





Article

Bézier Curves-Based Optimal Trajectory Design for Multirotor UAVs with Any-Angle Pathfinding Algorithms

Haitham AL Satai ¹, Musaddak M. Abdul Zahra ^{2,3}, Zaid I. Rasool ², Ridhab Sami Abd-Ali ²
and Catalin I. Pruncu ^{4,5,*}

¹ School of Electronics Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China; haithamalsatai@gmail.com

² Computer Techniques Engineering Department, Al-Mustaqbal University College, Babylon 51001, Iraq; musaddaqmahir@mustaqbal-college.edu.iq (M.M.A.Z.); Zaid.Ibrahim@mustaqbal-college.edu.iq (Z.I.R.); RidhabSami@mustaqbal-college.edu.iq (R.S.A.-A.)

³ Electrical Engineering Department, College of Engineering, University of Babylon, Babylon 51001, Iraq

⁴ Department of Mechanical Engineering, Imperial College London, Exhibition Rd., London SW7 2AZ, UK

⁵ Design, Manufacturing & Engineering Management, University of Strathclyde, Glasgow G1 1XJ, Scotland, UK

* Correspondence: c.pruncu@imperial.ac.uk or catalin.pruncu@strath.ac.uk

Abstract: Multirotor Unmanned Aerial Vehicles (UAVs) play an imperative role in many real-world applications in a variety of scenarios characterized by a high density of obstacles with different heights. Due to the complicated operation areas of UAVs and complex constraints associated with the assigned mission, there should be a suitable path to fly. Therefore, the most relevant challenge is how to plan a flyable path for a UAV without collisions with obstacles. This paper demonstrates how a flyable and continuous trajectory was constructed by using any-angle pathfinding algorithms, which are Basic Theta*, Lazy Theta*, and Phi* algorithms for a multirotor UAV in a cluttered environment. The three algorithms were modified by adopting a modified cost function during their implementation that considers the elevation of nodes. First, suitable paths are generated by using a modified version of the three algorithms. After that, four Bézier curves-based approaches are proposed to smooth the generated paths to be converted to flyable paths (trajectories). To determine the most suitable approach, particularly when searching for an optimal and collision-free trajectory design, an innovative evaluation process is proposed and applied in a variety of different size environments. The evaluation process results show high success rates of the four approaches; however, the approach with the highest success rate is adopted. Finally, based on the results of the evaluation process, a novel algorithm is proposed to increase the efficiency of the selected approach to the optimality in the construction process of the trajectory.

Keywords: path planning; trajectory planning; Basic Theta*; Lazy Theta*; Phi*; Bézier curves



Citation: Satai, H.A.; Zahra, M.M.A.; Rasool, Z.I.; Abd-Ali, R.S.; Pruncu, C.I. Bézier Curves-Based Optimal Trajectory Design for Multirotor UAVs with Any-Angle Pathfinding Algorithms. *Sensors* **2021**, *21*, 2460. <https://doi.org/10.3390/s21072460>

Academic Editors: Sung Kyung Hong and Le Nhu Ngoc Thanh Ha

Received: 26 February 2021

Accepted: 26 March 2021

Published: 2 April 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In recent years, multirotor Unmanned Aerial Vehicles (UAVs) have been getting considerable attention from many, whether researchers or ordinary people due to many features; for instance, their unique abilities to fly in various environments, whether indoor areas or outdoor densely populated civilian areas. Moreover, a unique capability of flying in any direction makes these flying vehicles suitable as a good research platform and in numerous real-world applications and missions. In addition to their low cost, high operability in various tasks with significant autonomy, and the replacement of humans in fundamental and hazardous activities, this makes UAVs preferably appropriate for unclean, tedious, or hazardous tasks compared to the conventional aircrafts that carry a human. In brief, UAVs are vehicles that can fly without a human pilot onboard [1]. There is a wide range of applications of UAVs, whether in military strategy or civilian applications, such as reconnaissance, archaeology, load transportation, surveillance, disaster relief, search and

rescue, terrain mapping, environment protection, dangerous site inspection, as a 5G flying base station for providing wireless mobile connectivity, integrating renewable energies, etc. [2–9]. Since UAVs are ubiquitous in a variety of scenarios characterized by a high density of obstacles with different heights such as warehouses, houses, and cities, the most relevant challenge is how to plan a safe and flyable path for UAVs without collisions with obstacles. Consequently, the problem of planning a suitable and flyable path for a UAV in a cluttered environment is one of the most outstanding research areas.

Path planning considers a significant process in robotic planning, especially in the operation of UAVs. It can be defined as a process of finding a low cost, feasible, optimal/near-optimal, shortest traversal path between two points (start and goal points) by meeting specific operational restrictions. Various restrictions should be taken into account when planning a path for a UAV, such as environmental uncertainties, acceleration and velocity restrictions, and inaccessible points in space. In addition to the inconvenience of lower battery duration, a myriad of the current research works consider the shortest distance of the UAV flight as the optimal aim to construct the optimal path. They merely think that the shortest the path, the least amount of energy will be consumed, forgetting the fact that the energy of the UAV can also be consumed during the change in direction (heading) in its flight. Moreover, further details can be accessed from the literature [9–11]. Planning a path for a multicopter UAV has numerous features compared to planning a path for a robot. The main dissimilarities for planning a path for a UAV are described as follows; first, the size of the UAV is small compared to space; it is usually represented in space as a moving point, which makes the pathfinding problem much simpler. Second, due to the UAV's high speed, the UAV does not have the possibility to turn at an acute angle in their path. This is totally different from the movement of the robot, in which it can move slowly or even halt to rotate. Due to this matter, the path of the UAV should be smooth to avoid these turns [12,13]. Third, the UAV has the ability to alter its flight altitude (for instance, there is a UAV flying in a cluttered environment, and sometimes there is a need to change its altitude to avoid obstacles) compared to the robot that commonly moves on the earth's surface. Therefore, the issue of planning a path for a robot is usually defined on the plane [14–16]. The three aforementioned features of planning a path for a UAV have been considered in this paper.

Interestingly, the growing interest and rapid development in planning the path for UAVs, as well as data stitching, has led to the emergence of several auxiliary desktop applications, such as pixel4D and drone2map that facilitate this process in terms of processing the collected data by lightweight, UAV or aircraft. For instance, the pixel4D high-resolution capabilities allow for retrieving the elevation and even the temperature value of each pixel of taken pictures to create highly accurate maps. In addition to Drone2Map that allows the user to process imagery from commercial drones and produce 2D, 3D products such as orthoimagery and 3D meshes that easily integrate with the ArcGIS platform. It also provides workflows for inspection of critical infrastructure and monitoring of situations. These provide significant cost and time savings over conventional survey methods, producing information on the same day as imagery collection. The ArcGIS platform, in turn, allows for the sharing of the information across organizations immediately [17].

In view of the abovementioned importance of studying path planning in UAVs, it is imperative when UAVs fly in cluttered environments that they have to proceed with their task while adhering to the operational restrictions and ensuring safe navigation concerning the encompassing objects. Different approaches to UAVs path planning and for various techniques of measurement employed by the UAVs are demonstrated in [18–24]. In addition, several inclusive reviews for path and trajectory planning were conducted for UAVs [25], ground vehicles [26], and mobile robots [27–29]. While the methods may be classified in a different manner into categories, several approaches employed fall within the “grid-based” definition for flying vehicles. The path is constructed on the top of the environment's nodes, in which this environment is discretized into a regular grid, where

some of these nodes represent the free areas, which allow the transit, whereas if they restrain the transit they represent the obstacles.

For planning a path, several grid search algorithms have been adopted to solve this problem; the most famous algorithms are Dijkstra's algorithm [30] and its variant A* algorithm [31]. Although many years have passed since their introduction, they are still frequently in use and no research work is free of their existence, in addition to the emergence of many new algorithms that were built on their basis, especially A* algorithm. These algorithm developments include Lifelong Planning A* (LPA*) [32], Focussed D* [33], Theta* and Angle-Propagation Theta* (AP Theta*) [34,35], whereas, the alternative version of Theta* algorithm is Basic Theta* that includes Lazy Theta* [36], Phi* and Incremental Phi* [37].

Due to the large number of algorithms that are incorporated into solving the path planning problem, the choice was made on three of any-angle pathfinding algorithms in this paper, which are Basic Theta*, Lazy Theta*, and Phi* algorithms. The selection of these algorithms is motivated by their short runtime and paths, their capability to create superior paths that are more realistic looking and close to the actual paths, and with fewer numbers of unnecessary heading changes and waypoints. Consequently, the paths of these algorithms are more proper for smoothing and make them more congruent with the motion of the UAVs. Above all, research work is scarce for generating smooth trajectories using these algorithms because most researchers dealt with other algorithms such as A* algorithms. Therefore, these three algorithms still have room to explore by smoothing their generated paths to create collision-free trajectories.

In general, the created grid-defined paths encompass a series of segments, which are determined by acute turns unsuitable for the aircraft flight. Due to this limitation, a model solution is presented by defining smoothing curves, which are well suited with the UAVs' flight that allows for compelling their kinematic restrictions, where a smooth trajectory has been created by parametrizing the time of the curve [38]. The construction of the path smoothing is carried out with different mathematical curves, usually polynomial, clothoid, or spline curves [26]. The use of convex optimization and successive convex approximation (SCA) have been applied to optimize the path [39,40]. Furthermore, the curves of Bézier, which are parametric curves are repeatedly utilized in computer graphics, animation, industrial design, modeling, and many other related fields. Moreover, these curves can be applied efficiently in the robotics scope. As a matter of fact, these curves ensure considerable flexibility and low cost of the computational processes due to the fact that Bézier curves are based on polynomials shaped with the utilize of suitable control points. Therefore, Bézier curves take their role to generate a continuous smooth path; consequently, numerous applications exist of Bézier curves for flying and ground vehicles (see references [41–44]). For example, a generated path from a parallel genetic algorithm for Multi-UAV systems was smoothed by utilizing Bézier curves [45]. A dynamic collision-free trajectory was designed for UAVs by applying a six-order Bézier curve [46].

In this paper, a flyable and smooth path (trajectory) was constructed for a multirotor UAV. First, a feasible and suitable path is generated by using a modified any-angle path search algorithm that adopts a modified cost function. The modification includes considering the elevation of the grid map nodes that the multirotor UAV may encounter while flying from the starting location to its destination. After that, Bézier curves have been exploited for constructing a smooth curve (trajectory) that makes the UAV fly smoothly without being bounded to the acute turns of the grid-defined path. Four Bézier curves-based approaches were presented to smooth the generated paths from the pathfinding algorithms. Subsequently, an innovative evaluation process is applied to determine the most efficient approach of the four proposed approaches in a variety of different size environments. Then, the evaluation process is repeated to guarantee that the generated trajectory from the previous evaluation process is safe to be adopted. Finally, a novel algorithm is introduced to cope with the preferable approach of the proposed Bézier curves approaches directly to conduct it to the optimum state during the trajectory generating process.

This paper is arranged as follows: Section 2 presents the strategy of path planning. Next, the trajectory design is described in Section 3. After that, the results and discussion are illustrated in Section 4. Finally, the conclusions of the research are in Section 5.

2. Path Planning

Path planning is a process of generating a feasible and suitable path in a cluttered environment for a UAV from a specific start point to a specific goal point. In this study, the path of the flight is created with Basic Theta*, Lazy Theta* and Phi* algorithms within a 2D grid environment. Additionally, the definition of the grid and its data structure is primarily demonstrated in this section. Subsequently, the three any-angle pathfinding algorithms and the modified cost function are described.

2.1. Grid Representation

This section explains the properties of the environment upon which the pathfinding algorithms were implemented by creating a simple grid (binary coded) with dimensions of 50×50 nodes, which represents a known cluttered environment as shown in Figure 1a. The grid (a simple matrix with 50×50 dimensions) is used to represent the environment, where it is occupied with obstacles that are marked as 1, whereas the free spaces are marked as 0. Furthermore, the obstacles were randomly distributed over the grid with discrete values from 0 to 100, which represent their elevations, in addition to the change in their locations and elevations at each new search of the algorithms. Compared to other representation methods, the grid representation method is simple in structure and easy to implement, so it is commonly used as a way to represent the environment for path planning. To increase the integrity of the path, a safety feature (safe zone) with yellow color was augmented around the obstacles as shown in Figure 1b. The elevation of the grid map nodes, which represents E , is classified into three categories, and the severity of their danger on the UAV during the flight is relative to the designated flight altitude of the UAV, which is set to 60 during the implementation of the algorithm “drone_altitude = 60”. The elevation of the grid map nodes is set as follows:

1. $E < 50$. These nodes were highlighted in magenta color. These nodes represent the safest elevation for the UAV to fly above taking into consideration the UAV's flight altitude.
2. $50 < E < 59$. These nodes were highlighted in red color. They are defined as a risky elevation, which is close to the flight altitude of the UAV and it is preferred to be avoided unless there is another path to be traversed, so it will be used to fly above.
3. $E > 60$. These nodes were highlighted in black color. They can be avoided by the UAV while it is trying to arrive at its goal by turning around them.

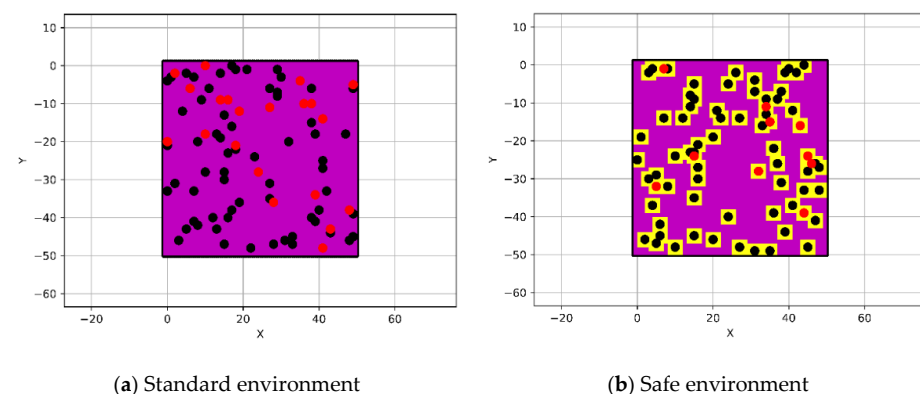


Figure 1. The configuration of the cluttered environment.

2.2. Any-Angle Pathfinding Algorithms

Any-angle pathfinding algorithms are developed to address some of the issues in the standard pathfinding algorithms such as A* algorithm by achieving a short unblocked path from a specific initial point to a specific target point with less computational time. Therefore, one of these issues of the standard pathfinding algorithms that have been addressed by these algorithms is the dissemination of information along graph borders and restrict paths to be constructed by these borders, as shown in Figure 2a. As a result, this restriction makes the resulting paths long and unrealistic looking compared to the actual shortest paths. For that reason, Nash and Koenig [47] developed any-angle algorithms, which disseminate information along graph borders, without restricting paths to be constructed by these borders. They can be utilized to briskly find short paths while retaining the characteristics of the standard algorithms, as shown in Figure 2b. Any-angle algorithms are based on A* algorithm which are namely Basic Theta* [34,35], Angle-Propagation Theta* (AP Theta*) [34,35], Lazy Theta* [36] and Phi* and Incremental Phi* [37] algorithms.

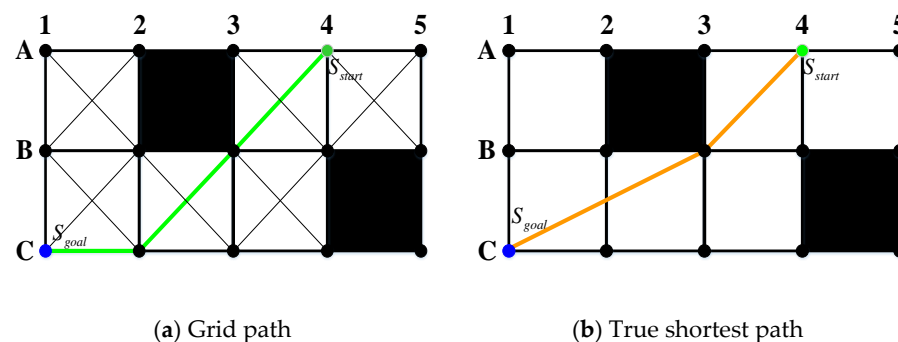


Figure 2. Standard find-path algorithm vs. any-angle find-path algorithm paths.

2.2.1. Basic Theta* Algorithm

Basic Theta* algorithm is an alternative version of A* algorithm that disseminates information along graph edges without restricting the paths to be constructed by graph edges [34,35]. Additionally, it allows the direction of the path to be changed at any angle. Consequently, the motivation for utilizing the Basic Theta* algorithm is the capability of generating paths that are shorter and more realistic-looking paths than the resulting paths from the standard algorithms. Basic Theta* performs a line of sight checking, where if there is a free line of sight between two nodes, the path skips the neighbor nodes, otherwise, it traverses across the adjacent nodes as A* algorithm. Thus, the essential difference between Basic Theta* and A* algorithms is that Basic Theta* algorithm allows the parent of a node to be any node, unlike A* algorithm where the parent must be a visible neighbor. This algorithm is correct, complete, and marginally slower than A* algorithm.

2.2.2. Lazy Theta* Algorithm

Lazy Theta* algorithm is the second example of any-angle pathfinding algorithms and a different version of Basic Theta* algorithm that highlights some of the deficiencies in Basic Theta* algorithm in the issue of the line of sight check when the dimension of the environment gets larger. This algorithm adopts a different idea in the matter of line-of-sight testing that differs from its predecessor; that is, it does not perform a line-of-sight test unless it is necessary. Therefore, the number of checks will be reduced, thus making it faster than the Basic Theta* algorithm. This technique is called a lazy evaluation technique, where lazy indicates the name of the algorithm itself [36]. The algorithm is correct, complete, and faster than its predecessor.

2.2.3. Phi* Algorithm

Phi* algorithm is another version of Basic Theta* algorithm, which is more convenient for working in unknown environments compared to the aforementioned algorithms, espe-

cially with its incremental version. It could be incremental as a result of its concentration on some of the insufficiency in its predecessor by preserving the feature of Local Parent Path [37]. Therefore, it is described as an incremental algorithm, which means it does not need to recompute the path from scratch by each newly blocked cell, where it reuses information from preceding searches to accelerate the next one.

The main difference between Phi* algorithm and its predecessor is that Phi* algorithm disseminates angle ranges to restrict the directions and utilizes them to locate whether two nodes have a line of sight or not. In addition, the angle range feature is employed to determine the best local parent on the path. Local parents are saved for additional utilization and are for further alteration of the path due to the fact that, on each new unexpanded visible neighbor node is carried out the line-of-sight checking. Furthermore, Phi* algorithm preserves two further values for every node, which are called lb-lower and ub-upper angle bounds, where together they form the angle range of the node [lb,ub]. Moreover, it comes up with a good tradeoff to the runtime of the search and the length of the resulting path. Also, it can find paths one order of magnitude faster than repeated Basic Theta* searches while finding paths with nearly the same lengths. The algorithm is characterized by its suitability for known and unknown environments, especially with its incremental version. It is correct, complete, and faster than the Basic Theta* algorithm.

The common and key elements associated with the three aforementioned pathfinding algorithms when they search for a path are the cost function, open list, closed list, and the line of sight. The open list is sometimes known as the open table that stores the nodes that have not been tested yet, whereas, the closed list is sometimes known as the closed table that stores the nodes that have been tested or explored. The cost function ($f_{Any-angle}$) is the evaluation function of the shortest path from the start node, across the current node that the algorithm on, up to the goal node. The line of sight is a smart mechanism that inspects if there is a free straight line between two nodes that are not adjacent. This technique has been utilized to loosen the restriction over the resulting path to be formed by grid edges by the standard algorithms such as A* algorithm. This is because the standard algorithms only allow the motion between neighbor nodes. For a more inclusive demonstration of the algorithms, interested readers could check the following references [34–37,47].

2.3. Cost Function

The cost function ($f_{Any-angle}$) is the current evaluation of the shortest path from the start node, across the current node that the algorithm on, up to the goal node. Similarly to what was implemented with A* algorithm in [48], a modified cost function has been employed within the scope of this work, which targets the flying vehicles by considering the elevation of the grid map nodes. The starting point P_{start} and the goal point P_{goal} will be connected through defining a set of waypoints by the three pathfinding algorithms. These waypoints are selected among the available nodes of a discretized environment. Initiating from P_{start} , the algorithms generate a group of paths that will be expanded more and more as the algorithms continue their search for a suitable path to P_{goal} , until one of these paths will be adopted as the final path. The aforementioned algorithms follow two paths which are path1 and path2 or they are known as a constrained path and unconstrained path to connect two points, where if there is a free straight line between two evaluated nodes, the algorithms will connect them, otherwise, they traverse across an adjacent node similar to A* algorithm search.

The evaluation function ($f_{Any-angle}$) can calculate the valuation of all the nodes, which can search in the next step, then choose the minimum to be the next node. So, this path is the best. Consequently, it can be calculated according to Equation (1).

$$f_{Any-angle} = k_g G + k_h H + k_e E \quad (1)$$

where G is the actual distance of moving from the start node to the current node on the map. Whereas, H is the valuation distance (heuristic estimated cost) from the current node to the target. The pathfinding algorithms in this paper depend on Euclidean distance

for calculating H , as shown in Equation (2). Also, E represents the cost of the risk that is associated with the elevation of the node. The parameters k_g, k_h, k_e represent the weights to tune the characteristics of the path and allow it to reach the goal point. The set of the parameters $k_g = 0.6, k_h = 1, k_e = 1.8$ through the implementation of the algorithms is observed to guarantee a proper compromise between the numbers of the extended nodes.

$$H = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \quad (2)$$

where $x(x_i, y_i)$ and $y(x_j, y_j)$ represent two points in a 2D space.

3. Trajectory Design

Trajectory planning is the process of finding a smooth and continuous curve to move along the path. Therefore, the trajectory is a path parameterized by time. Designing a trajectory usually indicates to the matter of getting the solution from the algorithms of pathfinding and defining how to navigate along the path. First, this section explains Bézier curves and then discusses the limitation of UAVs flying in sharp turns by presenting several Bézier curves-based approaches to solve this matter.

3.1. Bezier Curves

In 1962, the French engineer Pierre Bézier discovered a method of defining curves by control polygons. Based on the principle of approximation, this method constructs a set of special polynomial basis functions, which can effectively describe mathematical curves or sketches drawn by designers. Therefore, Bézier curves are parametric polynomial curves defined by the control points that are considerably employed in computer graphics to create smooth curves. The most frequent examples of Bezier curves applied are linear, quadratic, and cubic, as shown in Figure 3, in order to avoid complex computation and derivation of the high order curves.

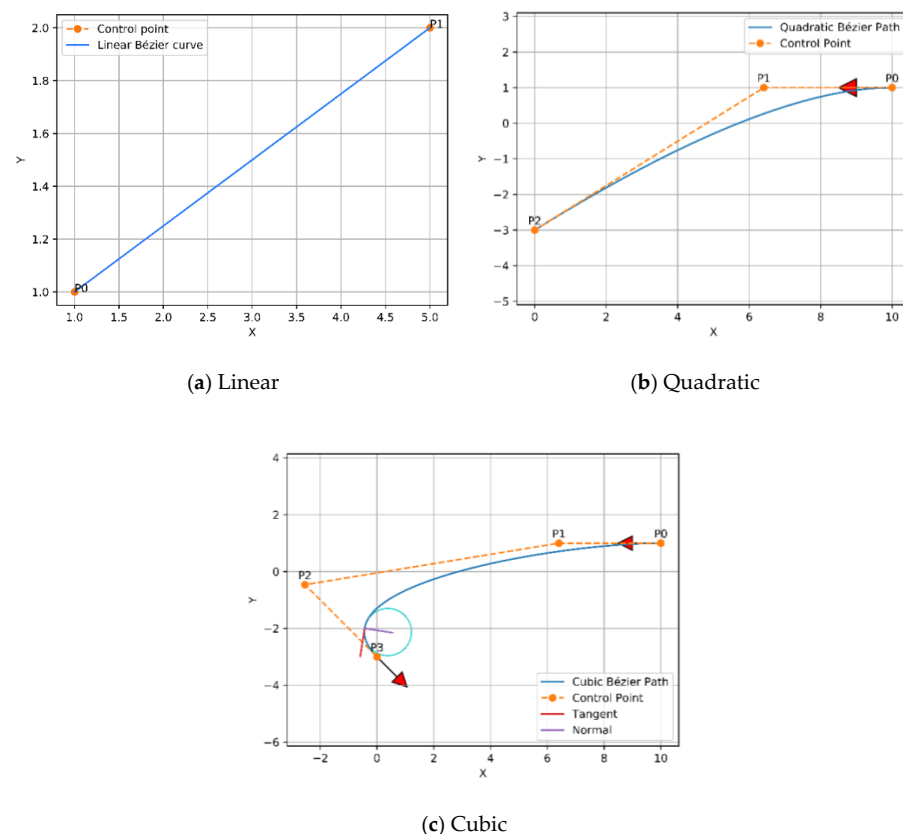


Figure 3. Examples of Bézier curves.

As previously mentioned, the definition of Bézier curve strictly depends on the number of control points of the curve. $n + 1$ can define the curve of degree n polynomial. The parameter equation of each point on Bézier curve is as follows:

$$P(t) = \sum_{i=0}^n P_i B_{i,n}(t), \quad t \in [0, 1] \quad (3)$$

where P_i are the control points and $B_{i,n}$ is the Bernstein basis function, and its polynomial expression is as follows:

$$B_{i,n}(t) = C(n, i) t^i (1 - t)^{n-i}, \quad i = 0, 1, 2, \dots, n \quad (4)$$

where $C(n, i) = \frac{n!}{i!(n-i)!}$, by substituting in Equation (3), the Bernstein basis function can be rewritten as follows:

$$B_{i,n}(t) = \frac{n!}{i!(n-i)!} t^i (1 - t)^{n-i} \quad (5)$$

The curves of Bézier always pass through the endpoints, which are the first and last control points $P(t = 0) = P_0$, $P(t = n) = P_n$, whereas, the lines that connect P_0 to P_1 and P_{n-1} to P_n are tangent to these curves [44]. To design the trajectory, the waypoints of the three modified any-angles pathfinding algorithms paths are endorsed as control points of the curve. Since the environment is filled with obstacles, a safer flight of UAV is achieved by finding a midpoint between each of the two control points on each segment of the constructed path, and the waypoints with the midpoints will be connected by curves. As a result, the constructed trajectory will be constituted of several joined curves. Linear and quadratic Bézier curves were employed to design the trajectory, which are shown in Figure 3 and described in Equations (6) and (7), respectively.

$$P(t) = (1 - t)P_0 + tP_1 \quad (6)$$

$$P(t) = (1 - t)^2 P_0 + 2t(1 - t)P_1 + t^2 P_2 \quad (7)$$

3.2. Proposed Approach

As previously mentioned in Section 3.1, the most frequent examples of Bézier curves have been utilized (linear, quadratic, and cubic curves) to avoid the monotonous and exhausting computation processes. In the same section, it is also mentioned that constructing these curves strictly depends on the control points, which are the waypoints of the generated paths. It is evident that the constructed paths from the aforementioned pathfinding algorithms will contain numerous waypoints, and therefore there will be many control points that require massive mathematical operations. In fact, the curves of Bézier guarantee a significant degree of resiliency and the cost of calculation is insignificant, as they are based on polynomials that were created utilizing the appropriate control points. If the degree of the curves has risen, the previous characteristics commence disappearing when the control points number is large. A proper solution has been introduced by finding a midpoint between each two control points on each segment of the constructed path. The endpoints and control points of the curve will be connected with these midpoints by adopting quadratic Bézier curves. As a result, the final trajectory will be constituted of several joined curves. This method will guarantee an uncomplicated and efficient way to draw a curve for an innumerable number of control points. This method is achieved by using an array of points, where the points are inserted in this array and then passed to the function, thus intense and complicated mathematical operations will be avoided.

For example, Bézier curve has the following points $[P_0, P_1, P_2, P_3, P_4, P_5, P_6]$, where P_0 and P_6 represent the endpoints. Whereas, P_1 and P_2 represent the first two control points, while P_2 and P_3 represent the second two control points and so on. To apply the above solution, a first midpoint is called M_0 and is placed between the first two control points to create $[P_1, M_0, P_2]$. Next, a second midpoint M_1 is placed between the second two control

points to form $[P_2, M_1, P_3]$, and so on. Subsequently, from the first endpoint P_0 towards the first midpoint M_0 , the first quadratic Bézier curve will be formed. After that, from M_0 to M_1 , the second quadratic Bézier curve will be created as well. The two previous quadratic Bézier curves will force C^2 derivative continuity as a result of these points $[P_1, M_0, P_2]$ being in perfect alignment. The first and second derivatives of Bézier curves according to t are presented in Equations (8) and (9). Finally, this way will continuously follow the same previous approach until arriving at the final endpoint. Thus, from the first endpoint to the final endpoint, a smooth curve was constructed, as shown in Figure 4.

$$\dot{P}(t) = \sum_{i=0}^{n-1} (P_{i+1} - P_i)nB_{n-1,i}(t), \tag{8}$$

$$\ddot{P}(t) = \sum_{i=0}^{n-2} (P_{i+2} - 2P_{i+1} + P_i)n(n-1)B_{n-2,i}(t). \tag{9}$$

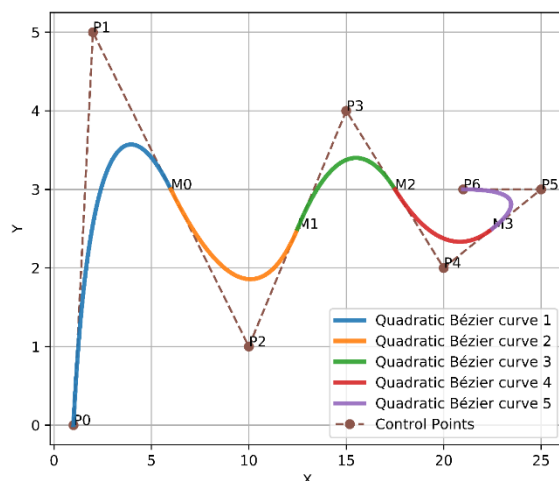


Figure 4. Bézier curves approach 1.

A new approach was presented by placing a couple of midpoints between each two control points on each segment of the generated path. This will add more control to the generated trajectory, as the midpoints were placed in $[0.25, 0.75]$ positions, as shown in Figure 5.

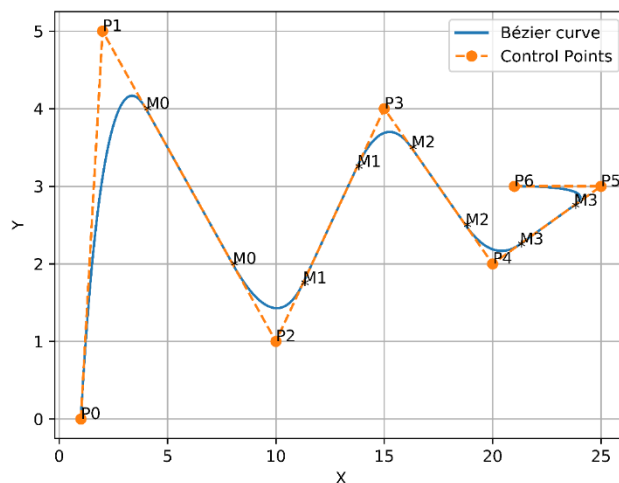


Figure 5. Bézier curves approach 2.

To guarantee more safety to the UAV on the end extremities of the trajectory, an additional single midpoint was placed between the endpoints and control points, as shown in Figure 6. As previously mentioned in Section 2.1, the obstacles are randomly located in the environment, that is, there is no full control on how they will behave or where they will be on the next execution to the algorithm. This may present the possibility of some obstacles close to the resulting trajectory at the final points, which could add danger to the flying vehicle. Therefore, adding a midpoint on the final extremities means adding more control to the trajectory by pushing it towards the path. In this approach, linear Bézier curves were also exploited to connect the final endpoints of the curve with the midpoints. This is because of the final extremities points of the trajectory's just two points, which are $[P_0, M_0]$ and $[M_5, P_6]$, and it is inadmissible to utilize quadratic Bézier curves to represent them.

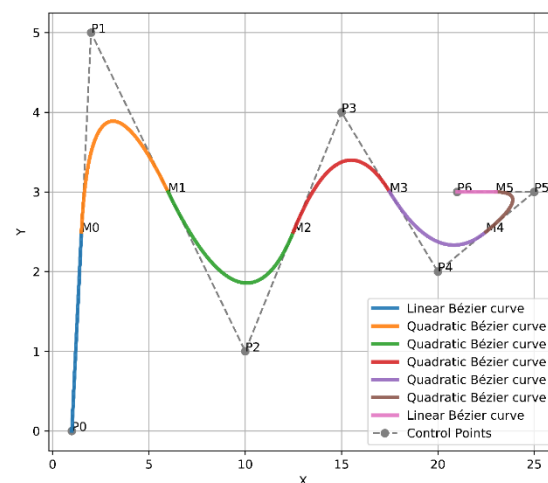


Figure 6. Bézier curves approach 3.

Combining the idea of the second and third approaches, a final and new suggestion has been presented by inserting two midpoints over the whole segments of the generated path, since they were placed at the same positions of the second approach $[0.25, 0.75]$ positions of each segment of the path, as shown in Figure 7. The addition of more points should take into account the characteristics of the generated trajectory as ensuring short, small curvature, and flyable trajectories, when it pulled out towards the path for generating more safety to the aircraft. Figure 8 clarifies the four Bézier curves-based approaches.

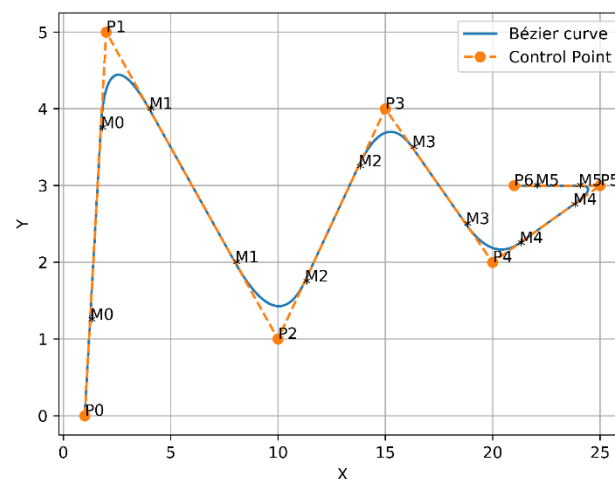


Figure 7. Bézier curves approach 4.

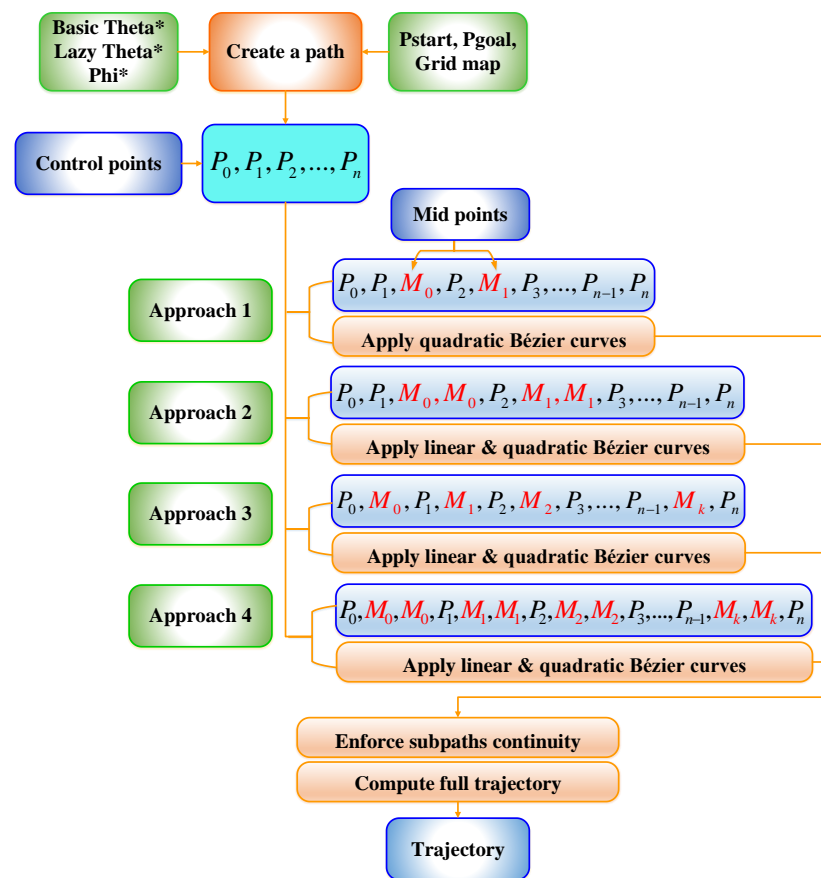


Figure 8. Bézier curves-based approaches.

4. Results and Discussion

This section displays the results of three modified any-angle pathfinding algorithms (Basic Theta*, Lazy Theta*, and Phi*) implementation and the aforementioned four Bézier curves-based approaches over the resulting paths from the pathfinding algorithms. Additionally, it presents an innovative evaluation process of these four approaches with its results to select the near-optimum one among them in designing a flyable and collision-free trajectory with the safety feature. Simultaneously, it illustrates the re-evaluation process of the above method with its results without the safety feature. Finally, this section is concluded by presenting a new algorithm with its results that pushes the selected approach to the optimality in constructing the trajectory.

4.1. Implementation of the Modified Any-Angle Pathfinding Algorithms

The unblocked paths of the modified any-angle pathfinding algorithms, which are Basic Theta*, Lazy Theta*, and Phi*, have been displayed with red color in this section, as shown in Figure 9. The positions of the start and goal points have been located at (5,5) coordinates with a green color and (45,45) coordinates with a blue color within a grid size of 50×50 nodes. For more clarification, the color of the nodes indicates that their elevation less than 50 is changed from one algorithm to the other.

4.2. Performance Comparison of the Three Any-Angle Pathfinding Algorithms

A comparison has been made among the aforementioned any-angle pathfinding algorithms in terms of runtime, where it is intended to know how long each algorithm takes while searching for an unblocked path between the starting and goal points. Time plays an important role in pathfinding algorithms' work and the faster the algorithm is, the more appropriate it is to be adopted, especially in larger environments and particularly in some

current applications that depend considerably on pathfinding algorithms. The runtime is calculated by using Python programming code for each algorithm where the code was executed 100 times and the average value has been taken. All the algorithms were applied to a 2D environment with two different grid sizes (size 1 = 50×50 and size 2 = 100×100). Furthermore, the start and goal points have been set to $(x_{start} = 5, y_{start} = 5)$, $(x_{goal} = 45, y_{goal} = 45)$ for a grid size 50×50 and $(x_{start} = 5, y_{start} = 5)$, $(x_{goal} = 95, y_{goal} = 95)$ for a grid size 100×100 . Table 1 displays the elapsed time that has been taken by each algorithm to find the path on the environment with two different grid sizes.

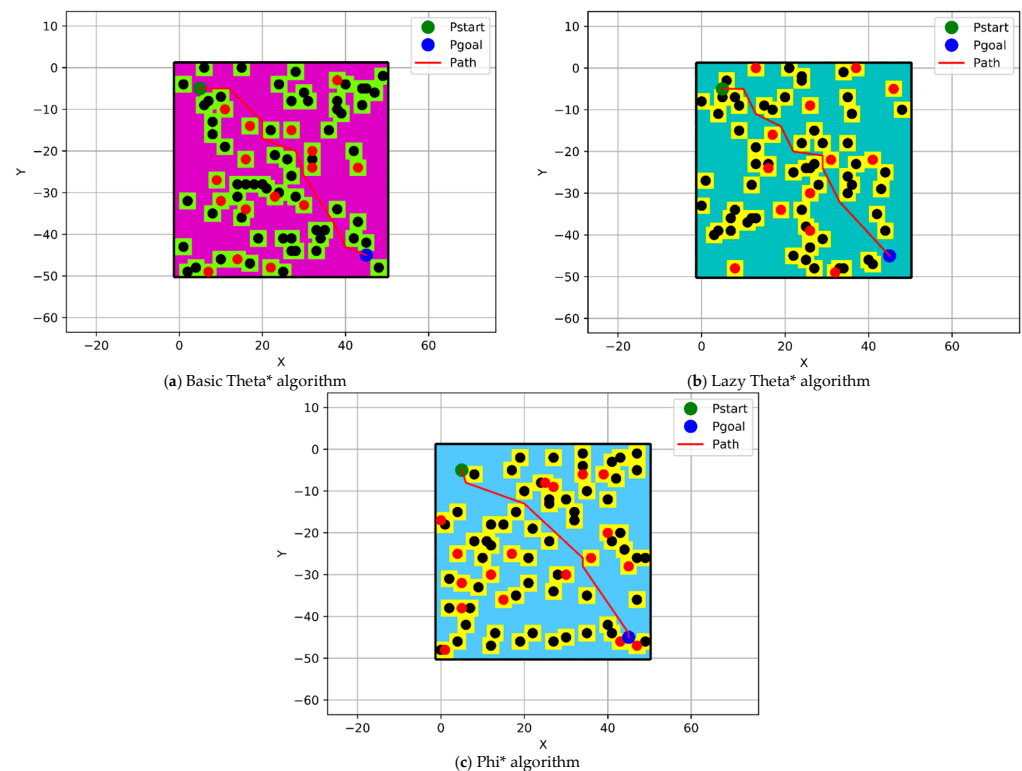


Figure 9. Modified any-angle pathfinding algorithms paths.

Table 1. Performance comparison of any-angle pathfinding algorithms.

	Grid Size	Basic Theta*	Lazy Theta*	Phi*
Time [Sec]	50×50	0.08818	0.03178	0.03403
Time [Sec]	100×100	0.1939	0.0603	0.06112

It is obvious by observing the runtime of each algorithm to find a path from the initial node to the goal node based on varying the environment size; it can be observed that Basic Theta* algorithm is the slowest algorithm between the other two implemented algorithms, whereas the Lazy Theta* and Phi* algorithms have shown convergence in the runtime for finding the path, and they both are faster than the Basic Theta* algorithm.

4.3. Implementation of Bézier Curves-Based Approaches on the Pathfinding Algorithms

In this section, a continuous flyable trajectory for a multirotor UAV has been constructed using the four aforementioned Bézier curves-based approaches with the three modified any-angle pathfinding algorithms. The characteristics of the environment are well explained in Section 2.1. The positions of the start and goal points will be changed for each algorithm and the design of the trajectory will be located within a grid size of 50×50 nodes. The four proposed Bézier curves-based approaches are demonstrated as follow:

1. A single midpoint has been located at the center of each segment of the generated path excluding the first and the final segments. This approach was named the Bézier1 curve approach.
2. Two midpoints have been located at [0.25, 0.75] positions of each segment of the generated path excluding the first and the final segments. This approach was named the Bézier2 curve approach.
3. A single midpoint has been located at the center of each segment of the generated path including the first and the final segments. This approach was named the Bézier3 curve approach.
4. Two midpoints have been located at [0.25, 0.75] positions of each segment of the generated path including the first and the final segments. This approach was named the Bézier4 curve approach.

4.3.1. Basic Theta* Algorithm

This section displays the design of the flyable trajectories with Basic Theta* algorithm using the aforementioned four approaches, as shown in Figure 10. The positions of the start and goal points have been located at (5,5) coordinates with a green color and (45,45) coordinates with a blue color within a grid size of 50×50 nodes.

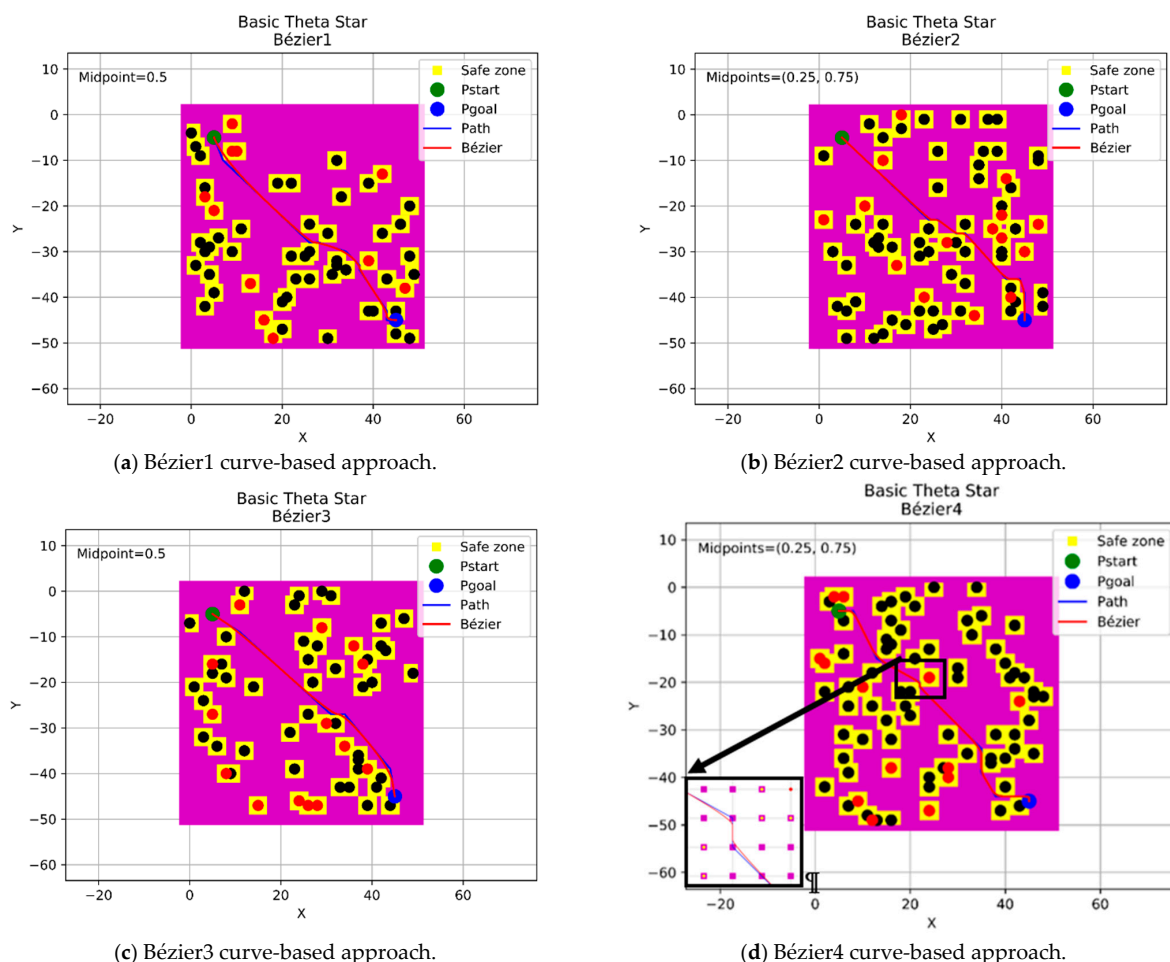


Figure 10. The trajectories of the four Bézier curves-based approaches with Basic Theta*.

4.3.2. Lazy Theta* Algorithm

This section displays the design of the flyable trajectories with the Lazy Theta* algorithm using the four Bézier curves-based approaches, as shown in Figure 11. The positions

of the start and goal points have been located at (5,45) coordinates with a green color and (45,5) coordinates with a blue color within a grid size of 50×50 nodes.

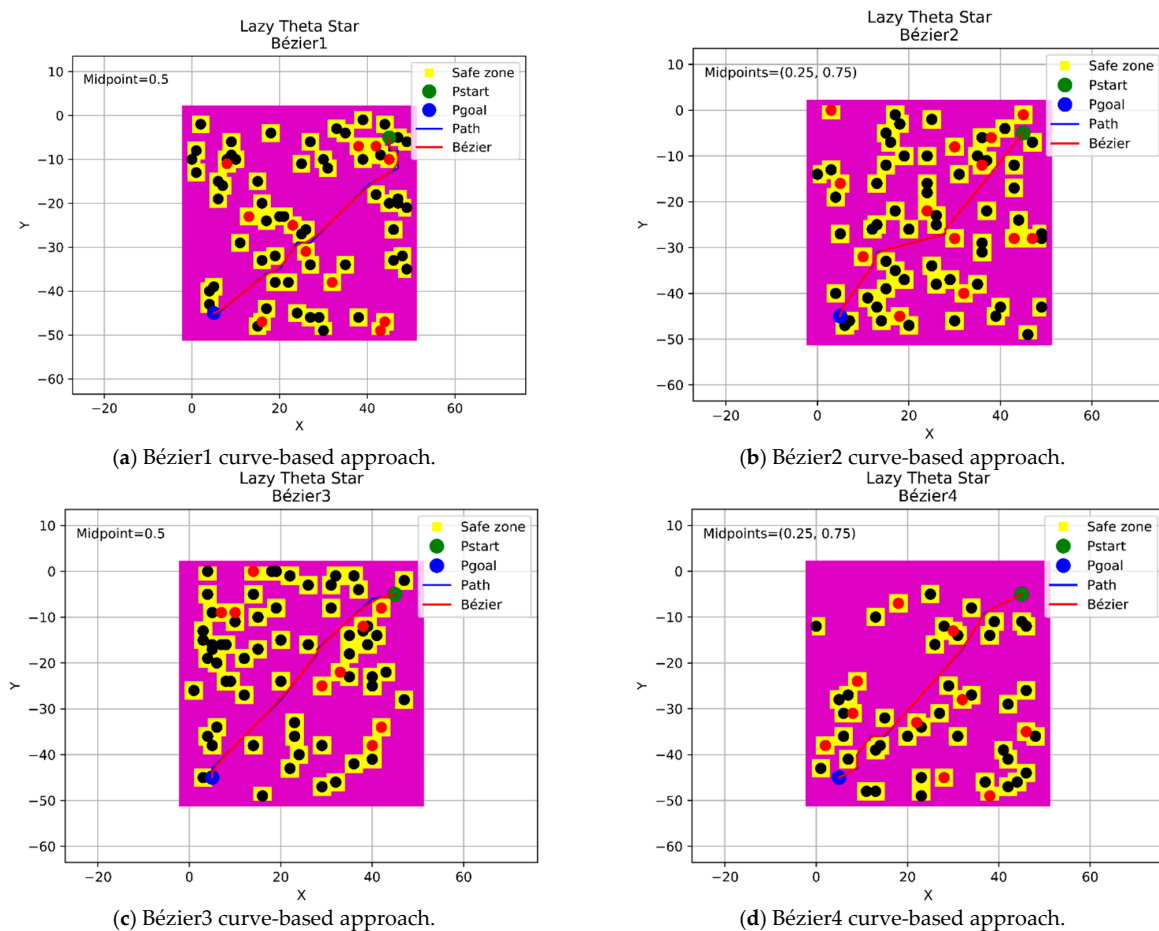


Figure 11. The trajectories of the four Bézier curves-based approaches with Lazy Theta*.

4.3.3. Phi* Algorithm

This section displays the design of the flyable trajectories with the Phi* algorithm using the four Bézier curves-based approaches, as shown in Figure 12. The positions of the start and goal points have been located at (45,5) coordinates with a green color and (5,45) coordinates with a blue color within a grid size of 50×50 nodes.

It is obvious from the above Figures 10–12 that the four approaches of Bézier curves have conducted their job in creating a flyable and smooth trajectory for the aircraft when they were applied to the generated paths of the three modified any-angle pathfinding algorithms. Simultaneously, some questions may arise. How safe and successful are these trajectories? Are they safe enough to be adopted? Are all these approaches applicable? These questions will be answered in the next sections by presenting a novel and innovative evaluation processes to validate the integrity and applicability of those approaches, additionally, specifying the near-optimal and the most suitable approach to be adopted in designing the trajectory.

4.4. Evaluation Process of the Proposed Bézier Curves-Based Approaches with the Safety Feature

This section answers the questions that have been asked at the end of the previous section in detail by introducing a new and innovative mechanism to evaluate the four approaches, which were studied in Section 3.2. This process aims to find a proper approach that guarantees the construction of a flyable and smooth trajectory from among the four Bézier curves-based approaches that have been proposed. The four approaches were

evaluated by applying them to a variety of different size grid maps; the more environments there are, the greater the confidence in knowing whether one of these approaches is effective to be adopted. Consequently, 1000 randomly generated environments were employed to evaluate the four approaches, which are Bézier1, Bézier2, Bézier3, and Bézier4. Additionally, one of the three any-angle pathfinding algorithms in this paper was utilized in each environment beside the four Bézier curves-based approaches with randomly generated obstacles and start and goal points. Due to the research work being scarce on the Phi* algorithm and the fact that it has a lot of room to explore, it will be directly adopted in the evaluation process. After that, a validation mechanism was constructed for each operation that takes place in each environment, which includes (path and trajectory design) examining their rate of success. After generating the path and the four trajectories on each environment, this mechanism checks each point along each constructed trajectory with a rounding process.

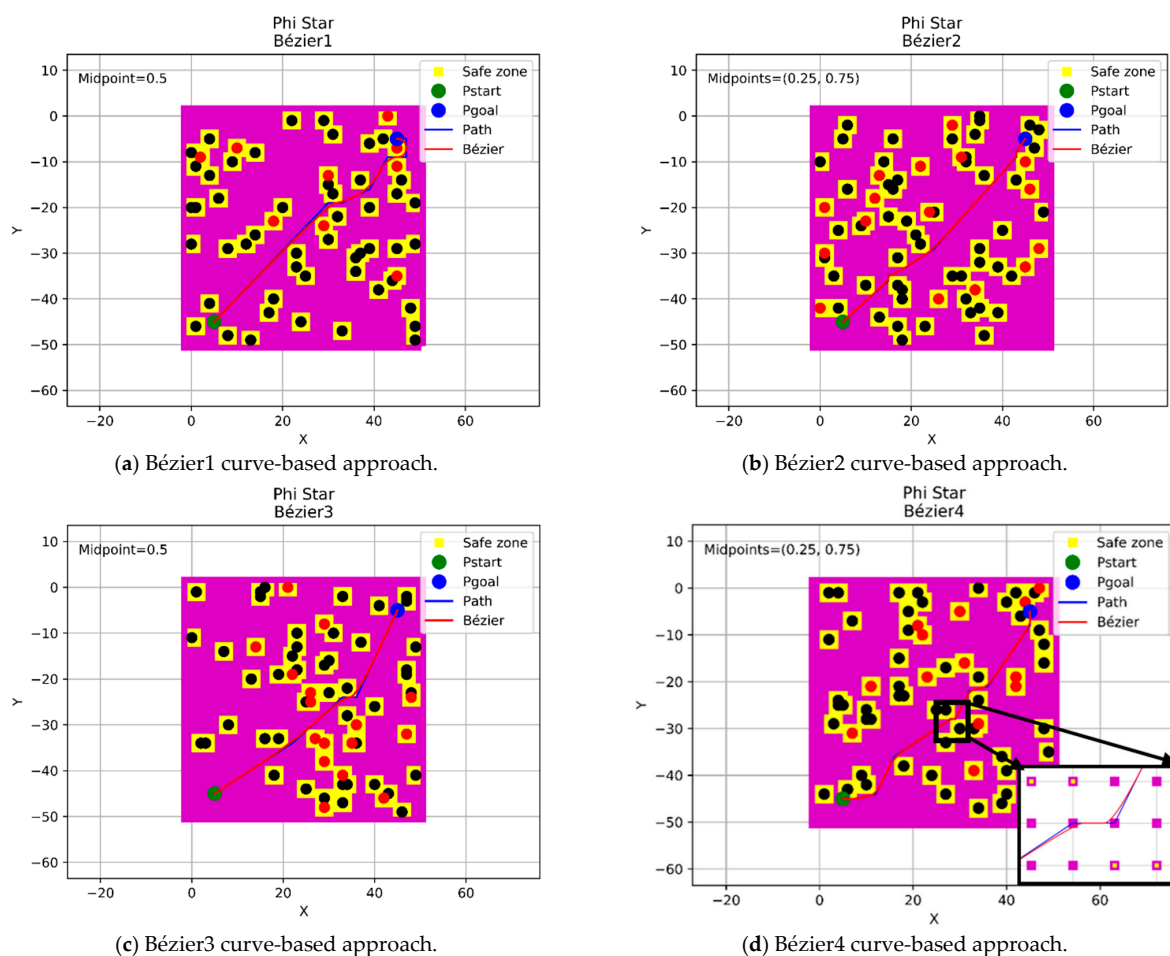


Figure 12. The trajectories of the four Bézier curves-based approaches with Phi*.

For example, for a trajectory that crosses a specific point that existed in the free space between the generated path from the modified Phi* algorithm and the safety feature, which is the yellow buffer area that surrounds the obstacles, if this point exists on a 7.3 position, it is rounded to be 7, this will be considered as an inappropriate point. Whereas, if the location of this point is located on a 7.7 position, it is rounded to be 8; this will be considered a suitable point and so on. The distance between the safety feature and the path is one of the grid environment coordinates, as shown in Figure 13. The points of Bézier curves lie within the grid map, so the validator examines their positions within the grid and rounds their coordinates to check if they are close to the buffer area or the path. It may happen that a point on the trajectory does not cross the buffer area and at the same time is considered

inadmissible due to the rounding process, while in reality, it could be a good point. The above approach is considered conservative. For this reason, the success rate was not found to be 100%, but it does not mean that the results are unsatisfactory because it still shows very high results of success rates of up to 89.9%. This evaluation process is only to confirm that these approaches are generating safe and smooth trajectories and even far from the safety feature. It is not about proving that Bézier curves are proper to be adopted or not for trajectory design, since all the aforementioned approaches had the ability to create flyable trajectories.

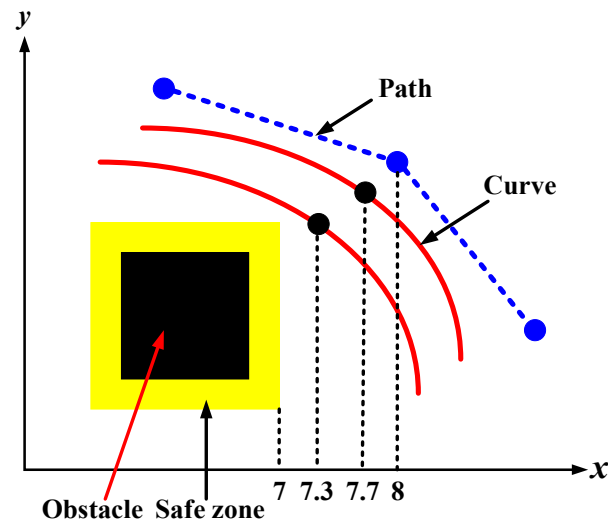


Figure 13. The rounding process of Bézier curve points.

The evaluation process is performed by creating a 4×1000 matrix. Each row of this matrix represents one of the Bézier curves-based approaches, while each column will include the evaluation result (success rate) for each whole operation on each environment. As a result, if the rounding process has proven that the whole constructed trajectory from the starting point to the goal point is far from the safety feature, set 1 in the corresponding matrix position, otherwise 0.

Finally, the success rate of each approach is calculated based on the number of ones in each row. The higher the number of ones within the row, the more appropriate this approach is to be adopted in creating the trajectory. The evaluation process has been implemented on a machine, which has the configurations as in Table 2. The evaluation process of the four Bézier curves-based approaches is illustrated in Figure 14.

Table 2. The properties of the test platform.

Properties	Values
Computer	LEGION—Lenovo
CPU	Intel(R) Core(TM) i7-8750h 2.20 GHz
RAM	8.00 GB
Operating System	64 bit Windows 10 Pro edition
Programming Language	Python
IDE	JetBrains PyCharm Community Edition 2019.1.1 x64

Looking at Figure 15 that illustrates the results of the evaluation process, which were applied to three different sizes of environments (30×30 , 50×50 , 100×100), it is obvious that the first approach takes the least amount of success rates, whereas, the fourth approach imposed its dominance in all cases among the others. This explains that the Bézier4 curve-based approach is a suitable approach to be adopted. It was also seen, with the increase in the size of the environments, that the success rates are decreasing, due to the fact that the

generated trajectory will be long, which explains that it will encounter more obstacles, and therefore there is a higher probability to pass close to or inside the safety feature area. As a result, this evaluation process will be repeated in the next section without considering the safety feature just to guarantee the extent of the integrity of the generated trajectory.

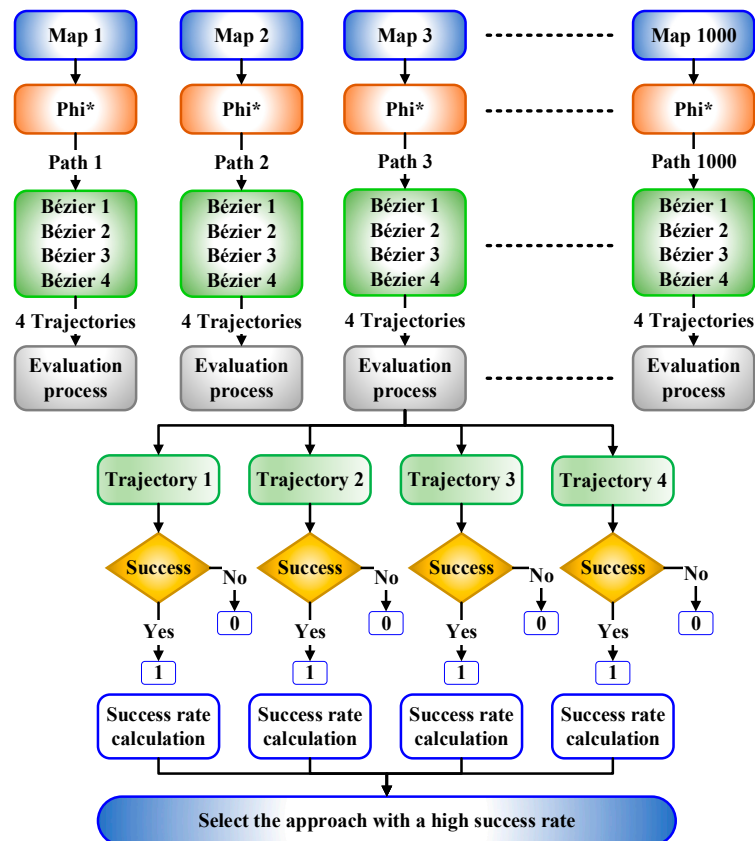


Figure 14. The evaluation process of Bézier curves-based approaches.

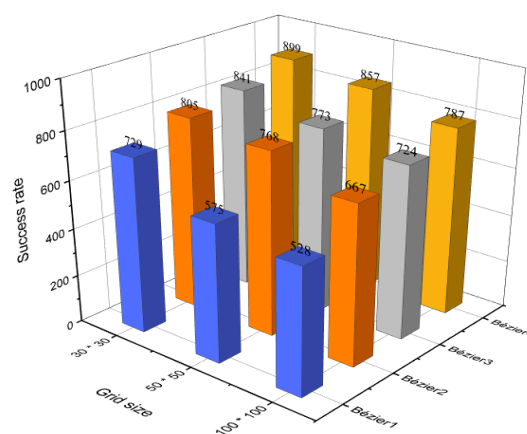


Figure 15. The success rate of Bézier curves-based approaches with the safety feature.

4.5. Re-Evaluation Process of the Proposed Bézier Curves-Based Approaches without the Safety Feature

To investigate the possibility of generating collision-free trajectories, the aforementioned evaluation process of the four Bézier curves-based approaches in Section 4.4 was repeated with the same previous conditions by examining the points on Bézier curves without considering the surrounded buffer area. From Figure 16, it is clear that the Bézier4

curve approach demonstrated again its dominance with a success rate of 100% in the three different sizes of the environment. Whereas, Bézier1, Bézier2, and Bézier3 curves approaches also accentuated high success rates and they decreased gradually with the increase of the environment size due to the length of the generated trajectories and the large number of obstacles, which may cause an increase in the probability of trajectory colliding with obstacles due to the rounding process.

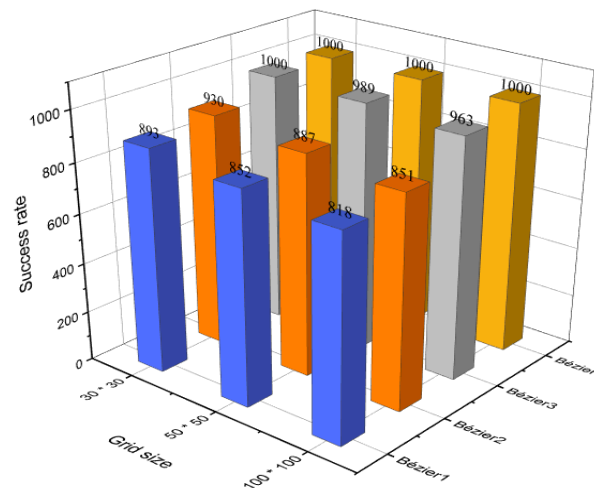


Figure 16. The success rate of Bézier curves-based approaches without the buffer area.

4.6. A New Approach to Increasing the Efficiency of Bézier4 Curve-Based Approach to the Optimality

After the evaluation process was conducted on the four Bézier curves-based approaches, it was found that the fourth approach is the most suitable to be adopted in designing the trajectory compared to the three other approaches, since it showed high success rates, as shown in Figures 15 and 16. To reach the optimality in generating a safe and smooth trajectory, a new algorithm has been proposed to work with the Bézier4 curve approach. This algorithm suggests adding further midpoints, which guarantees to increase the safety of the generated trajectory with a very high success rate and generate an optimal trajectory far from the yellow buffer area. To illustrate the work of the proposed algorithm, a schematic diagram of the algorithm is presented in Figure 17.

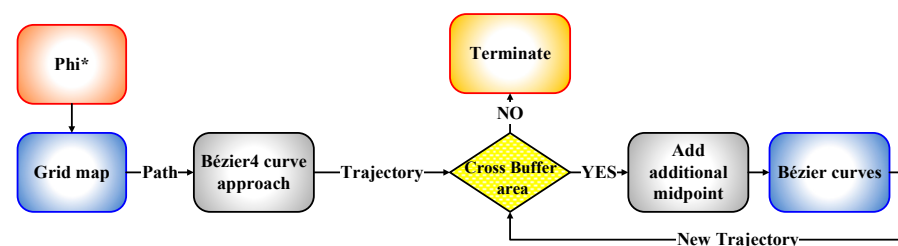


Figure 17. The schematic diagram of the new proposed algorithm.

From the process described above, the proposed algorithm showed its success in generating an optimal trajectory from a specific starting point to a specific goal point, as shown in Figure 18. The intended near-optimal and/or optimal criterion means the guarantee of generating a flyable path from the specific starting point to the specific goal point without colliding with the obstacles that have random and uncontrolled behavior, in terms of generation, in addition to ensuring that the safety feature that was constructed around the obstacles is not bypassed by the UAV. Therefore, the amount of this optimality was determined based on the success rates that were shown in Sections 4.4–4.6 which are up to 100%. The added midpoints to the generated path assured to drag the generated

trajectory towards the path and keep it far from the buffer area. In addition to the previous two midpoints that are already within the fourth approach of the Bézier curve, another two midpoints have been added according to the new algorithm. Therefore, the algorithm has been iterated twice by adding a midpoint at each iteration, so that the sum of the added midpoints is four, which are distributed equally in each segment of the path in order [0.20, 0.40, 0.60, 0.80] and are illustrated with black asterisks in Figure 18. Subsequently, the evaluation process has been applied for the generated trajectory with the Bézier4 curve approach; as a result, the success rate has reached up to 100% under the three different sizes of the environment, as shown in Figure 19.

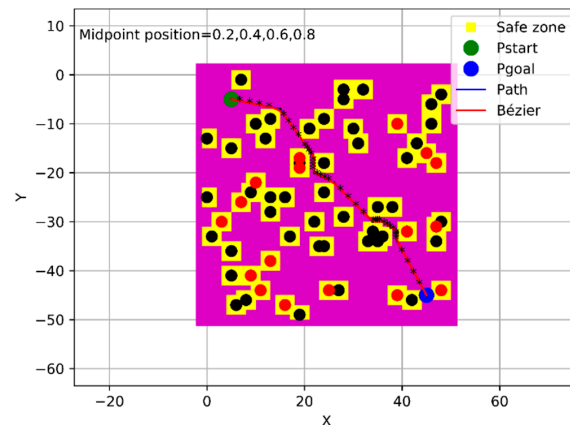


Figure 18. Phi* path with the Bézier4 curve approach within the new algorithm.

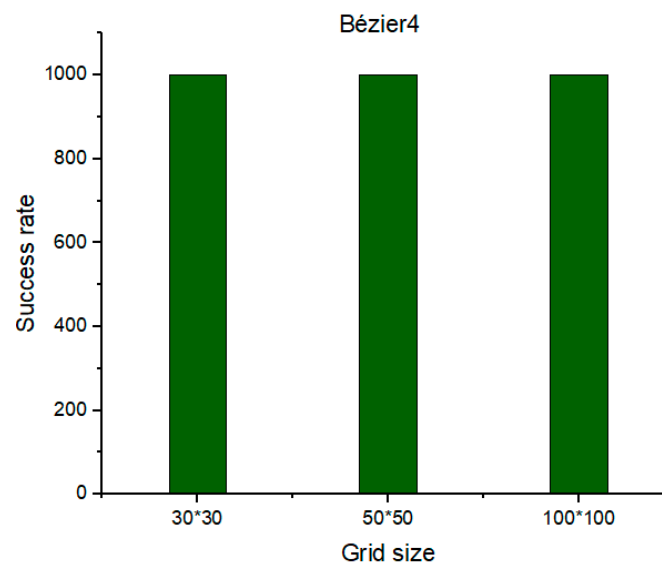


Figure 19. The success rate of the Bézier4 curve approach over Phi* algorithm.

5. Conclusions

This paper debates how to construct an optimal and flyable trajectory for a multirotor UAV in a cluttered environment. Mainly, three algorithms of any-angle pathfinding algorithms (Basic Theta*, Lazy Theta*, and Phi*) were employed for solving the problem of path planning. The conclusion is drawn as follow:

1. Feasible paths were calculated by employing three modified algorithms of any-angle pathfinding algorithms over a conventional grid.
2. The resulting paths were characterized by angular or sharp turns that are inappropriate for the flying vehicles; therefore, the properties of Bézier curves were exploited for smoothing the path by presenting four Bézier curves-based approaches.

3. An innovative evaluation process was proposed to determine the most proper approach of the four Bézier curves-based approaches in constructing a flyable and collision-free trajectory.
4. Based on the evaluation process results, a novel algorithm was proposed to push the chosen Bézier curve-based approach to be optimal by adding further midpoints. As a result, it has proven its effectiveness by generating an optimal trajectory with a success rate of up to 100%.

In future work, the characteristics of the 3D environment are planned to be considered for application with multirotor UAVs.

Author Contributions: For research articles with several authors, a short paragraph specifying their individual contributions must be provided. The following statements should be used “Conceptualization, H.A.S.; methodology, H.A.S.; M.M.A.Z.; software, Z.I.R.; R.S.A.-A.; validation, H.A.S., M.M.A.Z. and Z.I.R.; formal analysis, Z.I.R., M.M.A.Z.; investigation, H.A.S., R.S.A.-A.; resources, Z.I.R., R.S.A.-A.; data curation, M.M.A.Z., C.I.P.; writing—original draft preparation, H.A.S., R.S.A.-A.; writing—review and editing, M.M.A.Z., Z.I.R., C.I.P.; visualization, Z.I.R., C.I.P.; supervision, H.A.S., R.S.A.-A.; project administration, H.A.S., C.I.P.; funding acquisition, M.M.A.Z., Z.I.R., C.I.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Nature Science Foundation of China (NSFC) under grant number 61873163.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: We acknowledge that the under Grant has supported this work. Additionally, we appreciate the great assistance and support of Raneen Abd Ali, Angga Setyawan, and the Institute of Sensing and Navigation at Shanghai Jiao Tong University. We would also like to give our gratitude to the editor of the journal and the reviewers for their constructive and encouraging suggestions and comments on the research.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Gupta, S.G.; Ghonge, M.M.; Jawandhiya, P.M. Review of unmanned aircraft system (UAS). *Int. J. Adv. Res. Comput. Eng. Technol.* **2013**, *2*, 1646–1658. [\[CrossRef\]](#)
2. Hu, C.; Zhang, Z.; Yang, N.; Shin, H.S.; Tsourdos, A. Fuzzy multiobjective cooperative surveillance of multiple UAVs based on distributed predictive control for unknown ground moving target in urban environment. *Aerosp. Sci. Technol.* **2019**, *84*, 329–338. [\[CrossRef\]](#)
3. Albeaino, G.; Gheisari, M.; Franz, B.W. A systematic review of unmanned aerial vehicle application areas and technologies in the AEC domain. *J. Inf. Technol. Constr.* **2019**, *24*, 381–405.
4. Hildmann, H.; Kovacs, E. Using unmanned aerial vehicles (UAVS) as mobile sensing platforms (MSPS) for disaster response, civil security and public safety. *Drones* **2019**, *3*, 59. [\[CrossRef\]](#)
5. Shirani, B.; Najafi, M.; Izadi, I. Cooperative load transportation using multiple UAVs. *Aerosp. Sci. Technol.* **2019**, *84*, 158–169. [\[CrossRef\]](#)
6. Li, P.; Duan, H. A potential game approach to multiple UAV cooperative search and surveillance. *Aerosp. Sci. Technol.* **2017**, *68*, 403–415. [\[CrossRef\]](#)
7. Claesson, A.; Fredman, D.; Svensson, L.; Ringh, M.; Hollenberg, J.; Nordberg, P.; Ban, Y. Unmanned aerial vehicles (drones) in out-of-hospital-cardiac-arrest. *Scand. J. Trauma. Resusc. Emerg. Med.* **2016**, *24*, 1–9. [\[CrossRef\]](#)
8. Chiaraviglio, L.; D’Andreagiovanni, F.; Choo, R.; Cuomo, F.; Colonnese, S. Joint optimization of area throughput and grid-connected microgeneration in UAV-based mobile networks. *IEEE Access.* **2019**, *7*, 69545–69558. [\[CrossRef\]](#)
9. Chiaraviglio, L.; D’Andreagiovanni, F.; Liu, W.; Gutierrez, J.; Blefari-Melazzi, N.; Choo, K.K.; Alouini, M.S. Multi-Area Throughput and Energy Optimization of UAV-aided Cellular Networks Powered by Solar Panels and Grid. *IEEE Trans. Mob. Comput.* **2020**. [\[CrossRef\]](#)
10. Di Franco, C.; Buttazzo, G. Energy-aware coverage path planning of UAVs. In Proceedings of the 2015 IEEE International Conference on Autonomous Robot Systems and Competitions, Vila Real, Portugal, 8–10 April 2015; pp. 111–117.
11. Ji, X.; Meng, X.; Wang, A.; Hua, Q.; Wang, F.; Chen, R.; Zhang, J.; Fang, D. E2PP: An Energy-Efficient Path Planning Method for UAV-Assisted Data Collection. *Secur. Commun. Netw.* **2020**, *2020*, 8850505. [\[CrossRef\]](#)

12. Szczerba, R.J.; Galkowski, P.; Glicktein, I.S.; Ternullo, N. Robust algorithm for real-time route planning. *IEEE Trans. Aerosp. Electron. Syst.* **2000**, *36*, 869–878. [[CrossRef](#)]
13. Zheng, C.; Li, L.; Xu, F.; Sun, F.; Ding, M. Evolutionary route planner for unmanned air vehicles. *IEEE Trans. Robot.* **2005**, *21*, 609–620. [[CrossRef](#)]
14. Mittal, S.; Deb, K. Three-dimensional offline path planning for UAVs using multiobjective evolutionary algorithms. In Proceedings of the 2007 IEEE Congress on Evolutionary Computation, Singapore, 25–28 September 2007.
15. Roberge, V.; Tarbouchi, M.; Labonté, G. Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning. *IEEE Trans. Industr. Inform.* **2012**, *9*, 132–141. [[CrossRef](#)]
16. Yang, K.; Sukkariéh, S. 3D smooth path planning for a UAV in cluttered natural environments. In Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, France, 22–26 September 2008.
17. Badea, A.C.; Badea, G. An Overview of Geoprocessing and Export Options for Creating 3D GIS Models Using Drone2Map. *RevCAD J. Geod. Cadastr.* **2020**, *28*, 7–14.
18. Kim, J.; Crassidis, J.L. UAV path planning for maximum visibility of ground targets in an urban area. In Proceedings of the 2010 13th International Conference on Information Fusion, Edinburgh, UK, 26–29 July 2010.
19. Yao, W.; Qi, N.; Wan, N.; Liu, Y. An iterative strategy for task assignment and path planning of distributed multiple unmanned aerial vehicles. *Aerosp. Sci. Technol.* **2019**, *86*, 455–464. [[CrossRef](#)]
20. Lee, J.Y.; Chung, A.Y.; Shim, H.; Joe, C.; Park, S.; Kim, H. UAV flight and landing guidance system for emergency situations. *Sensors* **2019**, *19*, 4468. [[CrossRef](#)]
21. Wu, W.; Wang, X.; Cui, N. Fast and coupled solution for cooperative mission planning of multiple heterogeneous unmanned aerial vehicles. *Aerosp. Sci. Technol.* **2018**, *79*, 131–144. [[CrossRef](#)]
22. Primatesta, S.; Rizzo, A.; la Cour-Harbo, A. Ground risk map for unmanned aircraft in urban environments. *J. Intell. Robot. Syst.* **2020**, *97*, 489–509. [[CrossRef](#)]
23. Qiu, Z.; Chu, X.; Calvo-Ramirez, C.; Briso, C.; Yin, X. Low altitude UAV air-to-ground channel measurement and modeling in semiurban environments. *Wirel. Commun. Mob. Comput.* **2017**, *2017*, 1587412. [[CrossRef](#)]
24. Zhou, H.; Xiong, H.L.; Liu, Y.; Tan, N.D.; Chen, L. Trajectory planning algorithm of UAV based on system positioning accuracy constraints. *Electronics* **2020**, *9*, 250. [[CrossRef](#)]
25. Yang, L.; Qi, J.; Xiao, J.; Yong, X. A literature review of UAV 3D path planning. In Proceedings of the 11th World Congress on Intelligent Control and Automation, Shenyang, China, 29 June–4 July 2014.
26. González, D.; Pérez, J.; Milanés, V.; Nashashibi, F. A review of motion planning techniques for automated vehicles. *IEEE Trans. Intell. Transp. Syst.* **2015**, *17*, 1135–1145. [[CrossRef](#)]
27. Zafar, M.N.; Mohanta, J.C. Methodology for path planning and optimization of mobile robots: A review. *Procedia Comput. Sci.* **2018**, *133*, 141–152. [[CrossRef](#)]
28. Gasparetto, A.; Boscarior, P.; Lanzutti, A.; Vidoni, R. Trajectory planning in robotics. *Math. Comput. Sci.* **2012**, *6*, 269–279. [[CrossRef](#)]
29. Yang, L.; Qi, J.; Song, D.; Xiao, J.; Han, J.; Xia, Y. Survey of robot 3D path planning algorithms. *J. Control Sci. Eng.* **2016**, *2016*, 7426913. [[CrossRef](#)]
30. Dijkstra, E.W. A note on two problems in connexion with graphs. *Numer. Math.* **1959**, *1*, 269–271. [[CrossRef](#)]
31. Hart, P.E.; Nilsson, N.J.; Raphael, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Syst. Sci. Cybern.* **1968**, *4*, 100–107. [[CrossRef](#)]
32. Koenig, S.; Likhachev, M. Increment A*. *Neural Inf. Process. Syst.* **2001**, 1539–1546.
33. Stentz, A. The focussed D* algorithm for real-time replanning. *IJCAI* **1995**, *95*, 1652–1659.
34. Nash, A.; Daniel, K.; Koenig, S.; Felner, A. Theta*: Any-angle path planning on grids. *AAAI* **2007**, *7*, 1177–1183.
35. Daniel, K.; Nash, A.; Koenig, S.; Felner, A. Theta*: Any-angle path planning on grids. *J. Artif. Intell. Res.* **2010**, *39*, 533–579. [[CrossRef](#)]
36. Nash, A.; Koenig, S.; Tovey, C. Lazy Theta*: Any-angle path planning and path length analysis in 3D. In Proceedings of the AAAI Conference on Artificial Intelligence 2010, Atlanta, GA, USA, 11–15 July 2010.
37. Nash, A.; Koenig, S.; Likhachev, M. Incremental Phi*: Incremental any-angle path planning on grids. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), Pasadena, CA, USA, 11–17 July 2009.
38. Jun, M.; D’Andrea, R. Path planning for unmanned aerial vehicles in uncertain and adversarial environments. In *Cooperative Control: Models, Applications and Algorithms*; Springer: Boston, MA, USA, 2003; Volume 1, pp. 95–110.
39. Yao, J.; Zhong, C.; Liu, Z.; Xu, J. 3D trajectory optimization for secure UAV communication with CoMP reception. In Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM), Waikoloa, HI, USA, 9 December 2019; pp. 1–6.
40. You, C.; Zhang, R. 3D trajectory optimization in Rician fading for UAV-enabled data harvesting. *IEEE Trans. Wirel. Commun.* **2019**, *18*, 3192–3207. [[CrossRef](#)]
41. González, D.; Pérez, J.; Lattarulo, R.; Milanés, V.; Nashashibi, F. Continuous curvature planning with obstacle avoidance capabilities in urban scenarios. In Proceedings of the 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), Qingdao, China, 8–11 October 2014.
42. Askari, A.; Mortazavi, M.; Talebi, H.A.; Motamedi, A. A new approach in UAV path planning using Bezier–Dubins continuous curvature path. *Inst. Mech. Eng. G J. Aerosp. Eng.* **2016**, *230*, 1103–1113. [[CrossRef](#)]

43. Neto, A.A.; Macharet, D.G.; Campos, M.F. Feasible RRT-based path planning using seventh order Bézier curves. In Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan, 18–22 October 2010.
44. Choi, J.W.; Curry, R.; Elkaim, G. Path planning based on bézier curve for autonomous ground vehicles. In Proceedings of the Advances in Electrical and Electronics Engineering-IAENG Special Edition of the World Congress on Engineering and Computer Science, San Francisco, CA, USA, 22–24 October 2008.
45. Sahingoz, O.K. Generation of bezier curve-based flyable trajectories for multi-UAV systems with parallel genetic algorithm. *J. Intell. Robot. Syst.* **2014**, *74*, 499–511. [[CrossRef](#)]
46. Yang, L.; Song, D.; Xiao, J.; Han, J.; Yang, L.; Cao, Y. Generation of dynamically feasible and collision free trajectory by applying six-order Bezier curve and local optimal reshaping. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–2 October 2015.
47. Nash, A.; Koenig, S. Any-angle path planning. *AI Mag.* **2013**, *34*, 85–107. [[CrossRef](#)]
48. De Filippis, L.; Guglieri, G.; Quagliotti, F. A minimum risk approach for path planning of UAVs. *J. Intell. Robot. Syst.* **2011**, *61*, 203–219. [[CrossRef](#)]