Journal of Mines, Metals and Fuels, *71(12D): 47-52; 2023. DOI: 10.18311/jmmf/2023/47976*

Print ISSN : 0022-2755

Journal of Mines, Metals and Fuels

Contents available at: *www.informaticsjournals.com/index.php/jmmf*

Path Planning Method for Static Obstacle Avoidance Using Bezier Spline Interpolation in UAVs

S. Nithasree, K. Lokesha, Raji George, Shreeram Murali and Anubhav Kini

Department of Mechanical Engineering, Ramaiah Institute of Technology, Bengaluru – 560054, Karnataka, India; nithasree2747@gmail.com, lokesha.krish@msrit.edu, rgeorge@msrit.edu, shreeram.m@gmail.com, anubhavkini@gmail.com

Abstract

Small-scale autonomous unmanned aerial vehicles commonly used for mapping and imagery require long, straight, and level flight trajectories to eliminate distortion caused by the plane's roll. This paper presents the Software-in-the-Loop performance evaluation of a path-planning method for static obstacle avoidance using Bezier spline curves. This method generates two intermediate knots near the periphery of a static circular obstacle, through which a Bezier spline curve is interpolated. This method is then implemented in the firmware of a multi-rotor unmanned aerial vehicle to simulate and evaluate certain flight parameters based on three test cases where the waypoints and obstacle coordinates are already known.

Keywords: *Autonomous Systems, Bezier Curves, Obstacle Avoidance, Path Planning, Unmanned Aerial Vehicles*

1.0 Introduction

The last decade has seen a rapid increase in the usage and applications of autonomous unmanned aerial vehicles (UAVs). One important usage of UAVs is in the field of surveillance, cartography, and other imagery related applications. Large-scale imagery and mapping applications of the future will become highly dependent on the use of UAVs[1]. There is now a growing need for state of the art obstacle avoidance and path planning to optimize collision avoidance. In the case of imagery, small-scale UAVs tend to fly in defined trajectories, usually grid-like, in that area to map them[2]. These trajectories are required to be long, straight, and level, to minimize the distortion effect of the plane's roll[3].

This paper presents an approach to implement static obstacle avoidance in a GPS environment using Bezier-spline interpolation. The method presented presumes that the ground control station (GCS) is provided the information of the obstacle's coordinates, obstacle's radius, starting waypoint, and destination waypoint. The path is planned by finding two knots on the peripheral of an enlarged circle around the obstacle and interpolating a Bezier-spline curve through them. The generated path is then tested by running a Software-in-the-Loop (SITL) simulation of ArduPlane firmware.

Implementing Bezier-spline interpolation for static obstacle avoidance paves the way for more effective use in a number of applications, including mapping, delivery, and surveillance.

2.0 Background

This section discusses two areas of research: path planning for obstacle avoidance and Software-in-the-Loop (SITL) simulations. The first section describes how a path is planned around the obstacle using Bezier spline interpolation[4]. The second section describes the nature of simulations conducted to study the flight characteristics

**Author for correspondence*

of an unmanned aerial vehicle which would help in evaluating its performance during traversing the path.

2.1 Path Planning using Bezier Spline Interpolation

The path planning stage aims at generating an optimum trajectory around an obstacle. There are many established planning algorithms for mobile robots[5]. However, they may require large computational time. The method followed in this paper is to determine certain points called knots and to interpolate a curve through them. A Bezier spline curve is generated by gathering multiple polygonal segments. These segments can be linear, quadrilateral, cubic, or of higher dimensions[6].

A Bezier spline curve is interpolated through four control points and as shown in (1). Since there are four control points, the Bezier curve generated will be a cubic curve. Therefore, the 1st and 2nd derivatives of the curve will be constant at every knot, i.e. continuity is maintained[7].

$$S(t) = (1-t^3)P_0 + 3(1-t^2)tP_1 + 3(1-t)t^2P_2 + t^3P_3 \quad (1)$$

The four points are control points. To interpolate a Bezier curve through knots, the two derivative equations (2) and (3) are taken to find the control points.

$$S'(t) = -3(1-t)^2P_0 + 3(1-4t+3t^2)P_1 + 3(2t-3t^2)P_2 + 3t^2P_3 \quad (2)$$

$$S''(t) = 6(1-t)P_0 + 3(-4+6t)P_1 + 3(2-6t)P_2 + 6tP_3 \quad (3)$$

A tridiagonal system of equations is generated and is solved using the Thomas algorithm as shown[8].

2.2 Software-in-the-Loop Simulations

The Software-in-the-Loop (SITL) simulation allows one to execute an autopilot system firmware without hardware. The firmware used is ArduPlane, an open-source firmware stack highly popular among the UAV community[9]. The ArduPlane firmware runs on many popular off-the-shelf flight controllers such as the ArduPilot Mega (APM), Pixhawk, the Cube, the Navio2, etc.

The ground control station (GCS) software Mission Planner has an implementation of an SITL simulator which conducts a multi-parametric simulation of the ArduPlane code using a regular C++ compiler[10]. A flight dynamics model provides the necessary sensor data. The cases are simulated for a multi-rotor UAV frame on Mission Planner.

3.0 Method

UAVs need path planning as their initial step in autonomous navigation. Paths are specified in the form of waypoints[11]. The following section describes how the obstacle avoidance method is carried out. Section 3.1 describes the problem scope and the extent of the parametric evaluation. Section 3.2 and 3.3 describe the interpolation method carried out. Section 3.4 describes the specific test cases considered for evaluation.

3.1 Problem Scope

The start and destination waypoints are known. The straight-line connecting the two points is designated as the "original path." The static obstacle is a circle of constant radius whose center coordinates are known. The path will be planned by finding two convenient knots near the circle through which a B-spline curve will be interpolated from the starting waypoints to the destination waypoint. The deviation from the obstacle must ideally be minimum, but the obstacle must be avoided with sufficient clearance. The generated path will be simulated for a multi-rotor frame to be evaluated for the following parameters: path length, maximum roll angle, perpendicular deviation from straight-line path, and roll characteristics, i.e. variation of bank angle with respect to time.

3.2 Knot Computation and Interpolation

To construct a cubic Bezier curve around an obstacle in between two waypoints, four knots are required. The first and last knots are the waypoints themselves. The remaining two knots are calculated by drawing a safety circle of 1.25 times the radius of the obstacle in the position of the obstacle. Two lines are drawn from the center of the obstacle to the circumference of the safety circle at an angle of 60° and 120° respectively with the diameter of the safety circle that is parallel to the straight-

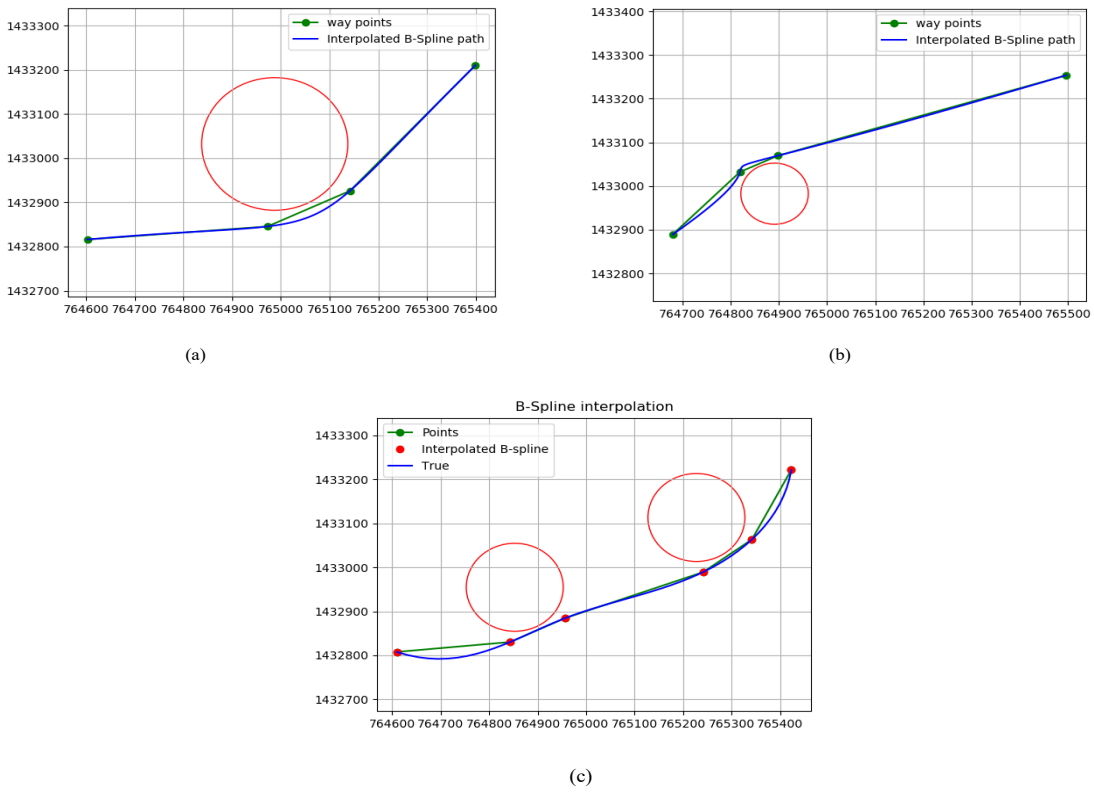


Figure 1. The 3 Figures depict the obstacle, waypoints, and path generated in a) Case 1, b) Case 2, and c) Case 3, respectively.

line path between the two waypoints. The points at which these lines intersect with the safety circle are used as the remaining knots. An illustrative depiction is shown in Figure 1.

3.3 Implementation

The path planning computation is done in a script which generates the waypoints to be fed for simulation. Since the waypoint and obstacle coordinates are GPS coordinates, they were converted into Universal Transverse Mercator (UTM) cartesian coordinates[12]. The UTM system used is World Geodetic System 1984 (WGS84)[13]. With the coordinates in cartesian format, a collision checker checks if the obstacle circle intersects with the straight-line joining the two waypoints. If there's a collision detected, the two interpolation knots are computed as described in section 3.2.

A Bezier curve is then generated with four knots, the result of which is a smooth cubic Bezier curve. If there are multiple obstacles, the collision checker is then run again, iteratively, to check for collision in the new trajectory. If another collision is detected, then the third knot acts as the first waypoint, and two other interpolation knots are computed. This process is repeated iteratively as many times as there are obstacles. For multiple obstacles, the generated path is a piecewise combination of multiple cubic Bezier curves, because only 4 knots are considered for interpolation at each time. Lastly, all the generated interpolation knots are converted back into GPS coordinates and written into the waypoint file for GCS simulation and execution. This work-flow is illustrated in Figure 2.

The generated waypoint file has the desired coordinates for execution. SITL is a multiparametric simulation of the ArduPilot firmware. The version of the firmware used was 4.0.5. The frame chosen to be simulated is that of a multi-rotor UAV. The cruise speed was set at 5 m/s and the maximum bank angle was capped at 60 degrees to avoid

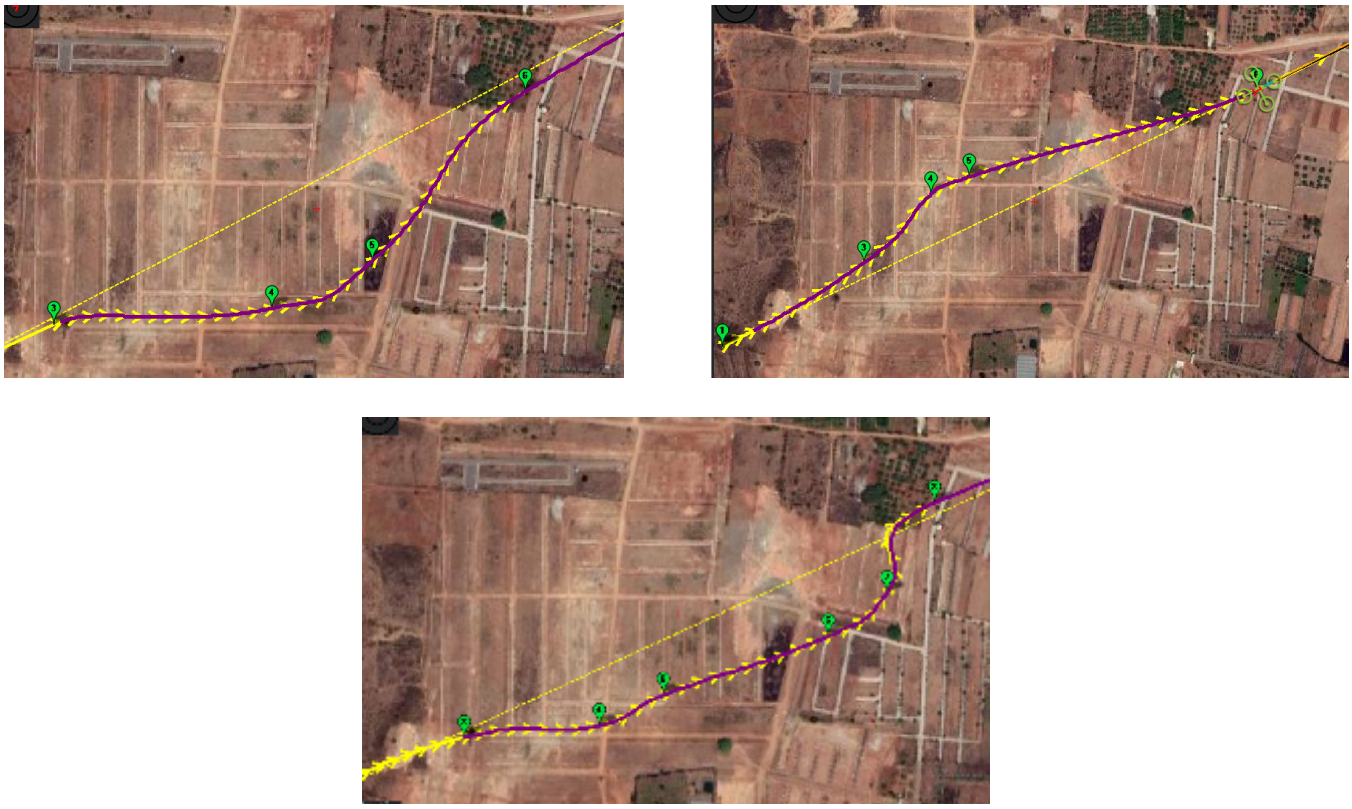


Figure 2. Figures illustrate the simulated trajectories implemented on the ground control station.

stalling. The total altitude was 100 meters with variation less than 10 m.

3.4 Test Cases

To test the obstacle avoidance setup, three test cases were chosen. The first case, Case 1, is with a large obstacle in the center of the two waypoints. The radius of this obstacle is 150 meters. Case 2 is with a smaller obstacle of radius 70 meters displaced to one side. Case 3 is with two obstacles, both of radius 70 meters. Case 1 was chosen for the maximum deviation from the straight-line path. Case 2 was chosen to evaluate the "straightness" of the generated path from an imagery context. Case 3 was chosen to observe how the path is planned in circumstances with multiple obstacles. However, all the three test cases are observed in equal standing for all parameters described in section 3.1.

The coordinates of the waypoints, obstacle center, and its radius is tabulated in Table 1. The radius shown is in meters.

Table 1. Waypoint and obstacle coordinates

Case	Start	Destination	Obstacle Center	Obstacle Radius	
1	12.949521, 77.438892	12.953014, 77.446268	12.951441, 77.442456	150	
2	12.950173, 77.439607	12.953398, 77.447167	12.951000, 77.441565	70	
3	12.949445, 77.438956	12.953113, 77.446487	12.950752, 77.441211 12.952153, 77.444676	70	70

4.0 Results

The simulation was conducted in three test cases as described in section 3.4 and labelled as Case 1, Case 2, and Case 3 in the following figures. Figure 1 shows the path planning stage of Case 1, Case 2, and Case 3.

Figure 2 shows the implemented path on Mission Planner using Bezier-spline waypoints. The trajectory is tracked by the means of a PID controller[14]. Based on the implementation shown in Figure 3, the simulated results were obtained, which is shown in Table 2 and Figure 5. Table 2 lists the observed parameters observed in the cases. It can be seen from Figure 3 that a path to avoid the static obstacles has been successfully created in all the 3 test cases. The blue path shows the interpolated B- spline path. Case-1 shows a highly deviated path and Case-3 shows the interpolated path in case of multiple obstacles.

Table 2. Observed parameters

Case	Path Length	Original Length	Maximum Perpendicular Deviation	Maximum Roll Angle
1	1,980	950	173	2
2	1,950	970	82	2
3	1,940	940	125	3

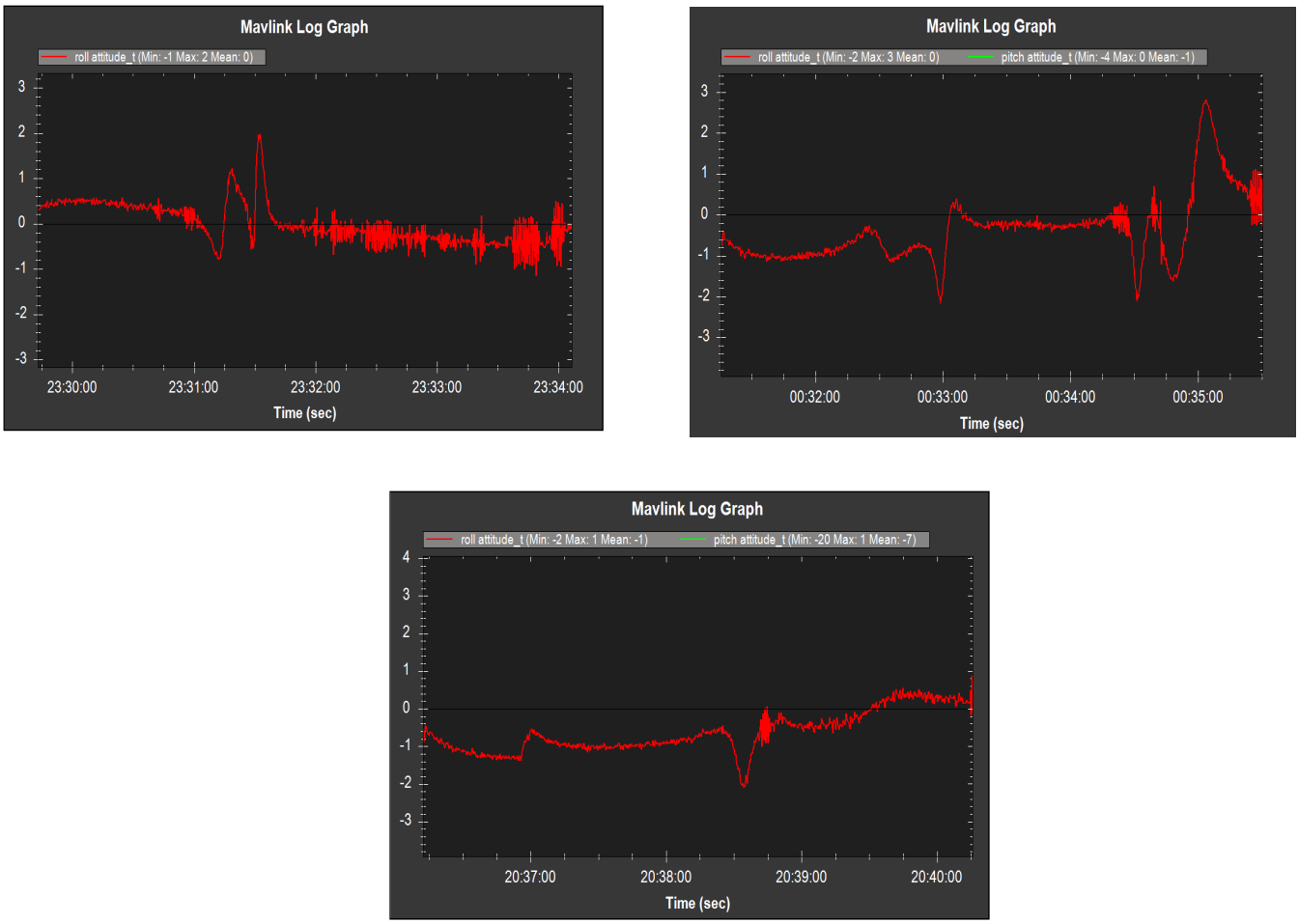


Figure 3. The 3 Figures plots the variation of bank angle with respect to time in Case 1, Case 2, and Case 3, respectively.

5.0 Conclusion

The Bezier spline-based path planning algorithm was tested on three cases based on: path length, original length, maximum perpendicular deviation from straight-line path, and maximum roll angle. It is observed that the path generated through B-spline interpolation is highly feasible in the context of imagery applications. This is largely due to two factors: the overall variation of roll during flight is low and the UAV maintains roll stability for the majority duration of the flight. Trivially, it is observed that the maximum deviation occurs when there are large obstacles. For the considered cases, the average excess distance covered due to obstacle avoidance is 1,003 meters. The maximum roll angle is achieved near the interpolation knots where the B-spline curve tends to generate moderately sharp turns. In the context of imagery, this path planning algorithm is preferred in small-scale UAVs because of the smooth and straight paths generated.

6.0 Acknowledgement

The support of the Head of Department of Mechanical Engineering, Dr. Raji George, is duly acknowledged. The support of Edhitha UAS—a student research group—and its laboratory facilities has played a vital role in producing this research.

7.0 References

1. Boccardo P, Chiabrando F, Dutto F, Tonolo FG, Lingua A. UAV deployment exercise for mapping purposes: Evaluation of emergency response applications. Sensors. 2015; 15(7):15717-15737. https://doi.org/10.3390/s150715717. PMid:26147728 PMCid: PMC4541851.
2. Auto Grids; Planning a Mission with Waypoints and Events. ArduPilot, ArduPilot Documentation. Available from: https://ardupilot.org/planner/docs/common-planning-a-mission-with-waypoints-and-events.html#auto-grid. Accessed 21 Jul 2020.
3. Lelong CC, Burger P, Jubelin G, Roux B, Labbé S, Baret F. Assessment of unmanned aerial vehicles imagery for quantitative monitoring of wheat crops in small plots. Sensors. 2008; 8(5):3557-3585. https://doi.org/10.3390/s8053557. PMid:27879893 PMCid: PMC3675559.
4. Zhang H, Feng J. B-spline interpolation and approximation. Digital Geometry Processing Course Lecture Notes, Zhejiang University. 2006.
5. LaValle SM. Planning algorithms. Cambridge: Cambridge University Press; 2006. https://doi.org/10.1017/CBO9780511546877. PMCid: PMC1482629.
6. Knott GD. Interpolating cubic splines. Vol. 18. Springer Science and Business Media; 2000. https://doi.org/10.1007/978-1-4612-1320-8.
7. Bartels RH, Beatty JC, Barsky BA. An introduction to splines for use in computer graphics and geometric modelling. San Francisco: Morgan Kaufmann; 1995.
8. Breida L. Smooth Bezier curve through prescribed points. 2012. Available from: https://www.particleincell.com/2012/bezier-splines/. Accessed 21 Jul 2020.
9. ArduPilot. Available from: https://ardupilot.org/ardupilot/index.html. Accessed 21 Jul 2020.
10. ArduPilot Developer Documentation. SITL Simulator, Software in the Loop. 2020. Available from: https://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html. Accessed 21 Jul 2020.
11. Fossen TI. Handbook of marine craft hydrodynamics and motion control. John Wiley and Sons; 2011. https://doi.org/10.1002/9781119994138.
12. Universal Transverse Mercator. 1983. Last updated Jul 2020. Available from: https://proj.org/operations/projections/utm.html?highlight=utm. Accessed 21 Jul 2020.
13. National Geospatial-Intelligence Agency. World Geodetic System (1984). Available from: https://www.nga.mil/ProductsServices/GeodesyandGeophysics/Pages/WorldGeodeticSystem.aspx. Accessed 21 Jul 2020.