Proceedings of the 39th Chinese Control Conference
July 27-29, 2020, Shenyang, China

# Air Traffic Network Generation for UAVs at a Low Altitude Based on Digital Maps

Xin He, Quan Quan
School of Automation Science and Electrical Engineering, Beihang University, Beijing 100191, P. R. China
E-mail: hex707@yeah.net, qq_buaa@buaa.edu.cn

**Abstract:** Unmanned Aerial Vehicle (UAV) has a rapid growth in recent years. An efficient air traffic system for UAVs working at a low altitude which is rich of obstacles is desired. However, as far as we know, the work on the generation of such an air traffic network at a low altitude lacks attention. It is expected to develop an algorithm in order to generate a traffic network automatically similar to the complex road traffic network by taking a matrix or image marked with obstacles as input. Considering those situations above, this paper addresses (i) automatic air traffic network generation that is suitable for UAVs working at a low altitude, (ii) several line segments (airlines) fitting for a curve , and (iii) a graph modeling. A simple experiment is also performed to show the effectiveness of the method proposed.

**Key Words:** Air Traffic, Network Generation, Graph Modeling

## 1 Introduction

UAVs are playing a more and more critical role in the world. UAVs can perform package delivery, searching, mapping, and other easy but time-wasting work for humans. Meanwhile, the rapid growth of UAVs brings many challenges. For example, the risk of collisions is increasing. A UAV that is flying can cause unpredictable harm to people or facilities below when accidents happen. Disordered UAVs' routes even increase the risk. In order to decrease the risk, UAVs must be restricted to work in certain permitted areas and avoid collision with other flying objects at the same time. Here, certain permitted areas are the airspace within the air traffic network. This paper aims at generating an air traffic network automatically based on a matrix or image marked with obstacles.

Trajectory planning for UAVs is a classic problem. The design of trajectory for one or several UAVs to avoid collision and danger has been studied extensively [1][2]. However, these algorithms are inappropriate for the generation of a larger-scale air traffic network, because the trajectory generated is only for one task, one point to one point.

Road network design problem mainly focuses on the optimization of the structure of an existing road traffic network or the optimal placement of traffic facilities using traditional methods or heuristic algorithms [3][4][5][6]. A road network for hazardous materials transportation is designed as a bi-level optimization problem based on the existing road network in [7]. The robustness of an existing air traffic network is also studied in [4] when one or more air routes are added. A road network is generated using vehicle tracking data in [8]. Similarly, the structure of a traffic network can also be generated from satellite images based on reliable training areas generation and fuzzy classification in [9]. In summary, there is not much work about air traffic network generation in the presence of rich obstacles at a low altitude. The traffic structure to be established is similar to road traffic network. Meanwhile, road traffic networks are designed manually. It is time-wasting to perform the same process while designing the air traffic network. To this end, inspired by the Voronoi diagram for trajectory planning [10][11], a series of algorithms are developed to accomplish the automation of air traffic network design. Furthermore, the topological structure of the air traffic network is modeled as a graph for further usages.

This work is supported by National Natural Science Foundation (NNSF) of China under Grant 61973015.

## 2 Problem Formulation

The environment at a low altitude is complex with buildings and other UAVs. A geometrical structure of air traffic networks with airways, and intersections is proposed and shown in Fig.1, where UAVs fly straightly along airways and change the azimuths in intersections. A graph $G = (V, E, W)$ is suitable to represent the structure of such air traffic networks whose vertex set $V$ represents intersections, edge set $E$ represents airways between intersections, and weight set $W$ reflects the lengths and widths of airways. Compared to using detailed maps of traffic networks, using graphs is time-saving and memory-saving for trajectory planning and other work. However, the existence of obstacles and other dangers should be taken into account while generating air traffic networks. There are some restrictions mentioned later.

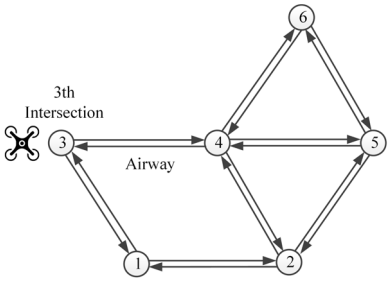



Fig. 1: Air raffic network

### 2.1 Input: Digital Maps

Let $\mathbb{Z}^2$ represent the discrete plane that has two dimensions and is in the form of a square mesh. A binary image $A$, or named as *a digital map*, is a subset of $\mathbb{Z}^2$ and can also be regarded as a set of $M \times N$ pixel points with their coordinates and values to represent the air space. The value

of point $p(i,j)$ belonging to $A$, while $i,j \in \mathbb{N}$, $i \in [1,M]$, and $j \in [1,N]$, is designated logical zero or one by function $f{:}(i,j) \rightarrow \{0,1\}$. Point $p$ is also named as a pixel. The function $f_p$ is equal to 0 only when $p$ represents an area banned for flying. In other cases, $f_p$ is equal to 1, meaning the area point $p$ represents is permitted for UAVs entrance. A binary image is shown in Fig.2 using Boolean values to distinguish restricted areas (e.g., obstacles, crowds of people) and permitted areas. However, a discrete map brings lots of challenges, such as, pixel connectivity, which are mentioned and solved later in this paper.

A digital 2-dimension map can be from SLAM algorithms or satellite images after processing to represent the conditions at a low altitude. It also can be from a 3D digital map through extracting a low-altitude layer of the map or through adding logically several layers at a range of altitudes together. The resolution ratio of a digital map is better to be not smaller than $1m \times 1m$ because of the computational complexity. $1m \times 1m$ is enough to represent the environment and the movement of UAVs, so it is unnecessary to generate a more detailed digital map. However, how the 2D or 3D digital map is generated detailedly is beyond the scope of this paper.

The rectangle shape of $A$ means that parts of discrete plane $\mathbb{Z}^2$ with complex conditions of obstacles can be generated and then a bigger map can be stitched from them. It is convenient for distributed computing and computer-aided design.

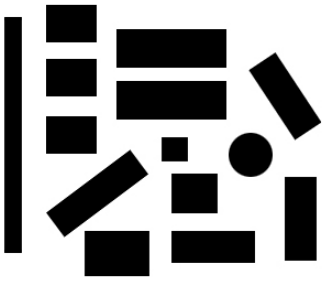


Fig. 2: A digital map $A$ (black for 0 and white for 1)

### 2.2 Objective: Topological Structure

An abstract undirected and weighted graph $G = (V,E,W)$ is ought to be modeled to represent an air traffic network. Every trajectory on the network can be represented by a series of vertices and edges. Such a graph has nonempty vertices set $V$ that includes branch points, endpoints, and other intersections at an air traffic network with a size of n and can be represented by

$$V = \{v_1, v_2, v_3, \ldots, v_n\},$$

and $\forall i, v_i \in \mathbb{Z}^2$, edges set $E$ which is a set of unordered tuples meaning the connectivity between two vertices whose size $m$ is not larger than $\frac{n(n-1)}{2}$ which can be represented by

$$E = \{(v_1,v_2),(v_1,v_3),(v_1,v_4),\ldots\}$$

if there is an edge between the two vertices of every tuples, and weight set $W$ has a same size of $E$ and represents the cost of edges which can be represented by

$$W = \{w_{12}, w_{13}, w_{14}, \ldots\}$$

when $w_{ij}$ means the weight of edge $(v_i, v_j)$.

In this paper, an unweighted graph $G = (V,E)$ is modeled. The weight of every edge which is not designated in this paper can be valued depending on the length and width of the corresponding airway.

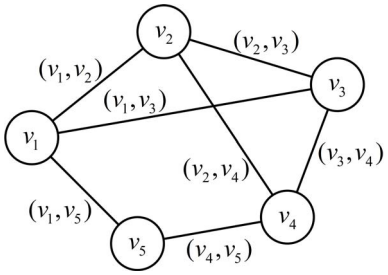



Fig. 3: A graph

### 2.3 Constraint: Air Traffic Network

This section is to describe the constraints while constructing an air traffic network considering obstacles and other dangers. The traffic network must be discretized into the plane $\mathbb{Z}^2$ for computing and processing. An airway is designed as shown in Fig.4. The airway $W_{[v_i,v_j]}$ is a set of points in the rectangle with $v_i \in V$, $v_j \in V$, $(v_i,v_j) \in E$, and width $r_D = 2r_{AW} + r_{IS} > 0$. The line segment $[v_i,v_j]$ is the centerline of the airway. The intersection $I_{v_j}$ is a set of points in the circle with center $v_j \in V$, and radius $r_{IT,j} > 0$. The digital map is a 2D map at a certain height. Therefore, the heights of airways and intersections are unnecessary for air-traffic-network generation. While $A_1 = \{p | p \in A, f(p) = 1\}$, $A_0 = \{p | p \in A, f(p) = 0\}$, and $N = (\bigcup_{i=1}^{m} W_{[v_i,v_j]}) \cup (\bigcup_{j=1}^{n} I_{v_j})$, then we need

$$N \cap A_0 = \emptyset$$

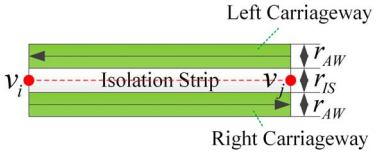



Fig. 4: Airway structure

## 3 Approach

### 3.1 Basic Idea

Major procedures are shown below in Fig.5 as a flowchart. Skeletonization is taken to find the centerline of an air traffic network. Line-segment fitting is to find the nodes where straight lines intersect to optimize the skeleton. Airports are also added into the air traffic network as intersections. Graph $G$ is modeled whose set $V$ consists of branch points and endpoints of the skeleton, endpoints of line segments for fitting, and airport points. And set $E$ consists of all the connectivities between points in set $V$ in the form of adjacency matrix in the algorithm.

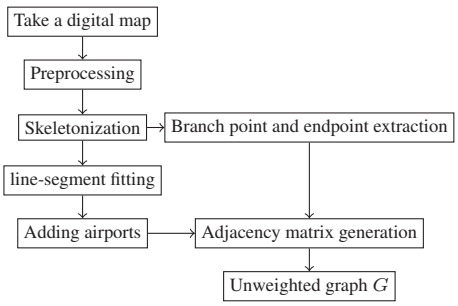



Fig. 5: A simple flowchart of our work

### 3.2 Skeletonization

Danger does not have a hard cut-off on the borders of restrict areas. Therefore, UAVs should stay far enough away from restricted areas. Besides, the airways to be built should be wide enough to contain as many airlines as possible. As for those questions, a centerline should be found which is in the same distance from two nearest obstacles to guide the construction of airways.

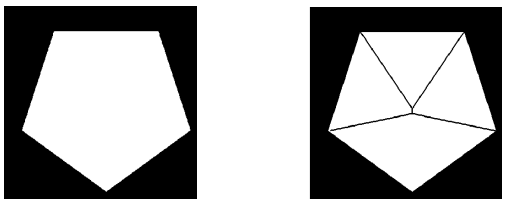
The method of generating a Voronoi Diagram to find a trajectory in [10] is inspiring. However, they only take dangerous abstract points into account. It is more reasonable to generate a Voronoi Diagram of all border points. The skeleton extraction of the digital map is a similar method and takes all points on borders into account. The airways based on it are safe for flying.

Skeletonization means repeating image thinning until the image is unchanged. Image thinning can be explained simply by two steps.

**Step 1**. Get the pixel set $A_1$ from image $A$. Get the templates for image thinning.

**Step 2**. Match the points in $A_1$ with the templates. If a point matches successfully, designate a logical 0 to it, then update image $A$.

Although skeleton extracting is not absolutely symmetrical, it performs well in practice.



(a) A raw image (b) After skeletonization

Fig. 6: A regular pentagon and its skeleton. As a result of scanning order of the image, its skeleton is not in a 5-point star shape

Airways have their minimum width $r_D$, and intersections have their minimum radius $r_{IT,min}$. And for other safety concerns, a higher prescribed minimum of road widths is reasonable for permitted areas to construct an airway. A corrosion method in morphology is taken several times (at least $min\{r_D/2, r_{IT,min}/2\}$ times) to filter out the airways generated from narrow areas whose width is smaller than or close to corrosion times and the intersections whose radius is smaller than or close to half of the corrosion times. Corrosion is anothor method about image thinning but uses different templates. The skeleton of Fig.2 is shown in Fig.7 after corrosion.

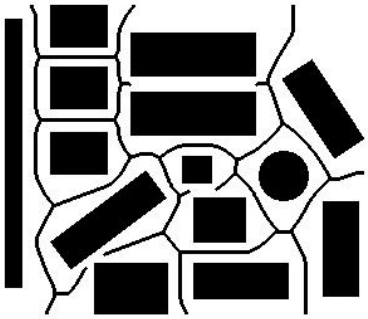


Fig. 7: After corrosion, the skeleton becomes more reasonable

### 3.3 Branch point and Endpoint Extraction

This section is mainly about the search of some important points in a skeleton, that is, branch points and endpoints, which can represent parts of the topology of an air traffic network. Endpoints mean each of them connects to only one airway representing an end, and each of the branch points connects at least three airways representing a junction in air traffic networks. A subset $B$ consisting of branch points and endpoints of point set $V$ should be found. One point belongs to set $B$ once it satisfies one of the following conditions.

(1) It has only one neighbor in its $3 \times 3$ neighborhood. It is an end point.

(2) It has at least three neighbors in its $3 \times 3$ neighborhood. It is a branch point.

A $3 \times 3$ neighborhood of point $a$ is shown in Fig.8. In other words, when $a$ is equal to 1, condition (1) means that $a_1 + a_2 + a_3 + a_4 + a_5 + a_6 + a_7 + a_8 = 1$, and condition (2) means that $a_1 + a_2 + a_3 + a_4 + a_5 + a_6 + a_7 + a_8 > 2$.

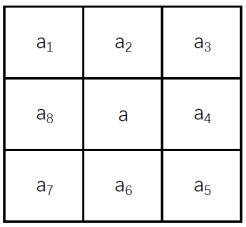



Fig. 8: $3 \times 3$ neighbors of point $a$

With branch points and endpoints extracted, their adjacencies are then analyzed to form an unweighted graph. The two types of points extracted are shown as Fig.9 when Fig.2 is the input.

### 3.4 Adjacency Matrix Generation

This section is about the next step of extracting a topological structure out of a raw air traffic network. This section is particularly aimed at extracting the topology of the skeleton. And then the adjacency relationships are changed with line-segment fitting and airport adding accordingly when new intersections are added. First, the concept of adjacency must be figured out. Let $x = (i, j)$ be in $A$. The four parts $(i \pm 1, j)$ and $(i, j \pm 1)$ are called the 4-neighbors of $x$, and are

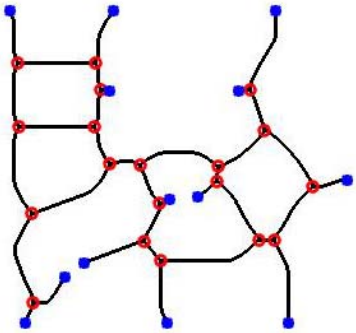


Fig. 9: Branch points and endpoints (shown as red circles and blue squares)

said to be 4-adjacent to $x$. Similarly, $(i \pm 1, j \pm 1)$ are called 8-neighbors of $x$ and are 8-adjacent to $x$. 8-neighborhood and 4-neighborhood of point $a$ are shown in Fig.10

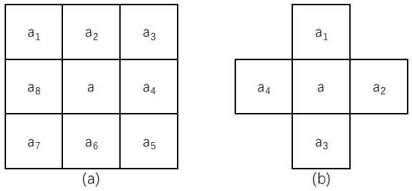



Fig. 10: 8-neighborhood and 4-neighborhood of point $a$

In place of extracting an edge set $E$, the adjacency of every intersection is figured out to make a graph complete, so different trajectories can be represented in a series of intersections and airways. Meanwhile, a simple method of extracting different airways is proposed in section 3.5 but is not chosen for graph modeling in this section.

The searching process starts from one intersection, search along the skeleton, and stops when finding all the other intersections connected to the start intersection. Following is a little piece of pseudocode explaining the procedure. Let the start point's coordinates be $(x_0, y_0)$ (and the procedure is shown in Algorithm 1).

The procedure is also illustrated in Fig.11. They are simple explanations not mentioning the fact that 4-neighborhood of one point should take a priority to be searched over other 8-neighbors. The searching process ends when find a neighbor point in the 4-neighborhood.

### 3.5 Fit Skeleton with Several Line Segments

The skeleton is hard to fly along because of its irregular shape. If intersections are set up, and between adjacent intersections, there is a straight airline to fly along, the air traffic network is more reasonable. Therefore, the curves in the skeleton should be replaced by line segments and their endpoints.

Divide-And-Conquer iterative boundary detection is taken to fit the skeleton with several line segments. The endpoints of line segments for fitness are parts of set $V$ and the adjacency matrix is changed accordingly. The fitness algorithm mainly contains two parts, edges separation and line-segment fitness.

As for edges separation, zeros are assigned to branch

**Algorithm 1** Find Neighbors of Point $p$

**Input:** the coordinates $(x_0, y_0)$ of point $p$, an binary image of the skeleton $S$
**Output:** a list of neighbors $NList$

$S \leftarrow S/2$
$OPENList \leftarrow \emptyset$
$OPENList \leftarrow (x_0, y_0)$
$NList \leftarrow \emptyset$
**while** $OPENList \neq \emptyset$ **do**
  $(x, y) \leftarrow OPENList(1)$
  **for** $i = -1; i < 2; i++$ **do**
    **for** $j = -1; j < 2; j++$ **do**
      **if** $S(x+i, y+j) = 0.5$ **then**
        Add $(x+i, y+j)$ to the end of $OPENList$
        $S(x+i, y+j) \leftarrow S(x, y)$
      **else if** $S(x+i, y+j) \neq 0$ **then**
        **if** $S(x+i, y+j) \neq S(x, y)$ **then**
          Add $S(x+i, y+j)$ to $NList$
        **end if**
      **end if**
    **end for**
  **end for**
**end while**

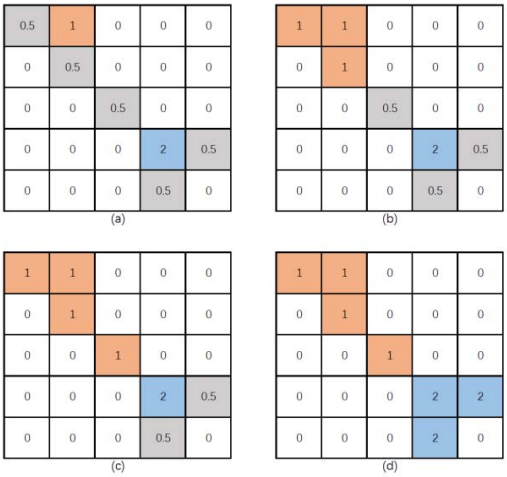



Fig. 11: A simple illustration when searching neighbors of an intersection

points' $3 \times 3$ neighborhoods. Therefore, the connectivity of different edges is broken. Then, a connected-component labeling method can be taken to separate and mark different airyways (Fig.12).

The points of different edges in a skeleton can also be regarded as different connected regions once they are separated. Thus, this algorithm is utilized to mark points of different edges and count the number of edges.

Then, for line-segment fitness, straight lines are utilized to approximate the curves of the skeleton. For example, in Fig.13, if $h$, which is the furthest distance from points on curve $\widehat{AB}$ to straight line $\overline{AB}$, is larger than the distance tolerance $H$, using straight lines $\overline{AC}$ and $\overline{CB}$ to approximate $\widehat{AB}$ is better. And this is a recursive procedure.

In our program, edges separation and connected-component labeling mentioned above are also performed to implement line-segment fitting.

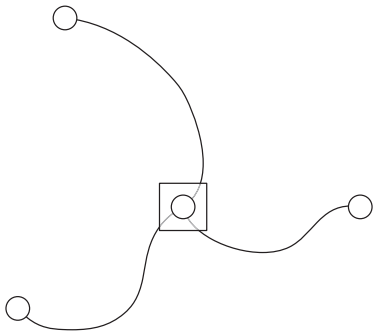


Fig. 12: Break the connectivity of three edges at a branch point

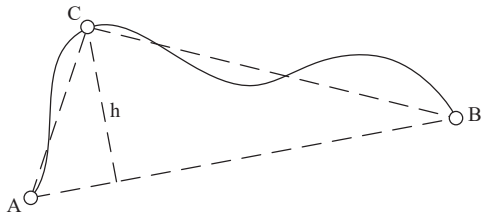



Fig. 13: An illustration of lines approximating

### 3.6 Airports

As a kind of essential facilities in air traffic networks, airports should be added into the air traffic network and set $V$. A distance definition $D = max\{(x_1 - x_2), (y_1 - y_2)\}$ is used to measure the distance between point $(x_1, y_1)$ and point $(x_2, y_2)$. It is also called as D4 distance because 4-neighbors are regarded as a distance of 1 away from the center point. Normally, the airports are close to the centerline, and finding the nearest airway by measuring D4 distances is more time-saving. A simple algorithm is implemented to add those facilities.

**Step 1**. Search for the nearest airway from one of the airports with priority in breadth using 4-neighborhood. Meanwhile, the searching space is diamond-shaped.

**Step 2**. After finding the nearest airway, connect the airport point with the corresponding two endpoints on that airway, including the points generated using for skeleton fitness. The airport point is a element of set $V$ and change adjacency matrix accordingly.

This algorithm is repeated until all the airport points are connected to the network. An airport point is added to the traffic network in Fig.15.

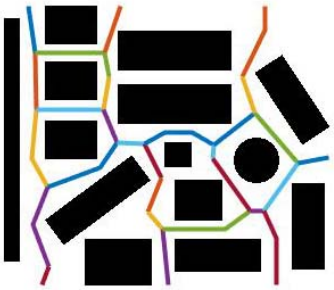


Fig. 14: Skeleton after fitness of Fig.2

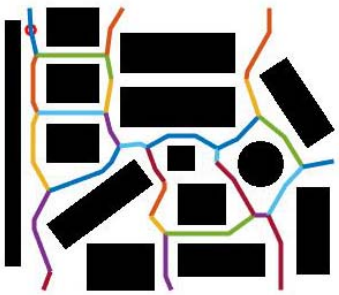


Fig. 15: An airport is added to the network (shown as a point with a red circle)

## 4 Simulation and Experiments

Our implementation is executed in MATLAB R2016a on a PC running Win 10, with an AMD Athlon X4 760k 3.8 GHz CPU and 8 GB RAM. The AMD Athlon X4 760k used is not a powerful CPU compared to the CPUs used nowadays. The speed of the implementation can be faster in newer PCs.

### 4.1 Parameter Analysis

Fig.1 is taken as input. There are two parameters that can be changed accordingly. Parameter $min\{r_D/2, r_{IT,min}/2\}$ mainly affects corrosion times and the deletion of skeleton generated from narrow areas. Parameter $H$ as a minimum distance by line-segment fitting affects mainly the number of line segments for approximating. Corrosion times and $H$ are changed as experiments to find their influences specifically. Generally, when $H$ in pixels, $H$ is smaller than the corrosion times to make sure $N \cap A_0 = \emptyset$.

The resolution ratio of the digital map is $400 \times 400$. Several experiments are carried out. The result is recorded in Table.1 and Fig.16. As for computing time, skeletonization itself takes 1.49 seconds at least. It takes over 80% of the computing time to generate a skeleton.

Table 1: Experiment Parameters and Computing Time

| Experiment Number | (1) | (2) | (3) | (4) |
|---|---|---|---|---|
| Corrosion Times | 6 | 3 | 6 | 8 |
| $H$(pixels) | 2 | 2 | 5 | 5 |
| Computing Time(s) | 1.87 | 1.83 | 1.83 | 1.88 |

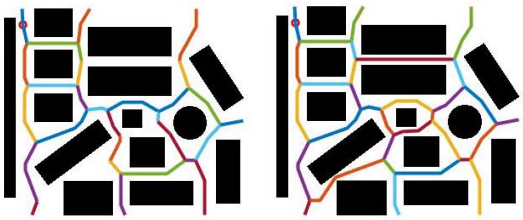
### 4.2 A Real-map Experiment

A real map is taken as a simple experiment. Fig.17 is a part of the 3D map from AMAP. This map is part of our campus. After image-processing, a layer of the map at an altitude of 10 meters is taken. And only obstacles are taken into account this time. The digital map as input is shown in Fig.18. With an airport near the center point added, the air traffic network generated is shown in Fig.19. The air traffic network has a similar shape as the road traffic network shown as white lines in Fig.17 but does not need to avoid the greenspace lying on the center of the map.

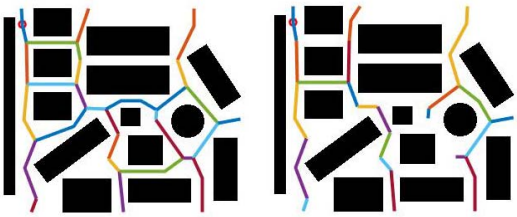
The resolution ratio is $286 \times 515$, and the computing time is 5.21s, with skeletonization taking 4.90s (94%).

### 4.3 Discussion

Through experiments, there is still room for improvement. The computing time of skeletonization is too long and in-



(a) Experiment (1) (b) Experiment (2)



(c) Experiment (3) (d) Experiment (4)

Fig. 16: The networks generated with different parameters

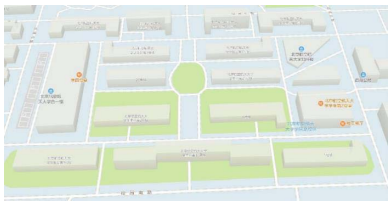


Fig. 17: A 3D map

creases with the enlargement of the permitted areas. The structure of the air traffic network generated needs to be optimized. For instance, intersections cancellation and intersections reposition to make roads more straight and merge roads into one can make the network more formal.

## 5 Conclusion

To decrease the danger while UAVs are working, a structure for UAVs' traffic is desired. In saving of labor, an approach for automatic air-traffic-network generation is proposed in this paper. The air traffic network generated is for UAVs' traffic flow from all directions and in all directions at a low altitude, and is designed for a large scale. Restricted areas and permitted areas are taken into account and generate a traffic network based on a binary image/digital map with restricted areas marked. The network generated is mostly safety-concerned because it was built based on the skeleton and airways are designed to be wide enough firstly. The skeletons of airways unsuitable (not wide enough) are deleted because of a corrosion method. Some optimal methods are taken to make the network more reasonable for flying, for example, line-segment fitting and airports adding. On the air traffic network, airways, and intersections can be constructed easily to provide an efficient air traffic structure. A graph $G$ is also modeled to represent the air traffic network for further usages.

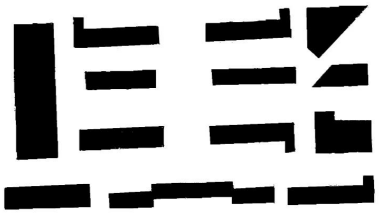


Fig. 18: An image (digital map) at an altitude of 10m

Fig. 19: The air traffic network generated

There are still some problems to be solved. Optimization of the network structure is essential. The coordinates of the intersections can be more reasonable, and the algorithm of skeletonization shoudle be optimized to be faster.

## References


[1] Sunan Huang, Rodney Teo, and Kok Kiong Tan. Collision avoidance of multi unmanned aerial vehicles: A review. *Annual Reviews in Control*, 48:147–164, 2019.

[2] Aaron Mcfadyen and Luis Mejias. A survey of autonomous vision-based see and avoid for unmanned aircraft systems. *Progress in Aerospace Sciences*, 80:1–17, 2016.

[3] Satish V Ukkusuri, Tom V Mathew, and S Travis Waller. Robust transportation network design under demand uncertainty. *Computer-Aided Civil and Infrastructure Engineering*, 22(1):6–18, 2007.

[4] Changpeng Yang, Jianfeng Mao, and Peng Wei. Air traffic network optimization via Laplacian energy maximization. *Aerospace Science and Technology*, 49:26–33, 2016.

[5] Reza Zanjirani Farahani, Elnaz Miandoabchi, Wai Yuen Szeto, and Hannaneh Rashidi. A review of urban transportation network design problems. *European Journal of Operational Research*, 229(2):281–302, 2013.

[6] Partha Chakroborty and Tathagat Wivedi. Optimal route network design for transit systems using genetic algorithms. *Engineering optimization*, 34(1):83–100, 2002.

[7] Bahar Y Kara and Vedat Verter. Designing a road network for hazardous materials transportation. *Transportation Science*, 38(2):188–196, 2004.

[8] Sophia Karagiorgou and Dieter Pfoser. On vehicle tracking data-based road network generation. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, pages 89–98. ACM, 2012.

[9] Uwe Bacher and Helmut Mayer. Automatic road extraction from multispectral high resolution satellite images. *Proceedings of CMRT05*, 36:3, 2005.

[10] Kevin Judd and Timothy McLain. Spline based path planning for unmanned air vehicles. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, page 4238, 2001.

[11] Randal W. Beard and Timothy W. Mclain. *Small Unmanned Aircraft:Theory and Practice*. Princeton University Press, 2012.

[12] Quan Quan, Gang Li, Yiqin Bai, Rao Fu, Mengxin Li, Chenxu Ke, and Kaiyuan Cai. Low altitude UAV traffic management:an introductory overview and proposal. *ACTA AERONAUTICAET ASTRONAUTICA SINICA*, 41(1):23238, 2020.

[13] Quan Quan. *Introduction to multicopter design and control*. Springer Singapore, 2017.