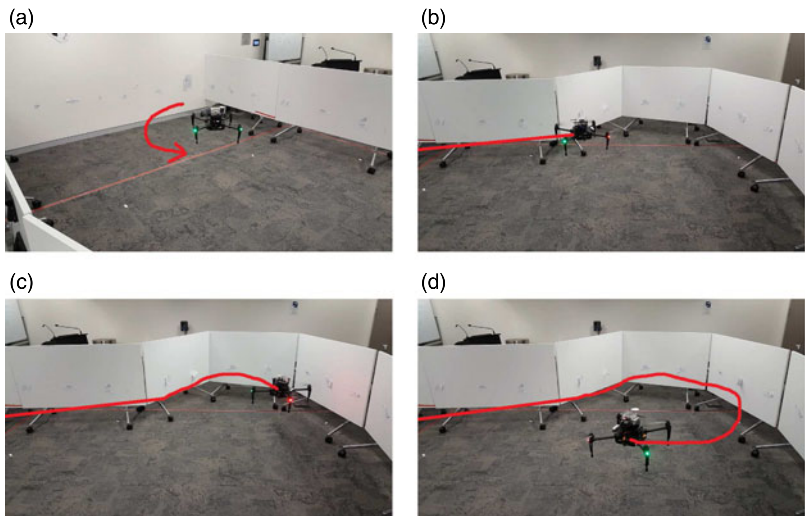


Fig. 11. The pictures of the experiment at different instants.

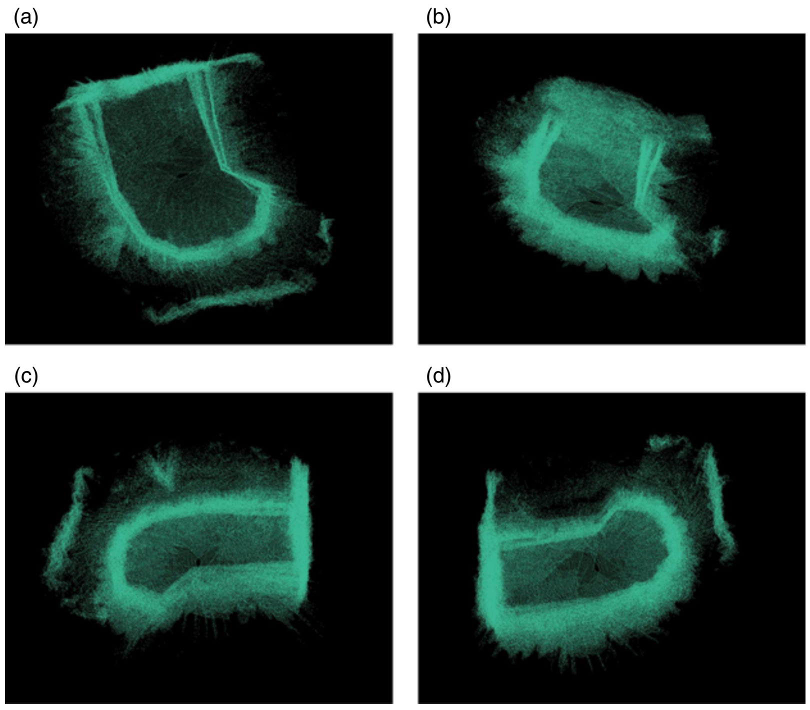


Fig. 12. The final 3D point cloud map in the experiment at different points of view.

can be reduced by combining SLAM simultaneous localization and mapping (SLAM) algorithms and do not affect the performance of the collision-free navigation because the navigation algorithm is independent of mapping process.

## 6. Conclusion

In this paper, we propose a safe navigation algorithm for a flying robot to explore an underground mine area and build a 3D map of the environment without collisions with tunnel walls. The robot model we considered is an extension of a 2D non-holonomic model. The horizontal motion control and altitude control are separated in our method. In our navigation algorithm, the measurements we used are the minimum distance to obstacles and the horizontal scan of obstacles. The computer simulations show that the performance of the proposed algorithm is as we expect in both flat ground and sloping ground. The simulations also show the use of the proposed algorithm with multiple drones. Furthermore, we verify our method with a real flying robot in an indoor environment. The experiment shows that the proposed navigation algorithm is very robust when applied to real flying robots in narrow tunnel-like environment.

The proposed algorithm is simple and reliable for practical applications. It performs an autonomous tunnel area search and 3D map building with flat or sloping grounds. The measurements and sensors we considered in the proposed algorithm work well in practical implementations. With the presented sliding mode control law, the presented algorithm can be applied to different drones with strong robustness. The performance of the algorithm is also mathematically proved. The presented method can be used in a lot of tunnel environments, such as underground mine tunnels, traffic tunnels and cave tunnels. In our future work, we will particularly consider the mapping accuracy analysis of flying robot mapping problem, and the SLAM (simultaneous localization and mapping) algorithms can be combined to improve the quality of the map in large scenes, like sewage tunnels or oil pipelines.

## References

1. S. Sankarasrinivasan, E. Balasubramanian, K. Karthik, U. Chandrasekar and R. Gupta, "Health monitoring of civil structures with integrated uav and image processing system," *Proc. Comput. Sci.* **54**, 508–515 (2015).
2. S. Siebert and J. Teizer, "Mobile 3D mapping for surveying earthwork projects using an unmanned aerial vehicle (UAV) system," *Auto. Constr.* **41**, 1–14 (2014).
3. C. M. Gevaert, J. Suomalainen, J. Tang and L. Kooistra, "Generation of spectral–temporal response surfaces by combining multispectral satellite and hyperspectral UAV imagery for precision agriculture applications," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **8**(6), 3140–3146 (2015).
4. S. Lee and Y. Choi, "Reviews of unmanned aerial vehicle (drone) technology trends and its applications in the mining industry," *Geosyst. Eng.* **19**(4), 197–204 (2016).
5. H. Li and A. V. Savkin, "Wireless sensor network based navigation of micro flying robots in the industrial internet of things," *IEEE Trans. Ind. Inf.* **14**(8), 3524–3533 (2018).
6. V. Nazarzehi and A. V. Savkin, "Distributed self-deployment of mobile wireless 3D robotic sensor networks for complete sensing coverage and forming specific shapes," *Robotica* **36**(1), 1–18 (2018).
7. C. Wang, A. V. Savkin and M. Garratt, "A strategy for safe 3D navigation of non-holonomic robots among moving obstacles," *Robotica* **36**(2), 275–297 (2018).
8. S. Scherer, J. Rehder, S. Achar, H. Cover, A. Chambers, S. Nuske and S. Singh "River mapping from a flying robot: state estimation, river detection, and obstacle mapping," *Auto. Robots* **33**(1), 189–214 (2012).
9. A. Lucieer, S. M. de Jong and D. Turner, "Mapping landslide displacements using structure from motion (SfM) and image correlation of multi-temporal UAV photography," *Prog. Phys. Geogr. Earth Environ.* **38**(1), 97–116 (2014).
10. A. Nemra and N. Aouf, "Robust Feature Extraction and Correspondence for UAV Map Building," *2009 17th Mediterranean Conference on Control and Automation*, Thessaloniki, Greece (2009) pp. 922–927.
11. I. Dryanovski, W. Morris and J. Xiao, "Multi-Volume Occupancy Grids: An Efficient Probabilistic 3D Mapping Model for Micro Aerial Vehicles," *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Taipei, Taiwan (2010) pp. 1553–1559.
12. A. Bircher, M. Kamel, K. Alexis, H. Oleynikova and R. Siegwart, "Receding horizon path planning for 3D exploration and surface inspection," *Auto. Robots* **42**(2), 291–306 (2018).
13. S. Song and S. Jo, "Online Inspection Path Planning for Autonomous 3D Modeling Using a Micro-aerial Vehicle," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, Singapore (2017) pp. 6217–6224.
14. S. Shen, N. Michael and V. Kumar, "Stochastic differential equation-based exploration algorithm for autonomous indoor 3D exploration with a micro-aerial vehicle," *Int. J. Robot. Res.* **31**(12), 1431–1444 (2012).
15. L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *American J. Math.* **79**(3), 497–516 (1957).
16. L. R. García Carrillo, A. E. Dzul López, R. Lozano and C. Pégard, "Combining stereo vision and inertial navigation system for a quad-rotor UAV," *J. Intell. Robot. Syst.* **65**(1), 373–387 (2012).
17. F. Caballero, L. Merino, J. Ferruz and A. Ollero, "Vision-based odometry and slam for medium and high altitude flying UAVs," *J. Int. Robot. Syst.* **54**(1), 137–161 (2009).
18. G. Zhou, L. Fang, K. Tang, H. Zhang, K. Wang and K. Yang, "Guidance: A Visual Sensing Platform for Robotic Applications," *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Boston, Massachusetts, USA (2015) pp. 9–14.
19. F. L. Lô Medeiros and J. D. S. da Silva, "A Dijkstra Algorithm for Fixed-wing UAV Motion Planning Based on Terrain Elevation," **In**: *Advances in Artificial Intelligence – SBIA 2010* (Springer, Berlin, Heidelberg, 2010) pp. 213–222.

20. F. Yan, Y. S. Liu and J. Z. Xiao, "Path planning in complex 3D environments using a probabilistic roadmap method," *Int. J. Auto. Comput.* **10**(6), 525–533 (2013).
21. L. Lu, C. Zong, X. Lei, B. Chen and P. Zhao, "Fixed-Wing UAV Path Planning in a Dynamic Environment via Dynamic RRT Algorithm," **In**: *Mechanism and Machine Science* (Springer, Singapore, 2017) pp. 271–282.
22. L. J. Guibas, D. Hsu, H. Kurniawati and E. Rehman, *Bounded Uncertainty Roadmaps for Path Planning* (Springer, Berlin, Heidelberg, 2010) pp. 199–215.
23. G. Kewlani, G. Ishigami and K. Iagnemma, "Stochastic Mobility-Based Path Planning in Uncertain Environments," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems IROS 2009*, St. Louis, Missouri, USA (2009) pp. 1183–1189.
24. M. Kothari and I. Postlethwaite, "A probabilistically robust path planning algorithm for uavs using rapidly-exploring random trees," *J. Intell. Robot. Syst.* **71**(2), 231–253 (2013).
25. M. Hoy, A. S. Matveev and A. V. Savkin, "Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey," *Robotica* **33**(3), 463–497 (2015).
26. A. S. Matveev, A. V. Savkin, M. Hoy and C. Wang, *Safe Robot Navigation Among Moving and Steady Obstacles* (Elsevier, Amsterdam, Netherlands, 2015).
27. L. Palmieri, S. Koenig and K. O. Arras, "RRT-Based Nonholonomic Motion Planning Using Any-Angle Path Biasing," *2016 IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, Sweden (2016) pp. 2775–2781.
28. R. P. Anderson and D. Milutinović, "A stochastic approach to dubins vehicle tracking problems," *IEEE Trans. Auto. Contr.* **59**(10), 2801–2806 (2014).
29. B. Di, R. Zhou and H. Duan, "Potential field based receding horizon motion planning for centrality-aware multiple UAV cooperative surveillance," *Aeros. Sci. Tech.* **46**, 386–397 (2015).
30. A. V. Savkin and H. Huang, "Optimal aircraft planar navigation in static threat environments," *IEEE Trans. Aerosp. Elect. Syst.* **53**(5), 2413–2426 (2017).
31. A. V. Savkin and H. Li, "A safe area search and map building algorithm for a wheeled mobile robot in complex unknown cluttered environments," *Robotica*, **36**(1), 96–118 (2018).
32. A. V. Savkin and M. Hoy, "Reactive and the shortest path navigation of a wheeled mobile robot in cluttered environments," *Robotica* **31**(2), 323–330 (2013).
33. A. S. Matveev, H. Teimoori and A. V. Savkin, "A method for guidance and control of an autonomous vehicle in problems of border patrolling and obstacle avoidance," *Automatica* **47**(3), 515–524 (2011).
34. A. V. Savkin and H. Teimoori, "Bearings-only guidance of a unicycle-like vehicle following a moving target with a smaller minimum turning radius," *IEEE Trans. Auto. Contr.* **55**(10), 2390–2395 (2010).
35. A. C. Burdette, *Analytic Geometry* (Academic Press, 2014) pp. 144–161.