Hindawi
International Journal of Aerospace Engineering
Volume 2017, Article ID 2849745, 13 pages
https://doi.org/10.1155/2017/2849745

Research Article

Efficient Energy Flight Path Planning Algorithm Using 3-D Visibility Roadmap for Small Unmanned Aerial Vehicle

Zahoor Ahmad,[1] Farman Ullah,[1,2] Cong Tran,[1] and Sungchang Lee[1]

[1]*School of Electronics and Information Engineering, Korea Aerospace University, Deogyang-gu, Goyang-si, Gyeonggi-do 412-791, Republic of Korea*
[2]*Department of Electrical Engineering, COMSATS Institute of Information Technology, Attock, Pakistan*

Correspondence should be addressed to Sungchang Lee; sclee@kau.ac.kr

Received 24 January 2017; Revised 28 September 2017; Accepted 9 October 2017; Published 4 December 2017

Academic Editor: Hikmat Asadov

This paper presents the flight path planning algorithm in a 3-dimensional environment with fixed obstacles for small unmanned aerial vehicles (SUAVs). The emergence of SUAVs for commercial uses with low-altitude flight necessitates efficient flight path planning concerning economical energy consumption. We propose the visibility roadmap based on the visibility graph approach to deal with this uprising problem. The objective is to approximate the collision-free and energy-efficient flight path of SUAVs for flight missions in a considerable time complexity. Stepwise, we describe the construction of the proposed pathfinding algorithm in a convex static obstacle environment. The theoretical analysis and simulation results prove the effectiveness of our method.

1. Introduction

In recent years, the small unmanned aerial vehicles (SUAVs) are gaining much attention due to lightweight, inexpensive, and low-altitude flights. Meanwhile, the SUAV encounters an increasing number of obstacles at low altitude. It increases the probability of collision during missions. The autonomous obstacle avoidance technology is one of the main development trends in this domain and energy consumption reduction is a significant concern for SUAVs. Due to the limitation of carried payloads on the SUAVs, the endurance of flight operation will be primarily dependent on control management, including optimized path planning.

The obstacle-avoidance pathfinding problem has been a major research field for autonomous robotics in 2-D space and aircraft and UAV path planning in 3-D environment. Probabilistic roadmap (PRM) [1] and rapidly exploring random tree (RRT) [2] arbitrarily generates the waypoints and flight paths using randomness and stochastic methods. The selected path is optimized and trained by machine learning techniques such as ant colony algorithm (ACO) [3], evolutionary algorithm (EA) [4], or particle swarm optimization (PSO) [5]. The optimization step depends on running duration of the selected machine learning algorithm. Slow but robust, they are dedicated for offline path planning in complex environments.

The appropriate approaches of finding a path in high resolution and large-scale data point maps are uniform grid [6], navigation mesh [7], Voronoi diagram [8], visibility graph [9], and Silhouette method [10]. All these discrete methods create a graph and the graph traversal locates the optimized path by pathfinding algorithms such as Dijkstra [11], A* [12], or newly developed algorithms which are variations of A* such as Jump Point Search [13]. The paths generated by these methods have a huge number of heading changes except the visibility graph. The issue is addressed by algorithms such as Field D* algorithm [14] using interpolation computation or Theta* [15] which combines the visibility checking with uniform grid method.

Artificial Potential Field (APF) [16] appears to be a promising applicable obstacle avoidance method for SUAV as it requires little computation time [17, 18]. However, this approach is typically efficient for the 2-D environment and limited shape of obstacles such as spheres, cylinders, or cones, and the process easily converges into local

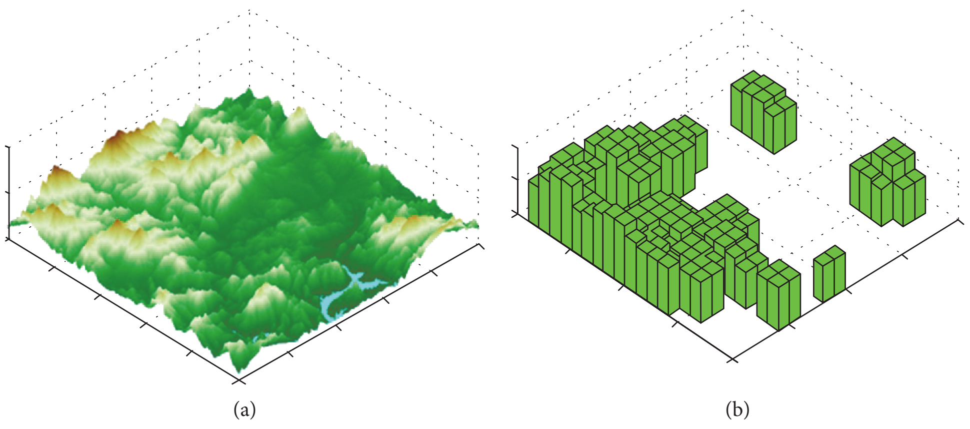

Figure 1: Modeling of SUAV operating environment: (a) sample of digital moving map in 1 kilometer square; (b) generated convex modeling of obstacles based on a given map.

minima. Moreover, the APF method does not create an optimized path.

A lot of research has been done in the area of pathfinding and so many algorithms had been proposed. However, the SUAV energy-efficient path selection is a daunting research so far. Nachmani [19] presented a model which followed Carson's speed theory [20] to calculate the energy cost between uniform grids for fixed wing UAVs. Likewise, Phung and Morin [21] provides a calculation for vertical take-off and landing (VTOL) type UAVs. These energy conservation models considered the energy consumed in the form of potential energy, kinetic energy, and heading change energy. However, these methods did not consider the existence of obstacles.

In this paper, we focus on energy-efficient pathfinding that takes into account 3-D paths having a low number of obstacles and low number of heading changes. We exploit flight path planning and obstacle avoidance techniques in the topic of energy consumption optimization. Clearly, a Euclidian-shorter flight path which takes place above obstacles may incur more energy consumption in comparison with a longer but stable straight flight path. In this paper, we suggest a novel approach named 3-D visibility roadmap, an extension of visibility graph in 3-dimension to plan an optimized flight path for SUAVs. It will provide the flight energy consumption reduction. We also describe the method to reduce the time of flight path selection to be adaptable to users' interaction experience.

The paper is organized as follows: Section 2 describes the problem statement and background. Section 3 emphasizes on our contribution to this problem and explains in details about the functioning method of our algorithm. Our simulation and evaluation are presented in Section 4. This section also discusses the advantages and disadvantages of this approach. Section 5 states our conclusions and future work.

2. Pathfinding in 3-Dimensional Digital Map

2.1. Modeling of the Environment

2.1.1. Digital Map and Convex Obstacle Modeling. First, we model the obstacle in a 3-D map used for the mission planning of SUAVs. The environment is set as a wind vector field installed in the map.

We model the map elevation data with a set of convex obstacles by calculating the convex hull. Figure 1(a) shows the modeling of the 3-D map using the DTED standard [22], and Figure 1(b) shows the obstacles in the flight path planning that should be avoided. Considering the object modeling and pathfinding in the 3D environment, the paper research problem statement can be summarized as finding an efficient energy consumption flight path in 3-D configuration space of static convex obstacles and wind field under maneuverability of SUAV.

2.2. Visibility Graph. The visibility graph is one of the pathfinding approaches for autonomous robots in a known environment. The strongest point of the approach is to calculate the true shortest path with the least heading changes. We can define heading changes as the total number of turns in the shortest path. Finding an efficient flight path at the cost of time complexity of this algorithm is always the most concerned issue of researchers. In the subsections below, we briefly introduce the studies related to visibility graph.

2.2.1. 2-Dimensional Space. The visibility graph approaches are centered on the geometric aspect of the shortest path problem. Lee [23] proposed the pathfinding algorithm based on the binary search tree and had the running time up to $O(n^2 \log n)$. Consecutively, several other algorithms with similar method could run in $O(n^2)$ time [24–26]. Mitchell [27] showed a method to compute the shortest path in $O(n^{5/3+\varepsilon})$ time which was improved later to $O(n^{3/2+\varepsilon})$. Hershberger and Suri [28] developed an algorithm for searching the path along the direct line connecting the start node and the destination node and had an optimal time complexity of $O(n \log n)$. In a particular case, when the environment consists of only convex polygons, to the best of our knowledge, the least computing effort of this method was derived by Hans Rohnert in $O(n + f^2 \log n)$ time with f as the number of supporting segment in the visibility graph [29].

2.2.2. 3-Dimensional Space. Canny [30] proved that finding the shortest path in 3-D is an NP-hard problem. Determining the selection vertices for the graph plays the most important role in solving the problem. Jiang et al. [31] illustrated the shortest Euclidian path based on projections of obstacles on the reference plane in $O(n^3 v^k)$ time complexity. However, these projection approaches are not appropriate for the energy consumption optimization problem due to the lack of 3-D environment characteristic. Reif and Storer [32] proposed the algorithm for finding the shortest path in the 3-D environment in a single exponential upper-bound time complexity. The shortest path is found in polynomial time by selectively assigning nodes on obstacles and searching in the visibility graph connected by these nodes [33–37]. As far as we know, there has not been any progress in discovering lightweight methods for visibility graph in 3-D environment. Furthermore, there has not been any applications of visibility graph to the energy estimation in the path planning problem.

3. The Visibility Graph Approach for Energy Consumption Optimization

The nomenclature used in this paper is listed in Nomenclature.

The optimality and flight stability of visibility graph bring benefits for the energy consumption control. Therefore, we extend this method to apply for our specific problem of path-finding with convex obstacles in the 3-D digital moving map. However, to create an adaptable approach, we have to reduce the complexity of the method to an acceptable limitation. For that purpose, we propose a two-stage procedure: preprocessing phase and searching phase. Firstly, in the preprocessing phase, we create a full visibility graph called Visibility Roadmap which stores every possible connection of traversable waypoints having the direct line of sight with others in a node set N and an edge set P. We reduce the number of processing nodes by sampling traversable waypoints with an altitude interval d_{cut}. The modeling data of the environment and the roadmap is created offline in base stations. This data will only be processed one time. Secondly, the searching phase begins when the pathfinding module receives starting coordinate and destination coordinate. By connecting these coordinates to the roadmap, we obtain the optimized flight path using A* algorithm as soon as the mission starts.

3.1. Roadmap Creation Algorithm. In this subsection, we present the description of the roadmap and its generation algorithm. A given digital map M is represented by elevation values in a 3-D scalar matrix. Roadmap Γ is a set of waypoints and connection flight paths between waypoints. Mathematically, it is a directed double edge graph in which all visible vertices are interconnected: $\Gamma = [N, P]$.

To construct the roadmap, the first step involves sampling of the 3-D for possible waypoints. The map is divided into k-layers by k horizontal planes with equal distance d_{cut} from $H_{\min}$ to $H_{\max}$. We find the waypoints located within the boundary of the obstacles by calculating the intersection points of cut plane and edges of the obstacle. If the cut plane

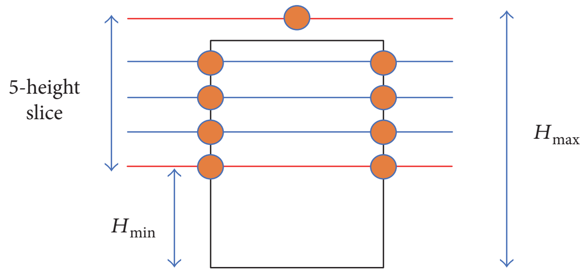

FIGURE 2: Roadmap creation for a single obstacle: side view of k-division on a single obstacle.

is higher than the obstacle, then we determine a single waypoint that locates above the obstacle which has the same altitude as the cut plane. After that, we enlarge the obstacle by a safe distance, which means pushing the intersections by a D_{safe} out of the obstacle. The new enlarged vertices' coordinate shall be stored in N set. Figure 2 presents the visualization of processing a single obstacle during the algorithm.

After discovering the node set N, we create the edge set P with visibility check. The time complexity of the algorithm depends heavily on the line of sight checking function. Given the convex characteristic of obstacles, we could derive a method to compute the line of sight in linear time. Figure 3 shows the steps of extending the obstacle and line of sight checking function.

There are two different cases in the checking procedure; for example, the pair of nodes is within the same obstacle or located on different obstacles. Regarding the convex characteristic, the origin and destination nodes possess the direct line of sight if they are in the same facet. Otherwise, they do not see each other. Come to the case of nodes originated from different obstacles, we process by exploiting convex intersection algorithm below [38] to check the visibility.

The line of sight checking algorithm considers an obstacle as a convex polyhedron Ω consisting a set of convex polygon faces F_i where i is a polygon face. The face is selected accordingly to the order in the doubly connected edge list (DCEL) of the obstacle as described in Section 2. The DCEL data structure enables us to easily compute the normal vector of F_i called $\mathbf{n}_i$ that has the outward direction to the obstacle. That means every point P_i which is on the side of $\mathbf{n}_i$ is exterior to the obstacle. Let us call V_i as a point in the plane of face F_i; we choose the vertex F_i for simplification. Denote P_0 and P_1 as the two waypoints in space. A line segment $S = P_0P_1$ represented by a parametric equation $P(t)$ is the line of sight between these two waypoints. If this line segment intersects with a convex polyhedron, the line of sight is blocked by the current-progressed obstacle, as depicted in Figure 4.

According to basic geometry, the intersection occurs when $(P(t) - V_i) \times \mathbf{n}_i = 0$. Therefore, the value t_i is computed as $(V_i - P_0 \times \mathbf{n}_i)/((P_1 - P_0) \times \mathbf{n}_i)$. Because the normal vector $\mathbf{n}_i$ points to the outward direction of face plane, we can specify the value of t_i when the line segment S is entering or leaving. For example, $(P_1 - P_0) \times \mathbf{n}_i < 0$ when t_i is entering and $(P_0 - P_0) \times \mathbf{n}_i > 0$ when t_i is leaving.

We need to compute $t_{\text{E}} = \max$ (0, t_i that are entering) and $t_{\text{L}} = \min$ (1, t_i that are leaving). If the order of t_{E} and t_{L} is

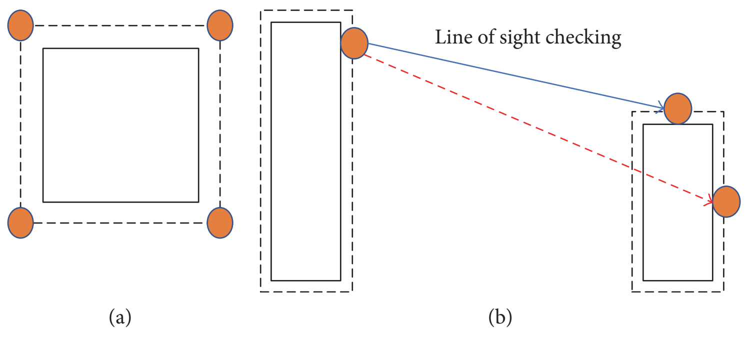

Figure 3: Visualization of steps in roadmap-creation algorithm: (a) extension of distance; (b) line of sight checking between nodes.

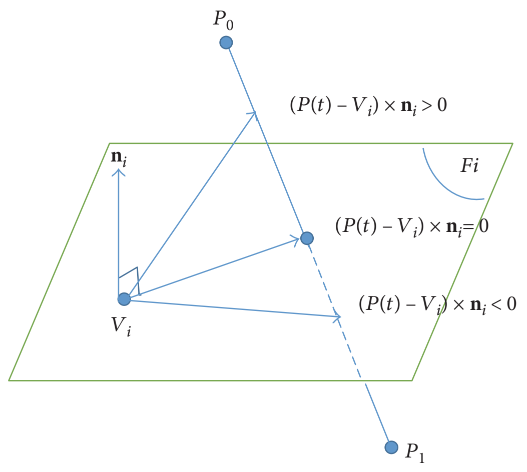
P_0
$(P(t) - V_i) \times \mathbf{n}_i > 0$
$\mathbf{n}_i$
Fi
$(P(t) - V_i) \times \mathbf{n}_i = 0$
$(P(t) - V_i) \times \mathbf{n}_i < 0$
V_i
P_1

Figure 4: Intersection of a line segment to a convex facet in 3-D.

$0 \le t_E \le t_L \le 1$, then the line segment S intersects with convex polyhedron Ω. It also means that the visibility check result is false. While checking the line of sight between each pair of elements in the set of nodes, if they have the direct line of sight, the connection (origin node and destination node indexes) is stored as an edge.

We summarize the algorithm of the roadmap generation as a pseudocode below.

With linear time complexity of the line of sight checking algorithm, the time complexity of the roadmap creation algorithm is $O((nkf)^2)$ where n is the number of obstacles, k is the number of layers, and f is the number of facets.

3.2. Energy Consumption Model. In this paper, we consider a 25 kg fixed wing SUAV. Since our prototype SUAV structure has similarities with [19], we extended this concept of the energy consumption to select the efficient path using the roadmap.

The total energy for traveling between a specified start node and the destination node would be the sum of the total energy spent in traveled edges. Hence, in order to calculate the energy of a path which consists of multiple straight flight paths, we need to compute energy consumption $\Delta E_{i,j}$ going through each straight line with distance Δd from node i to node j (Figure 5). The difference in the total energy consumption should be the sum of differences in potential energy and difference in kinetic energy plus the energy used to turn the SUAV between arcs.

$$\Delta E_{i,j} = \Delta E_{\mathrm{p}} + \Delta E_{\mathrm{k}} + E_{\mathrm{turn}}. \tag{1}$$

Assume that there is no energy gain for SUAV during decreasing altitude, the difference of potential energy is given as

$$\Delta E_{\mathrm{p}} = \max(W\Delta h, 0) = \max\left(mg\left(h_i - h_j\right), 0\right). \tag{2}$$

To calculate the kinetic energy, we focus on optimizing the velocity which depends on the drag-to-lift ratio of the SUAV. In his work, Carson [20] represented the drag-to-lift ratio as

$$\frac{D}{L} = AV^2 + \frac{B}{V^2}, \tag{3}$$

where V is the flight speed and A and B are variables calculated by air density; SUAV's parameters are given as

$$A = \frac{\rho f}{2W}, \qquad B = \frac{2W}{\rho b^2 \pi e}. \tag{4}$$

Most of the parameters are dependent on the structure of SUAV; here, $W = mg$ is the aircraft's weight, b is the wing span, f is the parasite area of the aircraft, and e is Oswald's efficiency factor of the SUAV. In addition, $\rho(h)$ is the air density at altitude h given in [39]:

$$\rho(h) = \frac{p(h)}{RT(h)} = \frac{P_0\left(1 - 0.0065\left(h/T_0\right)\right)^{5.2561}}{R\left(T_0 - 6.5(h/1000)\right)}. \tag{5}$$

Here, P_0 is the sea level atmospheric pressure (101,325 N/m^2), T_0 is the sea level standard temperature (288.15°K), and R is the universal gas constant (287.04 m^2/°Ksec^2).

We assume that the SUAV flies with a constant optimum speed $V_{\mathrm{opt}} = (B/A)^{1/4}$, occurring when the D/L ratio is optimum ($D/L_{\mathrm{opt}} = 2\sqrt{AB}$) [20]. If we assume that the consumption efficient of the SUAV is 100 percent, then this ratio is proportionate to the energy consumption, with or without the effect of wind force.

```
Input: Elevation matrix M in $R^3$.
Output: $\Gamma$ = [N, P]
1.  FOR (each cut layer in the map; i = 1, k)
2.      cutPlane = Slice(M, $H_{min}$, $H_{max}$, k); //the parametric equation of the cut plane.
3.      FOR (each obstacle; j = 1, n)
4.          IF Obstacle -> highestNode(Z) < k * Dcut //check if the obstacle highest node is lower than the cut plane
5.          THEN
6.              For (each edge of the obstacle)
7.              tentativeNode = FindIntersection (cutPlane, edge);
8.              newNode = enlarge(tentativeNode, Dsafe);
9.              N = Add(newNode,obstacleIndex, facetIndex);
10.         ELSE
11.             newNode = [Obstacle -> highestNode(X)
12.                        Obstacle -> highestNode(X)
13.                        k* Dcut]
14.             N = Add(newNode,obstacleIndex, facetIndex1, facetIndex2);
15.             BREAK;
16. FOR (Each pair of Node in set N; i = 1,NumberOfNode; j = i + 1,NumberOfNode)
17.     IF (LOScheck() is TRUE)
18.     THEN
19.         P = Add(N(i), N(j));
20.         P = Add(N(j), N(i));
21. R = (N,P);
22. SaveToFile(R);
```

Algorithm 1. RoadmapCreate

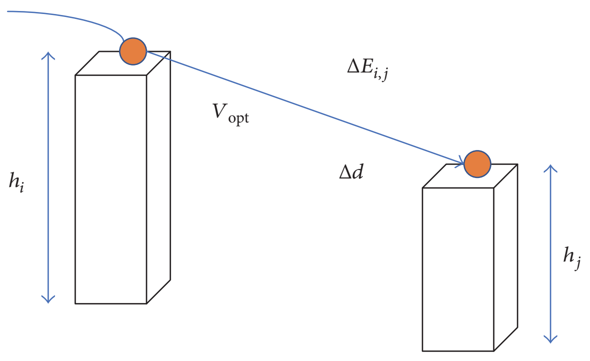

Figure 5: Energy consumption between two nodes in the visibility graph.

For a zero-wind scenario, the kinetic energy consumed will only be used to overcome the drag force D on a straight flight.

$$\Delta E_{\text{k}} = \int Ddxyz = D\Delta d. \quad (6)$$

Because the lift force L is equal to SUAV's weight, if the SUAV flies with the constant optimum Carson's speed, we have

$$\Delta E_{\text{k}} = W\left(\frac{D}{L}\right)_{\text{opt}} \Delta d = 2W\sqrt{AB}\Delta d \approx 2W\sqrt{\frac{f}{\bar{\rho}b^2\pi e}}\Delta d, \quad (7)$$

where $\bar{\rho} = (\rho(h_i) - \rho(h_j))/2$.

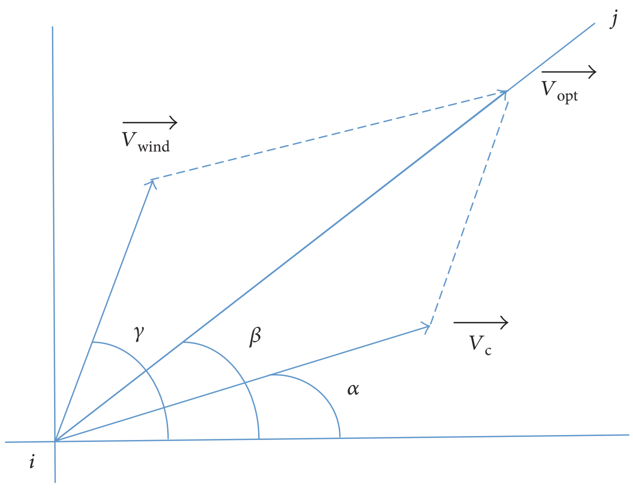
j
$\overrightarrow{V}_{\text{opt}}$
$\overrightarrow{V}_{\text{wind}}$
γ
β
$\overrightarrow{V}_{\text{c}}$
α
i

Figure 6: Speed adjustment under effect of wind velocity vector.

The next step includes the effect of the wind field. Practically, the operation range of SUAV is within 10 km, thus we can apply constant wind field in the configuration space. Considering a reference frame based on the plane comprising of wind velocity vector $\overrightarrow{\mathbf{V}}_{\text{wind}}$ and optimum speed $\overrightarrow{\mathbf{V}}_{\text{opt}}$ between node i and node j, let us call β and γ as the angle of $\overrightarrow{\mathbf{V}}_{\text{opt}}$ and $\overrightarrow{\mathbf{V}}_{\text{wind}}$ to this reference's horizontal axis, respectively. In order to keep the optimum speed, assuming that the minimum velocity of the SUAV can overcome the wind force, the SUAV needs to make

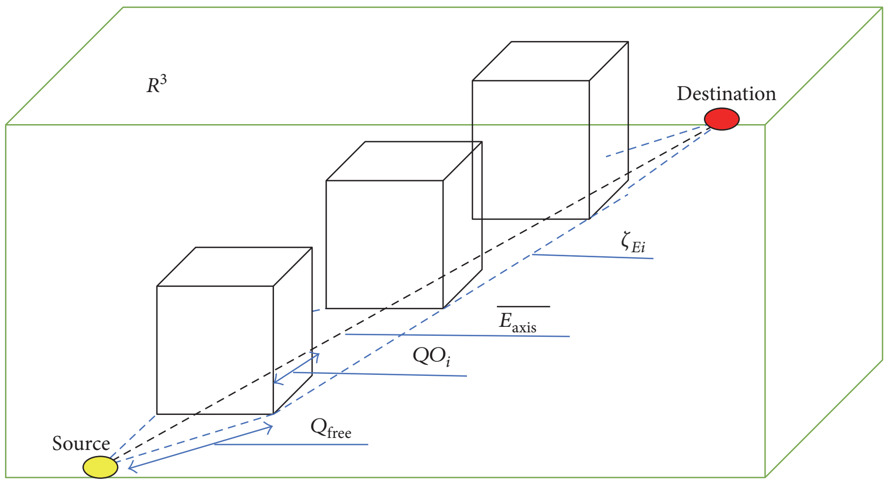

Figure 7: Illustration of the bounded energy space creation. The green outlined cuboid represents the configuration space R^3. The black dashed line represents the energy axis $\overline{E_{\text{axis}}}$. The blue dashed boundary represents the bounded energy space.

a turn with an angle α and a controlled speed $\overrightarrow{\mathbf{V}_c}$ that satisfy $\overrightarrow{\mathbf{V}_{\text{opt}}} = \overrightarrow{\mathbf{V}_{\text{wind}}} + \overrightarrow{\mathbf{V}_c}$ (Figure 6).

The duration for giving the controlled speed V_c equals to the duration used to travel with optimum speed from node i to node j, which means $t = \Delta d / V_{\text{opt}}$. The new equation for kinetic energy was derived as

$$\Delta E_k = D\Delta d = W\left(\frac{D}{L}\right)_c V_c t = W\left(AV_c^2 + \frac{B}{V_c^2}\right) V_c t. \quad (8)$$

The V_{opt}, in this case, can be computed by

$$V_{\text{opt}} = V_{\text{wind}} \cdot \cos(\gamma - \beta) + \sqrt{V_c^2 - V_{\text{wind}}^2 \sin^2(\gamma - \beta)}. \quad (9)$$

Following the calculation from [19], the consumed energy for turning between two arcs could be approximated by

$$\Delta E_{\text{turn}} \approx m\frac{D}{L} \sin \sigma |V|^2, \quad (10)$$

where σ is the steepest rolling angle that the SUAV needs to take for turning.

In the end, from (5), (8), and (9), the energy consumption for travelling between two nodes i and j is calculated as follows:

$$\begin{aligned}\Delta E_{i,j} = {} & \max\left(mg\left(h_i - h_j\right), 0\right) \\ & + \frac{W\left(AV_c^2 + B/V_c^2\right) V_c \Delta d}{V_{\text{wind}} \cos(\gamma - \beta) + \sqrt{V_c^2 - V_{\text{wind}}^2 \sin^2(\gamma - \beta)}} \\ & + \min\left(m\frac{D}{L} \sin \sigma |V|^2, 0\right).\end{aligned} \quad (11)$$

3.3. Bounded Energy Space. The time complexity of pathfinding increases with the increase in number of obstacles, layers, and facets in the visibility road map. To overcome this shortcoming, we define a bounded energy space in the region of interest to obtain an energy-efficient flight path with high probability. Figure 7 illustrates the energy space creation. First, draw a straight line between the source and destination which is called the energy axis. The shortest and energy-optimum path for SUAV is a straight line path in the absence of obstacles between source and destination. Second, select those obstacles whose edges intersect with the energy axis. By connecting the vertices of the selected obstacles, it will result in energy space 1 boundary. Third, select the obstacles whose edges intersect with the boundary of energy space 1 by connecting their vertices. It will result in energy space 2 boundaries. The obstacles that are inside energy space 1 and energy space 2 boundaries make a configuration space R^3. We called this configuration space as bounded energy space. The details for each step are given in Figure 7.

For a given 3D map, the configuration space for SUAV is simply defined as the set of configurations at which the SUAV can intersect with obstacle QO_i and the set of free configurations Q_{free}, that is,

$$\begin{aligned} QO_i &= \{q \in Q | R(q) \cap Wo_i \neq \Phi\}, \\ Q_{\text{free}} &= \{Q \setminus (\cup_i QO_i)\}, \end{aligned} \quad (12)$$

where q is configuration and Q is configuration space. Wo_i is the work space for SUAV as explained in [40]. Figure 7 shows the configuration space having a set of obstacle configuration QO_i and a set of free configurations Q_{free} in 3D space.

Energy axis $\overline{E_{\text{axis}}}$ is the straight energy optimum path between source and destination in the complete configuration space. Energy space QO_{e_i} consist a set of obstacle space that contains obstacles whose edges intersect with the energy axis we called energy space 1. The energy space QO_{e_i} can be represented as follows:

$$\begin{aligned} QO_{e_i} &= \left\{qo \in QO_i | q_0 \cap \overline{E_{\text{axis}}}\right\}, \\ \zeta_{Ei} &= \left\{\left(v_i + v_{i+1} + \cdots + v_{i-n}\right) \in QO_{e_i}\right\}, \end{aligned} \quad (13)$$

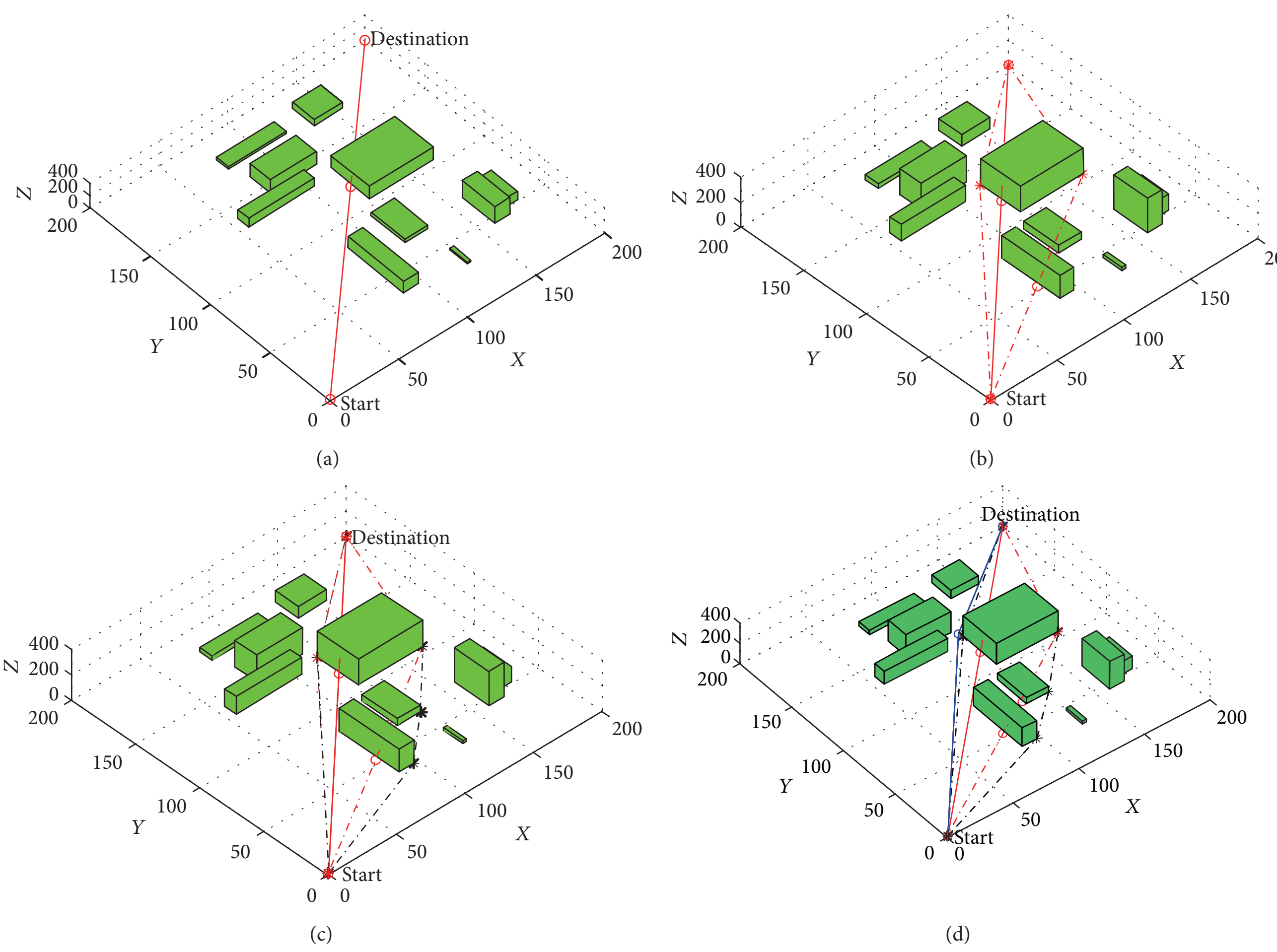

Figure 8: Explanation of bounded energy space selection algorithm. (a) Energy axis between source and destination. (b) Selection of energy space 1. (c) Selection of energy space 2. (d) Selection of path in the bounded energy space.

where ζ_{Ei} define the bounded energy space which contains the obstacles that intersect with the energy axis as shown in Figure 7. We can generalize (3) for energy space 2, 3, …, n as follows:

$$QO_{e_{i+1}} = \{qo \in QO_i | q_0 \cap \zeta_{Ei}\}, \quad i = 1, 2, 3, \dots, n. \tag{14}$$

In this paper, we will only extend our energy space up to 2. The energy space extension depends on the number of obstacles in the bounded energy spaces 1 and 2. Algorithm 2 shows the procedure of finding the bounded energy space.

A question which arises in the mind is, how will the proposed algorithm ensure an energy-efficient flight path in such a small portion of map? So, to solve this ambiguity, we perform 500 experiments on different maps with the number of obstacles varying from 5 to 50. In each experiment, the source and destinations are randomly selected with random placement of obstacles having different heights. From these experiments, we find the geometric position of the efficient energy path in the map. After analyzing the geometry of the efficient energy path, we start finding the probabilities of efficient energy path in our bounded energy space. From the results, we found that the probability is 0.80 to find an efficient path in energy space 1 and energy space 2. Figure 8 shows the workflow of the bounded energy space algorithm.

3.4. Searching Phase. The searching phase starts by loading the map configuration with obstacles data, nodes set, and edges set in the bounded energy space. This process takes $O(nkf)$ running time. Instead of using the Dijkstra search algorithm, we want to speed up the search process by using A* algorithm. Here, we minimize the determining function $f(n)$ by cost function $g(n) = \Delta E_{i,j}$ (11) and heuristic function $h(n)$ which is the Euclidian distance between the current processed node and the destination node. During the path selection process, we include a physical constraint of SUAV which is maximum flight steering angle to reduce the number of traversable edges. Then, we calculate the energy cost of expected traversable paths.

4. Evaluation and Discussion

We conducted a simulation to demonstrate the capability of finding the efficient energy consumption flight path of the visibility roadmap. The tested simulation hardware is an Intel

```
Input: 3D map with edge and node sets
Output: Bounded energy space $QO_{e_{i+1}}$
1.  $E_{\text{axis}}$ ← straight line between source and destination
2.  for each obstacle in the given map
3.          $QO_{e_i}$ ← select the obstacles intersecting with the $E_{\text{axis}}$
4.  end
5.  for each intersected obstacle
6.      $\zeta_{Ei}$ ← join their vertices with source and destination  // energy space 1 created
7.  end
8.  for each obstacle in the given map
9.          $QO_{e_{i+1}}$ ← select the obstacles intersecting with $\zeta_{Ei}$  // energy space 2 created
10. end
```

Algorithm 2. Bounded energy space selection

Table 1: Roadmap construction processing time.

Number of obstacles	Processing time
5	1.4 s
10	11.4 s
15	26.7 s
20	53.8 s
25	101 s
30	133 s

Table 2: Environment parameters.

Variable	Value	Unit
D_{cut}	20	Meter
D_{safe}	10	Meter
V_{wind}	5	Knot
horizontal_wind	Pi/4	Radius
vertical_wind	0	Radius

Table 3: SUAV parameters.

Variable	Value	Unit
V_{min}	10	Knot
V_{max}	50	Knot
Mass	25	kg
f (parasite area)	0.02	Meter square
b (wing span)	1	Meter
e (Oswald's factor)	0.7	
Maximum steering angle	Pi/6	Radius

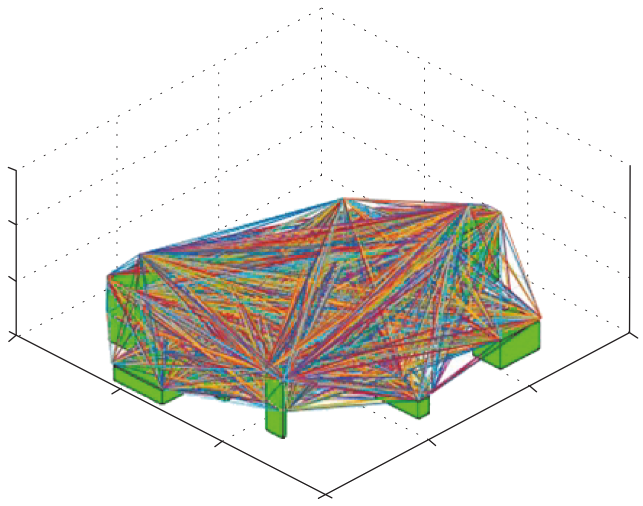
Figure 9: A sample of roadmap visualization in the simulation with 10 obstacles.

Core i5 2.80 GHz CPU and 8 GB memory desktop. In this simulation, we generate the convex polyhedral obstacles arbitrarily. As expected, the time to generate the roadmap increases rapidly as the number of obstacles rises (Table 1). Figure 9 illustrates a full connecting roadmap.

The parameters of the SUAV and the environment are given in Tables 2 and 3.

To the best of our knowledge, the uniform grid algorithm proposed by Nachmani [19] is the most updated and similar work to ours. In Nachmani's method, a grid cell only considers nine adjacent cells which located on the straight path from the start node to the destination node as neighbor nodes. Therefore, we implement his method as a comparison method for this approach. To be fair-minded, we also separate his algorithm into 2 phases: map modeling phase and searching phase. In the searching phase, we use A* algorithm for both methods. We present the visualization of these two methods in Figure 10. The red plot indicates our method and the blue plot represents the grid method.

Based on the simulation results (Tables 4 and 5), we explicitly see that the path generated by the visibility roadmap always has a reduced energy consumption compared with the grid method (12% in average of 50 runs). This is due to the reduction of distance of the former algorithm which is 5% less than the later algorithm and 62% smaller number of heading changes. In most of the cases, both

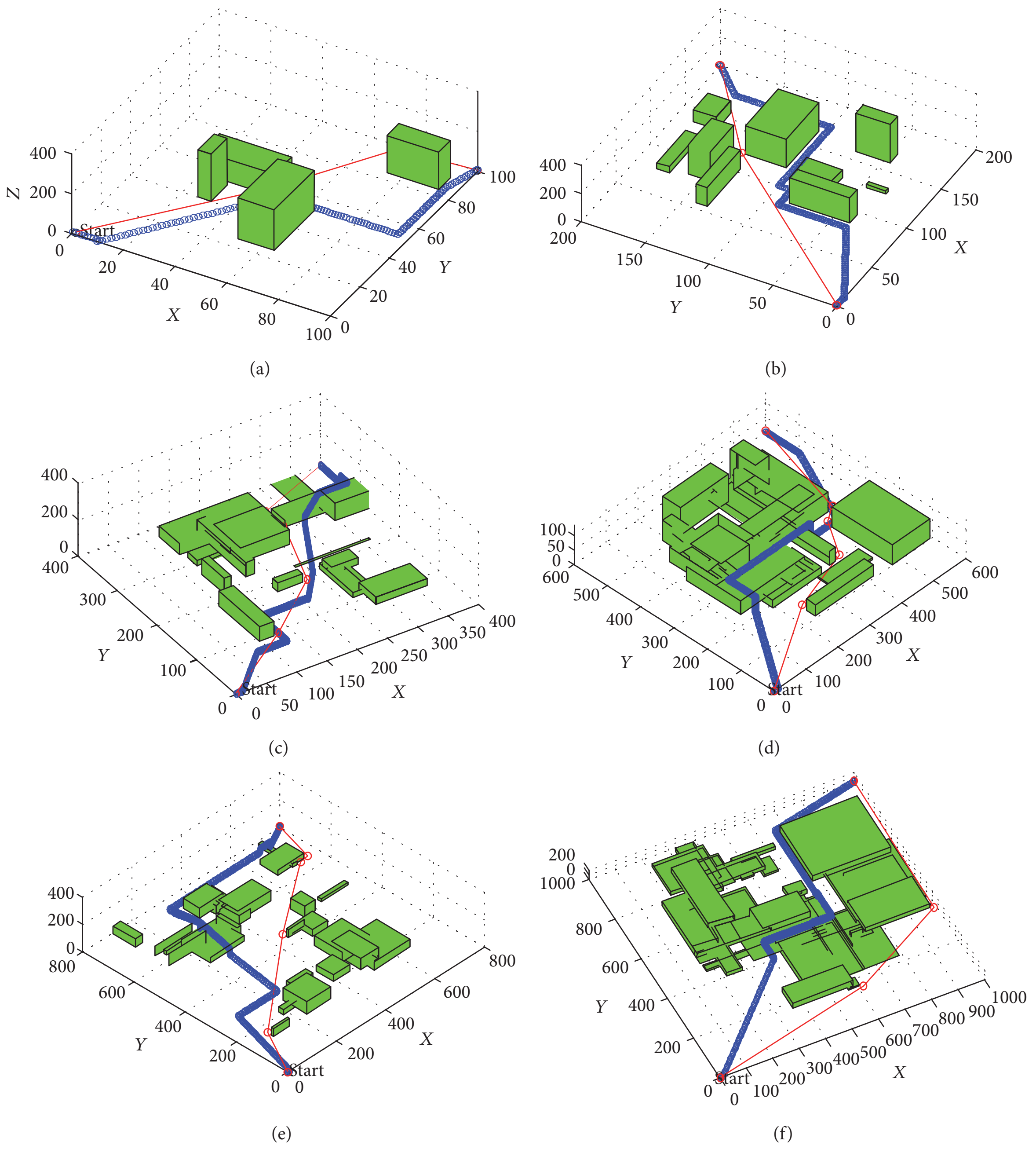

Figure 10: Simulation of two pathfinding algorithms: (a) 5 obstacles; (b) 10 obstacles; (c) 15 obstacles; (d) 20 obstacles; (e) 25 obstacles; (f) 30 obstacles.

Table 4: Simulation results based on number of tests.

Number of tests	Proposed visibility roadmap			Grid method		
	Energy	Distance	Heading changes	Energy	Distance	Heading changes
5	38,246	879.8	2.4	42,050	905.2	6.8
10	37,945	872.6	2.8	41,895	902.9	7.8
20	37,905	866.2	3.1	47,678	938.3	8
30	37,746	868	3.7	41,690	898.3	7.5
40	38,724	879.2	2.9	42,530	911.9	8.5
50	38,507	877.8	3.1	43,684	928.1	9.1

The energy unit is calculated in Joule, and the unit of distance is in meters. Map size is $600 \times 600 \times 300$ (meters) with 20 obstacles.

Table 5: Simulation results based on map size.

Map size/number of obstacles	Proposed visibility roadmap		Grid method	
	Distance	Energy	Distance	Energy
$100 \times 100 \times 300/5$	187.76	1.89×10^4	439.33	6.72×10^4
$200 \times 200 \times 300/10$	303.75	2.37×10^4	489.11	6.93×10^4
$400 \times 400 \times 300/15$	604.03	2.90×10^4	693.65	4.36×10^4
$600 \times 600 \times 300/20$	820.83	3.81×10^4	875.4	5.61×10^4
$800 \times 800 \times 300/25$	1172.2	5.29×10^4	1296.7	6.89×10^4
$1000 \times 1000 \times 300/30$	1696.9	7.50×10^4	1710.1	8.59×10^4

The energy unit is calculated in Joule, and the unit of distance is in meters. These are normalized values after five stochastic runs.

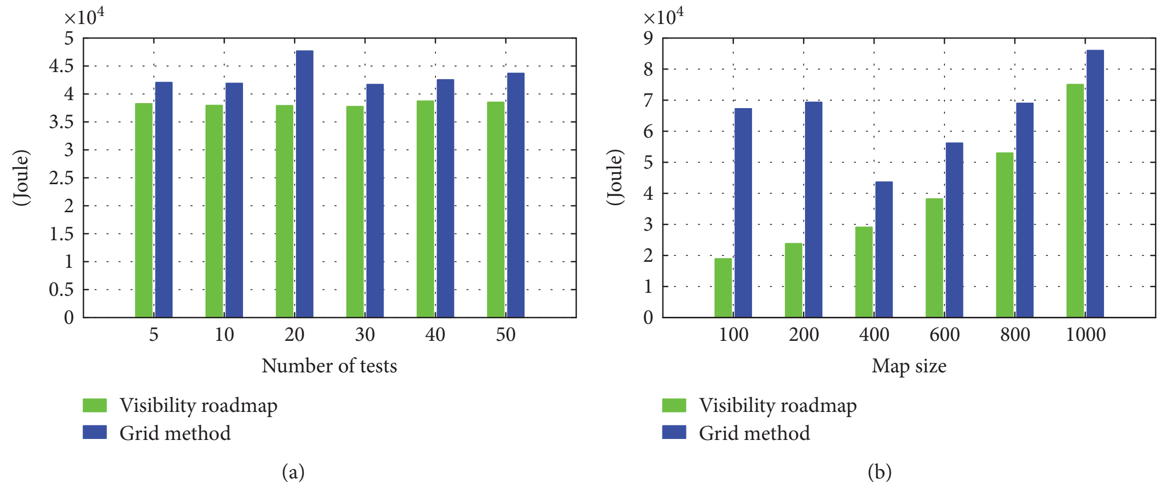

Figure 11: Comparison of two algorithms regarding energy consumption: (a) based on number of tests in Table 5; (b) based on map size in Table 6.

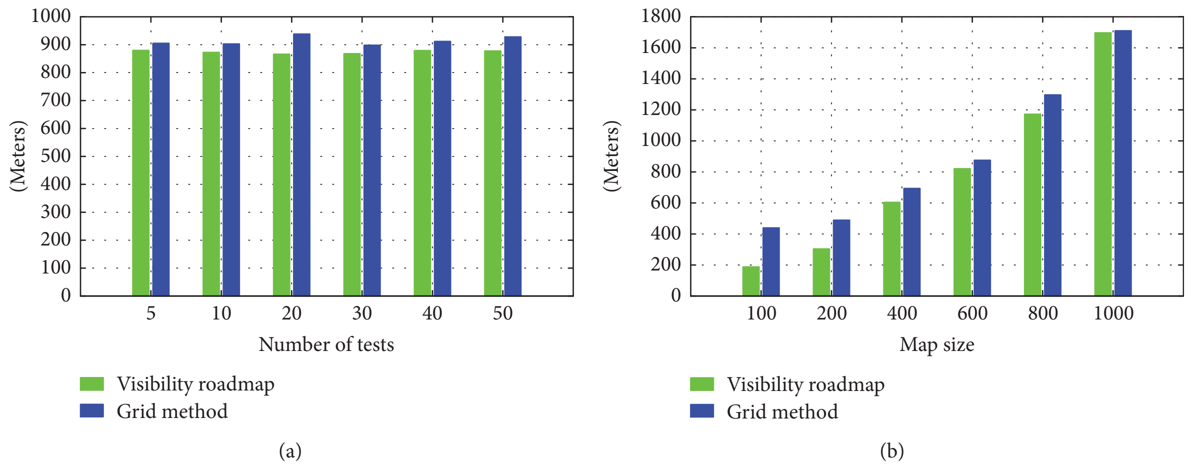

Figure 12: Comparison of two algorithms regarding traveled distance: (a) based on number of tests in Table 5; (b) based on map size in Table 6.

methods could find a similar path in terms of elevation. Figures 11 and 12 visualize the difference between performances of the two compared methods.

Figure 13 reports the computational efficiency of the proposed visibility roadmap scheme in comparison with grid method in synthetically generated maps. The average computation time taken by the proposed technique for pathfinding is only 0.68 s, which is much lower than the computation time of the grid method. This is expected since our approach does not use the complete visibility

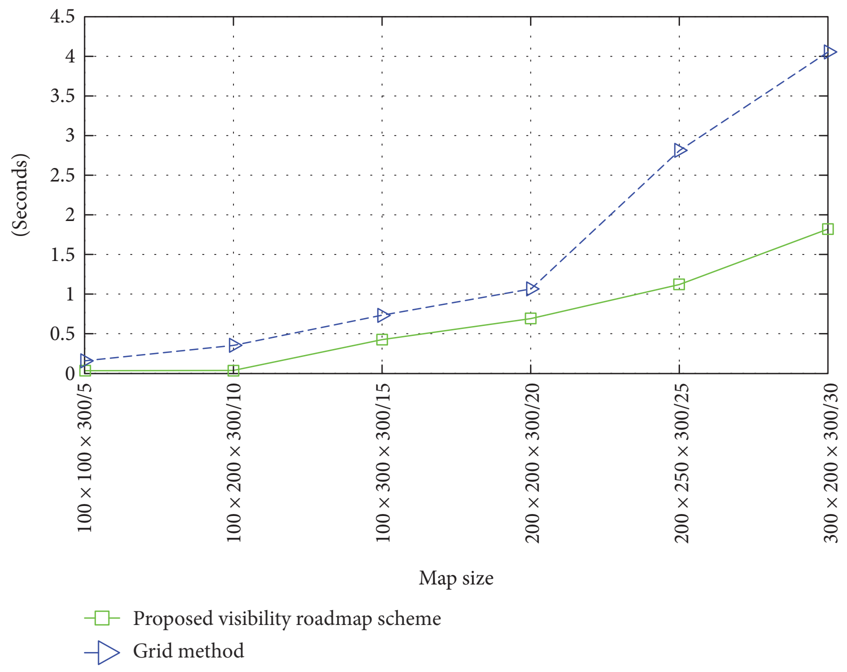

Figure 13: A comparison in terms of computing speed between two pathfinding algorithms.

Table 6: Simulation results based on NYC Open Data.

Dataset number	Proposed visibility roadmap		Grid method	
	Distance	Energy	Distance	Energy
NYC dataset 1	823.63	34,153	874.5	39,791
NYC dataset 2	907.1027	41,684	997.24	49,781
NYC dataset 3	785.006623	32,196	956	56,171

roadmap. The computational efficiency of the proposed technique will be useful for real-time pathfinding in static maps for UAVs.

To compare our proposed algorithm to the existing grid method, we also conduct some experiments on real-world datasets available at NYC Open Data (2017). Table 6 shows the performance of the proposed method which shows 10% improvement in path length an average of 49.8% improvement in energy than the existing grid method. The average computation time for the proposed method is 1.8 s when the number of obstacle is 30 in the selected dataset.

5. Conclusions

This paper presented an algorithm of finding the most efficient power consumption flight path for the small unmanned aerial vehicle (SUAV) in the 3-dimensional terrain provided by a digital map. The research contributed a solution to the sprouting demand for an obstacle-free path that efficiently utilize the battery power to perform the task of SUAV. In this paper, we dealt with the problem of efficient energy pathfinding in an environment with a number of convex obstacles and wind force field. We introduced the 3-D visibility roadmap which modeled airspace from a digital map to a graph representation using the bounded energy space. First, we model the visibility roadmap of the bounded energy space to reduce the time complexity. Second, we found efficient energy path in the graph using the graph search method based on the energy estimation cost function. The empirical results indicated that the proposed scheme had better results to find efficient energy path in less and comparable time complexity. In future work, we are focusing on real-time unknown obstacle detection and avoidance.

Nomenclature

M:	Digital map data
Γ:	Roadmap
N:	Traversable waypoints set in space
P:	Traversable paths set between waypoints
$б$:	Maximum steering angle
$H_{\min}$:	Minimum flight height
$H_{\max}$:	Maximum flight height
D_{safe}:	Safe distance when approaching obstacles
n:	Number of obstacles
γ:	Heading angle
B:	Wind vector angle
K:	Number of cut layers
E:	Energy consumption
LOS:	The line of sight between two waypoints
$Z_{\max}$:	Maximum altitude of an obstacle
d_{cut}:	Distance between cut layers.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by Institute for Information & Communications Technology Promotion (IITP) grant funded by the Korean Government (MSIT) (no. 2015-0-00893, Technology Development of DMM-based Obstacle Avoidance and Vehicle Control System for a Small UAV).

References

[1] S. Chaudhuri and V. Koltun, "Smoothed analysis of probabilistic roadmaps," *Computational Geometry*, vol. 42, no. 8, pp. 731–747, 2009.

[2] A. H. Qureshi and Y. Ayaz, "Intelligent bidirectional rapidly-exploring random trees for optimal motion planning in complex cluttered environments," *Robotics and Autonomous Systems*, vol. 68, pp. 1–11, 2015.

[3] E. S. Kaur, "Shortest path finding algorithm using ant colony optimization," *International Journal of Engineering Research & Technology*, vol. 2, no. 6, pp. 317–326, 2013.

[4] I. K. Nikolos, K. P. Valavanis, N. C. Tsourveloudis, and A. N. Kostaras, "Evolutionary algorithm based offline/online path planner for UAV navigation," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 33, no. 6, pp. 898–912, 2003.

[5] B. Ranjbar-Sahraei, K. Tuyls, I. Caliskanelli et al., "Bio-inspired multi-robot systems," *Biomimetic Technologies*, pp. 273–299, 2015.

[6] L. De Filippis, G. Guglieri, and F. Quagliotti, "Path planning strategies for UAVS in 3D environments," *Journal of Intelligent & Robotic Systems*, vol. 65, no. 1–4, pp. 247–264, 2012.

[7] R. Oliva and N. Pelechano, "NEOGEN: near optimal generator of navigation meshes for 3D multi-layered environments," *Computers & Graphics*, vol. 37, no. 5, pp. 403–412, 2013.

[8] H. Tong, "Path planning of UAV based on Voronoi diagram and DPSO," *Procedia Engineering*, vol. 29, pp. 4198–4203, 2012.

[9] M. De Berg, M. Van Kreveld, M. Overmars, and O. C. Schwarzkopf, *Computational Geometry*, Springer, Berlin, Heidelberg, 2000.

[10] B. Dasgupta, *Robot Motion Planning Methods: An Overview*, Department of Mechanical Engineering Indian Institute of Technology Kanpur, 2007.

[11] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[12] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[13] D. D. Harabor and A. Grastien, "Improving Jump Point Search," in *Proceedings of the Twenty-Fourth International Conference on Automated Planning and Scheduling*, pp. 128–135, Portsmouth, NH, USA, 2014.

[14] D. Ferguson and A. Stentz, "Using interpolation to improve path planning: the Field D* algorithm," *Journal of Field Robotics*, vol. 23, no. 2, pp. 79–101, 2006.

[15] A. Nash, K. Daniel, S. Koenig, and A. Feiner, "Theta*: any-angle path planning on grids," in *Proceedings of the National Conference on Artificial Intelligence*, pp. 1177–1183, Vancouver, BC, Canada, 2007, AAAI Press.

[16] M. G. Park, J. H. Jeon, and M. C. Lee, "Obstacle avoidance for mobile robots using artificial potential field approach with simulated annealing," in *ISIE 2001. 2001 IEEE International Symposium on Industrial Electronics Proceedings (Cat. No.01TH8570)*, pp. 1530–1535, Pusan, 2001, IEEE.

[17] T. Paul, T. R. Krogstad, and J. T. Gravdahl, "Modelling of UAV formation flight using 3D potential field," *Simulation Modelling Practice and Theory*, vol. 16, no. 9, pp. 1453–1462, 2008.

[18] S. Scherer, S. Singh, L. Chamberlain, and M. Elgersma, "Flying fast and low among obstacles: methodology and experiments," *The International Journal of Robotics Research*, vol. 27, no. 5, pp. 549–574, 2008.

[19] G. Nachmani, "Minimum-energy flight paths for UAVs using mesoscale wind forecasts and approximate dynamic programming," DTIC Document, 2007.

[20] B. H. Carson, "Fuel efficiency of small aircraft," *Journal of Aircraft*, vol. 19, no. 6, pp. 473–479, 1982.

[21] D. K. Phung and P. Morin, "Modeling and energy evaluation of small convertible UAVs," in *2nd Workshop on Research, Education and Development of Unmanned Aerial Systems*, Compiègne, France, November 2013.

[22] M. Wagner, "SRTM DTED format," 2003, Product Description SRTM/PD03/11/03, Version 1.1.

[23] D.-T. Lee, "Proximity and reachability in the plane," 1978, DTIC Document.

[24] T. Asano, T. Asano, L. Guibas, J. Hershberger, and H. Imai, "Visibility of disjoint polygons," *Algorithmica*, vol. 1, no. 1–4, pp. 49–63, 1986.

[25] H. Edelsbrunner, *Algorithms in Combinatorial Geometry, Vol. 10*, Springer Science & Business Media, 2012.

[26] E. Welzl, "Constructing the visibility graph for n-line segments in O(n^2) time," *Information Processing Letters*, vol. 20, no. 4, pp. 167–171, 1985.

[27] J. S. B. Mitchell, "Shortest paths among obstacles in the plane," *International Journal of Computational Geometry & Applications*, vol. 06, no. 03, pp. 309–332, 1996.

[28] J. Hershberger and S. Suri, "Efficient computation of Euclidean shortest paths in the plane," in *Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science*, pp. 508–517, Palo Alto, CA, USA, 1993, IEEE.

[29] H. Rohnert, "Shortest paths in the plane with convex polygonal obstacles," *Information Processing Letters*, vol. 23, no. 2, pp. 71–76, 1986.

[30] J. Canny, *The Complexity of Robot Motion Planning*, MIT press, 1988.

[31] K. Jiang, L. S. Seneviratne, and S. W. Earles, "Finding the 3D shortest path with visibility graph and minimum potential energy," *IEEE/RSJ International Conference on Intelligent Robots and System*, vol. 1, pp. 679–684, 1993.

[32] J. H. Reif and J. A. Storer, "A single-exponential upper bound for finding shortest paths in three dimensions," *Journal of the ACM*, vol. 41, no. 5, pp. 1013–1019, 1994.

[33] V. Akman, *Unobstructed Shortest Paths in Polyhedral Environments, Vol. 251*, Springer Science & Business Media, 1987.

[34] J. Choi, J. Sellen, and C.-K. Yap, "Precision-sensitive Euclidean shortest path in 3-space," in *Proceedings of the Eleventh Annual Symposium on Computational Geometry*, pp. 350–359, Vancouver, BC, Canada, 1995, ACM.

[35] J. Choi, J. Sellen, and C.-K. Yap, "Approximate Euclidean shortest paths in 3-space," *International Journal of*

Computational Geometry & Applications, vol. 07, no. 04, pp. 271–295, 1997.

[36] T. Lozano-Pérez and M. A. Wesley, "An algorithm for planning collision-free paths among polyhedral obstacles," *Communications of the ACM*, vol. 22, no. 10, pp. 560–570, 1979.

[37] C. H. Papadimitriou, "An algorithm for shortest-path motion in three dimensions," *Information Processing Letters*, vol. 20, no. 5, pp. 259–263, 1985.

[38] J. F. Hughes and J. D. Foley, *Computer Graphics: Principles and Practice*, Pearson Education, 2014.

[39] M. Cavcar, "The international standard atmosphere (ISA)," *Anadolu University, Turkey*, vol. 30, 2000.

[40] H. Choset, K. Lynch, S. Hutchinson et al., *Principles of Robot Motion: Theory, Algorithms, and Implementation*, pp. 1–150, 2007.