Aalborg Universitet

AALBORG UNIVERSITY

Comparison of path planning algorithms for an unmanned aerial vehicle deployment under threats

Danancier, Kevin; Ruvio, Delphine; Sung, Inkyung; Nielsen, Peter

Published in:
IFAC-PapersOnLine

DOI (link to publication from Publisher):
10.1016/j.ifacol.2019.11.493

Creative Commons License
CC BY-NC-ND 4.0

Publication date:
2019

Document Version
Publisher's PDF, also known as Version of record

Link to publication from Aalborg University

Citation for published version (APA):
Danancier, K., Ruvio, D., Sung, I., & Nielsen, P. (2019). Comparison of path planning algorithms for an unmanned aerial vehicle deployment under threats. *IFAC-PapersOnLine*, *52*(13), 1978-1983. https://doi.org/10.1016/j.ifacol.2019.11.493

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal -

Take down policy
If you believe that this document breaches copyright please contact us at vbn@aub.aau.dk providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from vbn.aau.dk on: May 02, 2026

Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 52-13 (2019) 1978–1983

Comparison of Path Planning Algorithms for an Unmanned Aerial Vehicle Deployment Under Threats

Kevin Danancier * **Delphine Ruvio** * **Inkyung Sung** ** **Peter Nielsen** **

* *Department of Industrial Engineering, Polytech Engineering School of Aix-Marseille University, France (kevin.danancier@etu.univ-amu.fr, delphine.ruvio@etu.univ-amu.fr)*

** *Operations Research Group, Department of Materials and Production, Aalborg University, Denmark (inkyung_sung@mp.aau.dk, peter@mp.aau.dk)*

Abstract: Following the massive interests in unmanned aerial vehicles (UAVs), various optimization algorithms have been proposed for a path planning problem that allow the units to navigate in a region filled with threats such as a radar detection in air defence systems. Among the algorithms, we address Dijkstra's algorithm and a heuristic algorithm for the path planning of a UAV. The algorithms are compared under various configurations of a region to navigate with respect to the optimality and the computational complexity of the algorithms.

© 2019, IFAC (International Federation of Automatic Control) Hosting by Elsevier Ltd. All rights reserved.

Keywords: Optimization and Control, Scheduling, Operations Research

1. INTRODUCTION

Unlike other automated vehicles, which require a physical line or makers to move, e.g. an automated guided vehicle, an unmanned aerial vehicle (UAV) has less restrictions on its operational environment because of its flexibility in movement. Providing an aerial viewpoint is also an important feature of a UAV, making its application promising and cost-efficient. A military and civil engineering are examples of application areas, where deploying a fleet of UAVs could dramatically reduce the cost and dangers of the operations related, while maximising the performance of the operations.

However, deploying UAVs to such applications brings a challenge that it is often difficult to guarantee the safety and reliability of UAVs during the deployment because an operation area for a UAV deployment often involves obstacles and threats such as a high-rise building and an enemy's air-defence system.

To address the challenge, a path planning problem has been actively studied, which derives a flight path for a UAV from a point to another with respect to navigating a region of interest safely (Alotaibi et al., 2018). Various operational constraints such as the maximum energy level of a UAV and safety distance from an object (e.g. a building or another UAV) are often imposed into the problem.

Following the importance of the path planning in a UAV deployment, various approaches based on exact and heuristic approaches have been proposed (Zhao et al., 2018). Among them, we present two algorithms, Dijkstra's algorithm and a heuristic algorithm we designed.

First, Dijkstra algorithm, a well-known exact approach for a shortest path problem is discussed with an emphasis on how to generate a waypoint network for a region to navigate and a cost for a flight between two waypoints. A waypoint is a coordinate that a UAV may use to change its flight conditions such as a direction and a speed.

One of the drawbacks of Dijkstra's algorithm in the path planning is that it takes a long time to derive a solution as the scale of a network (e.g. the numbers of nodes and arcs in the network) increases, which is not a desired characteristic of a solution method for the path planning, especially in real time.

Motivated by the fact, we present a heuristic algorithm with a waypoint generation scheme. The algorithm first draws a straight line from an origin to a destination and adjusts the line by inserting new waypoints to the line, when the line is overlapped with an obstacle or threats. Since the scheme is simple, the algorithm could find a path quickly. A performance comparison of the two algorithms is then carried out under various scenarios.

Specifically, we generate test scenarios varying a distribution of threats and targets on a region. A threat is a danger a UAV should avoid during it flight. Unlike an obstacle, a UAV can penetrate a threat but it increases the unit's treat level. A UAV also needs to conduct tasks, i.e. to survey targets. The path planning algorithms are then applied to generate flight paths to visit targets avoiding threats. The algorithms are then evaluated in terms of the threat level of paths derived and time to compute the paths. Based on the evaluation, we finally characterize the favourable conditions for each algorithm.

2405-8963 © 2019, IFAC (International Federation of Automatic Control) Hosting by Elsevier Ltd. All rights reserved.
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2019.11.493

2. RELATED WORK

The path planning for UAVs can be categorized into two classes depending on the mission type of a UAV deployment.

First, the path planning problem can be found for a mission, which requires a placement of a UAV on a position, e.g. delivering a material to a site. In this case, a path planning problem is often defined as to find a sequence of waypoints to avoid threats or obstacles (Bertuccelli et al., 2009; Alotaibi et al., 2018). Fig. 1 illustrates a path that allows a UAV to visit multiple points while avoiding collisions with objects.

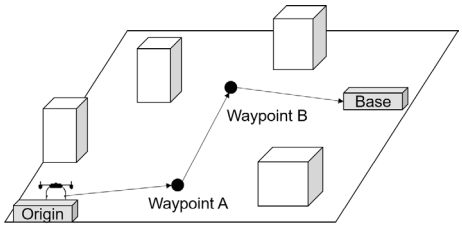

Fig. 1. Illustration of a path

Please refer to Radmanesh et al. (2018); Zhao et al. (2018) for comprehensive reviews on the exact and heuristic approaches for the planning algorithms. The path planning can also be embedded into a scheduling system for a fleet of UAVs. Coupled with a decision that assigns tasks to UAVs, the path planning problem is solved to find a path between tasks (Coutinho et al., 2018).

The path planning problem has also been studied to cover a region, termed as the coverage path planning (CPP). For a region of interest, the CPP consists of 1) a decomposition of the region into a set of sub-polygons, 2) finding a covering direction for each sub-polygon, and 3) determining a sequence of the sub-polygons to form a final covering path (Jiao et al., 2010; Barrientos et al., 2011). The concept of the approach is described in Fig. 2.

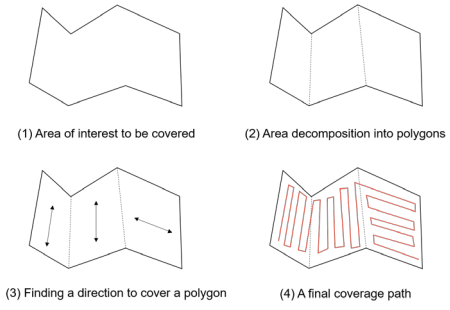

Fig. 2. General steps in the CPP solving

In this paper, we focus on the first type of the path planning problem. Specifically, we address a situation where a UAV needs to visit multiple target points following a given order. A region to navigate is filled with threats, which can damage a UAV. A UAV can fly over or through a threat, if necessary, unlike physical obstacles such as a building, but it increases the treat level of the unit. The objective of the path planning problem is set as to minimise the threat level of a UAV during a flight visiting multiple positions.

To find a path for the problem, we first discuss Dijkstra's algorithm, an exact algorithm for a shortest path problem. Given a network and a function to compute a cost between two nodes, Dijkstra's algorithm derives an optimal path to a destination which minimises the cost of the path. However, it takes a long time as the size of a problem, e.g. the number of nodes in a network, increases. Moreover, the performance of the algorithm is critically depending on the network used for the path planning, which brings an additional decision-making problem for the path planning. A grid network (Zhang et al., 2014) or a set of waypoints generated by heuristic rules (Alotaibi et al., 2018) are generally used to construct the network.

To overcome the challenges, we develop a simple heuristic algorithm expected to produce a path with an acceptable performance level within a short time. The performance of the algorithm is compared to that of Dijkstra's algorithm under various scenarios. Finally, favourable conditions for each algorithm are investigated.

Note that a region of interest in this paper is represented as a two-dimensional space. Although, actual operational environment of a UAV is a three-dimensional space, the path planning in two-dimensional environments has been extensively studied due to its simplicity with an assumption that a UAV flies by maintaining its altitude or manual adjustment (Zhao et al., 2018). For the path planning in three-dimensional environment, please refer to Yang et al. (2014); Ahmad et al. (2017).

3. PATH PLANNING ALGORITHMS

Let us first define a path for a UAV as a sequence of waypoints. Recall that a waypoint is a coordinate that a UAV may use to change its flight direction. Threats, which damage a UAV, and targets, which will be surveyed by a UAV, are distributed on a region of interest. The sequence of the targets to survey is given.

With the setting, the path planning problem in this paper is defined as to find a flight path that minimises the exposure of a UAV to threats, while surveying targets following a given order. The assumptions made in this paper are listed as follows:

- Threats are the only factors of risk;
- A threat is modelled as a stationary circle area;
- The locations of threats and targets are known *a priori*;
- The threat level to UAV is proportional to its flight time over threats and distance to threats;
- A UAV flies at a same speed during its flight.

To solve the problem, we implement Dijkstra's and heuristic algorithms.

3.1 Dijkstra's Algorithm

Given σ, a sequence of targets to survey, we first form a set of pairs of targets specifying an origin and a destination of a sub-path. For an origin and a destination of a sub-path, Dijkstra's algorithm is applied to find a path between them. For example, for sequence $b - t_1 - t_2 - b$, where b is a base for a UAV, pairs of waypoints, (b, t_1), (t_1, t_2) and (t_2, b), are generated. Then, for each pair, Dijkstra's

algorithm is used to find a sub-path between the waypoints in the pair. A complete path that visits all targets is finally created by connecting the sub-paths for each pair following σ.

A square grid-based waypoint network as illustrated in Fig. 3 is constructed for Dijkstra's algorithm. Such a network is easy to get and allows Dijkstra's algorithm to generate almost all possible paths on a region, when the density of nodes in the network is high enough.

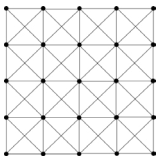

Fig. 3. Network structure for Dijkstra's algorithm

Here, the density of the network is determined by a parameter termed as *precision*. When the precision level is high, we have a dense network with many waypoints, whereas we get a sparse network, when the precision level is low. Fig. 4 illustrates the function of the precision parameter. Obviously, the performance of a path derived by Dijkstra's algorithm is significantly affected by the precision level.

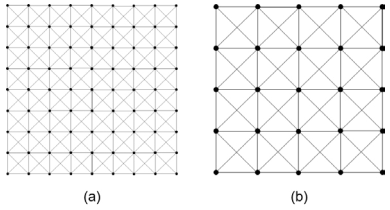

Fig. 4. Network with different precision parameters (a) high precision (b) low precision

A cost to travel from a waypoint to another is calculated considering a distance between them and the treat level of being at the destination waypoint. Following the work of Zhang et al. (2014), the threat level at a position (x, y) by a threat located at (x_t, y_t), can be calculated using an equation as follows:

$$f_t(x, y) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{d^2}{2\sigma}}, \tag{1}$$

where d is the Euclidean distance between (x, y) and (x_t, y_t). The total threat level at point (x, y) is then computed as:

$$F(x, y) = 1 - \prod_{t \in T} [1 - f_t(x, y)], \tag{2}$$

where T is a set of threats in a region. Finally, the travelling cost from node i to node j in a network, c_{ij}, is calculated as follows:

$$c_{ij} = d_{ij} + w \times F_j, \tag{3}$$

where d_{ij} is Euclidean distance between node i and j, w is the parameter for the weight of threats termed as *danger*, and F_j is the total threat level at node j. When w is zero, the problem is a common shortest path problem.

Based on the network structure and the cost function, Dijkstra's algorithm with a priority queue (Goldberg and Tarjan, 1996) is implemented to generate the minimum cost path between two targets.

3.2 A Heuristic Algorithm – Waypoint Generation

The heuristic algorithm termed as *waypoint generation* first draws a straight line between two positions and adjusts the line when the line is overlapping with a threat represented as a circle. The size of the circle is determined considering the threat level. In case of overlapping, in order to adjust the line, and thus avoid the threat, new waypoints generated are inserted into the line.

Let us first explain how to detect whether a line between point A and B is overlapped with a threat. Given a threat represented as a circle with radius r, if the distance between A and the centre of the threat or the distance between B and the centre are less than or equals to r, line $\overline{AB}$ is overlapped with the threat. We also need to check if the minimum distance d between the centre of the threat and $\overline{AB}$ is less than or equals to radius r. A situation, where the last condition is met, is illustrated in Fig. 5.

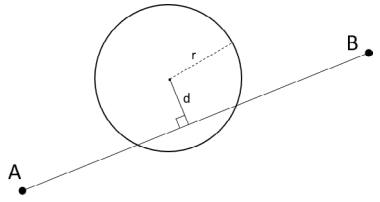

Fig. 5. Overlap between a path and a threat

When all conditions are not satisfied, a UAV can fly directly from A to B. Otherwise, additional waypoints to avoid threats are inserted into the line. The way to adjust a straight line between two targets with the minimum travel distance increase is illustrated in Fig. 6. Based on the concept, the waypoint generation algorithm is designed, as set out in Algorithm 1.

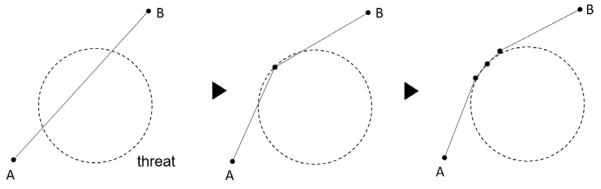

Fig. 6. Straight line update to avoid a threat

3.3 Path Smooting

As the final step, the paths derived by both Dijkstra's algorithm and the waypoint generation algorithm are smoothed using a function described in Algorithm 2.

This function is essential for the path planning algorithms, which can bring a 5% reduction in the travel distance. An example of the path smoothing is presented in Fig. 7.

Algorithm 1 Waypoint Generation

function GETWAYPOINTS(σ)
  Clear $SafePath$, Clear Q
  Add the first point in σ to $SafePath$
  for each pair (o,d) in the σ **do**
    $o^\star \leftarrow o$
    while there are threats overlapping with a line between $o^\star$ and d **do**
      Among all threats overlapping with the line, find threat t with the shorted distance to the line
      Generate waypoint w_p on the boundary of threat t, which leads the minimum distance increase in $SafePath$
      Add w_p to $SafePath$
      $o^\star \leftarrow w_p$
    end while
    Add d to $SafePath$
  end for
  return $SafePath$
end function

Algorithm 2 Path Smoothing ($Path$)

for each triple (o,m,d) in $Path$ **do**
  if m is a waypoint **and** o and d are not a base of a UAV **and** there is no threat **then**
    Remove m from $Path$
  end if
end for
return $Path$

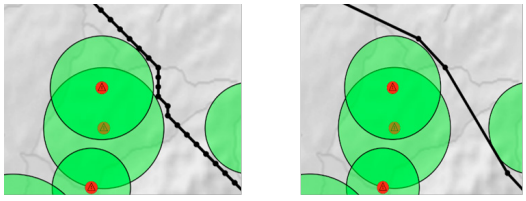

Fig. 7. Example of the path smoothing

4. EXPERIMENTAL RESULTS

4.1 Problem Instance Generation

The performance of the path planning algorithms is investigated under various operational scenarios with respect to the travel time in threats (optimality) and the computation time to derive a path (complexity).

We first define three different configurations for treats and targets as described in Fig. 8.

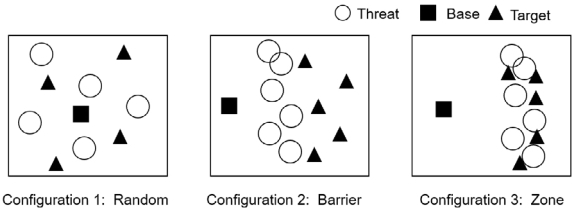

Fig. 8. Configurations of a threat/target distribution

The first configuration, *Random*, represents a random distribution of threats and targets. The second configuration, *Barrier*, describes a situation where targets are protected by threats as a barrier. The last configuration, *Zone*, represents a situation where threats and targets are relatively collocated in a zone.

Given the threat/target configurations, we also vary the numbers of threats and targets and the parameters for Dijkstra's algorithm, i.e. the network density (precision) and a weight of a threat (danger), as listed in Table 1.

Table 1. Parameter setting for the experiments

Parameter	Levels
The number of threats	40 – 20 – 5
The number of targets	40 – 20 – 5
Precision (Dijkstra's)	High – Medium – Low
Danger (Dijkstra's)	High – Low

We generate 10 problem instances for each set of the parameter setting and threat/target configuration. Thus, we conduct 1,620 experiments for Dijkstra's algorithm (54 parameter settings x 10 instances x 3 threat/target configurations) and 270 experiments for the waypoint generation algorithm (9 parameter settings x 10 instances x 3 threat/target configurations) in total. The experiments have been conducted on a PC with Intel Core i7-8700K@4.3GHz equipped with 16 GB of RAM.

4.2 Random Configuration

Fig. 9 summarises the quality of the paths, i.e. the travel time spent in threats, provided by the algorithms.

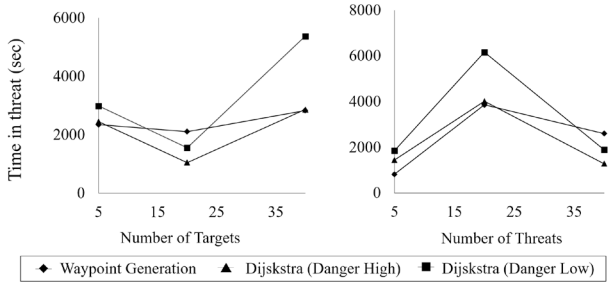

Fig. 9. The travel time in threats (sec) in the random configuration

First, it is observed that Dijkstra's algorithm with a low danger parameter (a low weight on a threat) works poorly. Intuitively, with a low danger parameter, Dijkstra's algorithm tends to generate a path penetrating threats to minimise a total travel distance.

On the other hand, the waypoint generation algorithm shows a comparable performance to Dijkstra's algorithm. The waypoint generation algorithm also outperforms Dijkstra's algorithm in terms of the computation time to get a path as presented in Fig. 10. The waypoint generation algorithm generates a path, on average, within 0.05 sec.

Fig. 10 also shows that the computation time of the algorithms is a function of the number of targets, whereas the computation time of the algorithms are relatively insensitive to the number of threats. Obviously, the computation

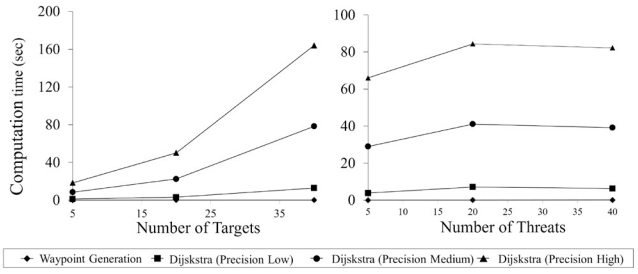

Fig. 10. Computational time (sec) of the path planning algorithms in the random configuration

time of Dijkstra's algorithm increases as the precision of a network for the algorithm increases.

4.3 Barrier Configuration

Fig. 11 shows the performance comparison between the path planning algorithms in terms of the travel time in threats. Unlike the results in the random configuration, the performance of the waypoint generation algorithm becomes poor in the barrier threat configuration. When threats are overlapped, the waypoint generation algorithm often fails to find a path that avoids the overlapped threats because a scheme to avoid a single treat is implemented in the algorithm. Dijkstra's algorithm with a high precision network provides the best results.

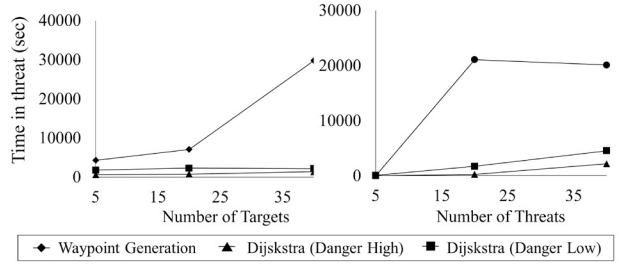

Fig. 11. The travel time in threats (sec) in the barrier configuration

Fig. 12 presents the analysis of the computation time of the path planning algorithms. From the figure, we can see that the computation time of Dijkstra's algorithm is proportional to the number of threats. The waypoint generation algorithm still finds a path within a short time.

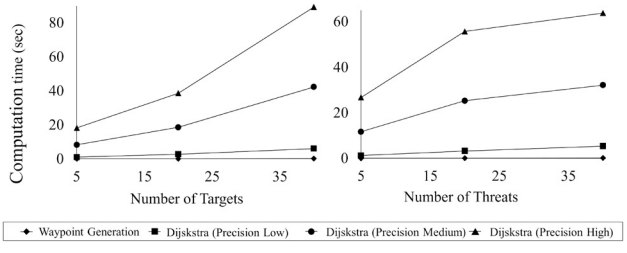

Fig. 12. Computational time (sec) of the path planning algorithms in the barrier configuration

4.4 Zone Configuration

Fig. 13 summarises the threat level of paths derived by the two path planning algorithms. We observe that the performance gap between the algorithms becomes significant as the number of threats increases. When the number of threats is high, the waypoint generation algorithm works poorly.

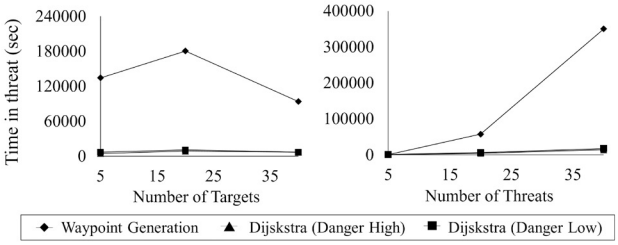

Fig. 13. The travel time in threats (sec) in the zone configuration

Fig. 14 shows the computational time of the algorithms. As shown in the figure, Dijkstra's algorithm with a low precision network can generate a path faster than the waypoint generation algorithm, when the numbers of targets and threats are high. Considering the results presented in Fig. 13, we conclude that Dijkstra's algorithm outperforms the waypoint generation algorithm under the zone threat configuration.

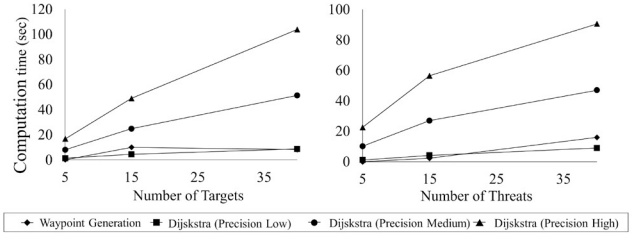

Fig. 14. Computational time (sec) of the path planning algorithms in the barrier configuration

5. SUMMARY

Following the active interest in path planning algorithms in a UAV development, this study analyses two types of path planning algorithms in terms of their ability to avoid threats and computation time to compute a path. We first present Dijkstra's algorithm, which is a common exact approach to solve the path planning problem. However, there are many design issues to apply the algorithm, i.e. how to design a network and the cost function of the algorithm. As observed, the performance of the algorithm is quite depending on the design. We also present the waypoint generation algorithm that finds a path in a simple manner. In the experiments, under the random configuration, the algorithm provides comparable paths to those of Dijkstra's algorithm with respect to the path's threat level, while reducing the time to compute the paths dramatically. Under the barrier and zone configurations, Dijkstra's algorithm outperforms the waypoint generation algorithm.

REFERENCES

Ahmad, Z., Ullah, F., Tran, C., and Lee, S. (2017). Efficient energy flight path planning algorithm using 3-d visibility roadmap for small unmanned aerial vehicle. *International Journal of Aerospace Engineering*, 2017.

Alotaibi, K.A., Rosenberger, J.M., Mattingly, S.P., Punugu, R.K., and Visoldilokpun, S. (2018). Unmanned aerial vehicle routing in the presence of threats. *Computers & Industrial Engineering*, 115, 190–205.

Barrientos, A., Colorado, J., Cerro, J.d., Martinez, A., Rossi, C., Sanz, D., and Valente, J. (2011). Aerial remote sensing in agriculture: A practical approach to area coverage and path planning for fleets of mini aerial robots. *Journal of Field Robotics*, 28(5), 667–689.

Bertuccelli, L., Choi, H.L., Cho, P., and How, J. (2009). Real-time multi-uav task assignment in dynamic and uncertain environments. In *AIAA guidance, navigation, and control conference*, 5776.

Coutinho, W.P., Battarra, M., and Fliege, J. (2018). The unmanned aerial vehicle routing and trajectory optimisation problem, a taxonomic review. *Computers & Industrial Engineering*.

Goldberg, A.V. and Tarjan, R.E. (1996). Expected performance of dijkstra's shortest path algorithm. *NEC Research Institute Report*.

Jiao, Y.S., Wang, X.M., Chen, H., and Li, Y. (2010). Research on the coverage path planning of uavs for polygon areas. In *Industrial Electronics and Applications (ICIEA), 2010 the 5th IEEE Conference on*, 1467–1472. IEEE.

Radmanesh, M., Kumar, M., Guentert, P.H., and Sarim, M. (2018). Overview of path-planning and obstacle avoidance algorithms for uavs: a comparative study. *Unmanned Systems*, 6(02), 95–118.

Yang, L., Qi, J., Xiao, J., and Yong, X. (2014). A literature review of uav 3d path planning. In *Proceeding of the 11th World Congress on Intelligent Control and Automation*, 2376–2381. IEEE.

Zhang, B., Liu, W., Mao, Z., Liu, J., and Shen, L. (2014). Cooperative and geometric learning algorithm (CGLA) for path planning of uavs with limited information. *Automatica*, 50(3), 809–820.

Zhao, Y., Zheng, Z., and Liu, Y. (2018). Survey on computational-intelligence-based uav path planning. *Knowledge-Based Systems*, 158, 54–64.