2008 IEEE/RSJ International Conference on Intelligent Robots and Systems
Acropolis Convention Center
Nice, France, Sept, 22-26, 2008

3D Smooth Path Planning for a UAV in Cluttered Natural Environments

Kwangjin Yang Salah Sukkarieh

ARC Centre of Excellence in Autonomous Systems
Australian Centre for Field Robotics
University of Sydney, Australia
Email : {k.yang, salah}@cas.edu.au

***Abstract*— This paper presents a 3D path planing algorithm for an unmanned aerial vehicle (UAV) operating in cluttered natural environments. The algorithm satisfies the upper bounded curvature constraint and the continuous curvature requirement. In this work greater attention is placed on the computational complexity in comparison with other path-planning considerations. The Rapidly-exploring Random Trees (RRTs) algorithm is used for the generation of collision free waypoints. The unnecessary waypoints are removed by a simple path pruning algorithm generating a piecewise linear path. Then a path smoothing algorithm utilizing cubic Bézier spiral curves to generate a continuous curvature path that satisfies the minimum radius of curvature constraint of UAV is implemented. The angle between two wapoints is the only information required for the generation of the continuous curvature path. The result shows that the suggested algorithm is simple and easy to implement compared with the Clothoids method.**

I. Introduction

The field of robot motion planning has greatly matured since the original work in the 1970s. However, autonomous navigation of Unmanned Aerial Vehicles (UAVs) in complex environments is still an active research topic. The path planning problem for UAVs is difficult since these vehicles have fast and complicated dynamics and are compounded by the issues of real time navigation in 3D space.

There are several considerations for an ideal path planner including: optimality; completeness; and computational complexity. There is a natural trade off between these elements [1]. For UAV applications, computational complexity is the most important requirement since path planning has to occur quickly due to fast vehicle dynamics. If previously unknown obstacles are detected, the UAV has to replan the path in real time to avoid these obstacles. If the path planner fails to generate a safe path within bounded time, collisions with obstacles may result. Since the computational time of deterministic and complete algorithms grows exponentially with the dimension of the configuration space, these algorithms do not provide an adequate solution for real-time UAV path planning in unknown natural environments [2].

Recently, sampling-based motion planning has gained much interest as an alternative to complete motion planning methods. The Maneuver Automaton, which is a kind of hybrid vehicle modeling strategies, is a good alternative to solve real time path planning problem [2]. However it is difficult to decide the finite set of motion primitives. If these sets are small the solution may be severely suboptimal but if these are too large the computational time can be increased dramatically. Rapidly-exploring Random Trees (RRTs) have been demonstrated successfully in UAV applications [2], [3], [4], [5]. The standard RRT algorithm produces a time-parameterized set of control inputs to move from the initial state (x_{start}) to the goal state (x_{goal}). The validity of the result is dependant on the accuracy of the state-space model being used. In real UAV applications, we encounter sensor inaccuracies, wind, and other unmodeled factors. These are some of the key disadvantages of using an open-loop path planner [3].

In this paper, we use closed-loop path planning instead of generating the time-parameterized set of control inputs. Collision free waypoints are generated for the UAV and this path consists of straight line segments. Since it is not possible for the helicopter to follow this path without stopping at sharp angles, it is necessary to smooth the path.

Smooth path planning has been tackled actively in the mobile robot community. A smooth path is essential for mobile robot navigation, because non-smooth motions can cause the slippage of wheels which degrades the robots dead-reckoning ability [6]. Several methods may be applied to generate a smooth path. A popular method is Dubins curves [7], [8], [9]. This algorithm computes the shortest path between two postures in the plane via the concatenation of line segments and arcs of circles taking into account the vehicles maximum rate of change of turn rate [10]. Dubins' seminal work has since been extended to other more complex vehicle models but is still confined to line segments and arcs of circles.

The non-continuous curvature of Dubins curve results in difficulty with control execution: at the junction of a straight line and an arc, mobile robots need to stop their wheel motion to make perfect tracking achievable [11]. To overcome this problem several researchers have proposed to smooth the path by combining line segments, circular arcs and clothids [12], [13]. Clothoid pairs provide smooth transitions with continuous curvatures and have the advantage of providing the minimum-length curves for a given limit on jerk [14]. However, such curves have the disadvantage that no closed-form expression of the position can be given [15], and they are not very flexible in matching the conditions placed at the endpoints of the path [16] and are dangerous in the presence of obstacles [17]. Moreover, the computational cost of clothoids are higher than parametric curves since the evaluation of Fresnel integrals are required to generate a curve.

Another approach to make continuous curvature path is to use splines. In [18], B-spline is used to generate a planar smooth curve avoiding obstacles and this method has been applied to the path planning of autonomous ground vehicles in [19]. However to apply this method, a safe corridor which does not intersect the obstacle must be constructed in advance for collision avoidance. A seventh order polynomial spline is used to generate a G^3 continuous path in [20], but its computational costs are expensive due to its high order.

In the UAV community, most researchers apply the Dubins algorithm [21], [22], [23] to generate a smooth path. Cubic splines are used to generate a smooth path for an autonomous unmanned helicopter in [4] but it is still of C^1 continuity as with the Dubins curves. When the helicopter begins to fly the circular path, there is a step change of angular velocity as the pervious angular velocity is zero as the vehicle would have been traversing along the straight line segment. The path can not be smoothly executed, since a helicopter can not immediately accelerate to the correct angular velocity [24].

Cubic Bézier curves can be used to generate a continuous curvature path. However a significant disadvantage of a parametric cubic curve is that its curvature is a complicated function of its parameter. It is thus not easy to use cubic curve segments in the design of curvature controlled curves [25]. Continuous curvature curve generation using composite Bézier curves is shown in [26] and results on the number and location of curvature extrema of planar parametric cubic curve is shown in [25]. It is a difficult problem to generate a parametric curve satisfying both G^2 continuity and maximum curvature requirements at the same time. This problem is solved in [27] but it is too complicated to be applied in real-time.

In this paper, a computationally efficient path smoothing method is suggested. The main contribution of this paper is a computationally efficient 3D path planning algorithm which satisfies a G^2 continuity and nonholonomic constraints.

The rest of the paper is organized as follows: Section II describes 2D path planning methods. Section III presents the curvature continuous path smoothing algorithm. Extending these methods to 3D is described in Section IV. Simulation results are shown at Section V. Conclusions and future works are presented thereafter.

II. 2D Path Planning

Fig. 1 (a) shows our autonomous helicopter flying in the flight test site. This helicopter will be used for both the surveying and spraying of weeds in areas where conventional weed control techniques could not be applied easily. There are two scenarios of interest for path planning. Firstly, the helicopter must survey where the weeds exist. For this mission, the helicopter has to fly close to the ground at a constant altitude and has to detect and classify the harmful

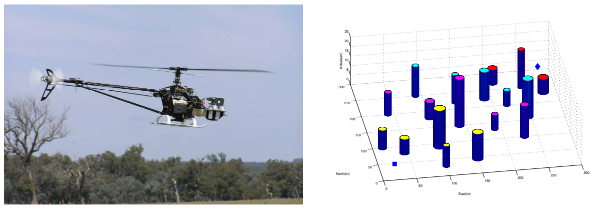

(a) (b)

Fig. 1. (a) Our helicopter flying in the flight test site located in Marulan. (b) The simulation environment cluttered with "trees".

weeds. In this case, 2D path planning is needed as the altitude is kept constant. After surveying the environment, the next task is to spray the chemical on to the weeds. To perform this operation, the helicopter must fly to the area where these weeds exist as soon as possible after take off to perform the spraying mission. In this case, 3D path planning is required

We first consider path planning with constant altitude for surveying the weeds. Fig. 1 (b) shows the simulation environment cluttered with trees. We assumed that there are trees between the starting point(square) and the target point(diamond) with different radii and height and the helicopter has to fly with constant altitude(5m).

A. RRT Algorithm

RRT was first suggested in [28] as a alternative to complete path planning in high degree of freedom situations. First RRT selects a random state x_{rand} and then finds the nearest node to the x_{rand} in terms of distance metric ρ. The algorithm then select the control input u_{new} that minimizes the distance from x_{near} to x_{rand} and finally checks for collision. If there is no collision, x_{new}, u_{new} are added as a new vertex in the tree. This control-theoretic method is open-loop planning and works well if the state-space model of the platform is exact. However, if there is a model mismatch, then the generated path may be differ from the real path. Moreover, since the sensor information is not perfect, the path may also be unsafe. Finally, in natural environments, there are always winds, and these disturbances will affect a UAVs maneuver. An open loop path planning method cannot consider these disturbances at the planning stage and cannot compensate for their effects.

Instead of generating a time-parameterized set of control inputs, we decouple the path planning and control problems. We generate a purely geometric path using a variant of RRT to define waypoints. Our RRT algorithm operates as follows. First, a position x_{rand} is chosen at random from within the workspace, and this point is compared with existing tree nodes to find the closest point in the tree, x_{near}. A line is drawn connecting x_{near} to x_{rand}, and a new point x_{new} is generated along this ray at a fixed distance d from x_{near}. If there is no collision on the interval between x_{near} and x_{new}, the latter is added to the tree. A greedy variation of the algorithm generates points successively along the ray connecting x_{near} and x_{rand} until a collision occurs. We can reduce the computational time for path planning

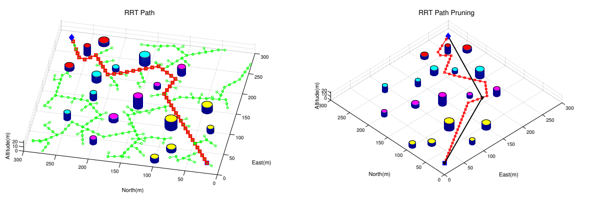


Fig. 2. (a) Path planning using Biased-Greedy RRT algorithm. (b) Path pruning algorithm: The path consists of 45 nodes but after removing redundant waypoints, the number of nodes is reduced to only 1.

by biasing the generation of x_{rand}, such that 10% of the time it equals x_{goal} rather than a random location. Further efficiency can be obtained by combining Greedy and Biased RRT. We tested the performance of Biased-Greedy RRT with different environments (different size of environment, different number and location of obstacles) and in every case it shows better performance than biased RRT. This Biased-Greedy RRT algorithm is used for waypoint generation in the rest of this paper.

Fig. 2 (a) shows the path planning result of Biased-Greedy RRT Algorithm. The green lines are all trees generated by the algorithm and the red line is the shortest path which connects the starting point and the target point.

The comparison of performance of this algorithm with different environments is shown in [29] with a 1000 simulation runs. Despite the random nature of RRT, our following path smoothing algorithm generates a curvature continuous path without any failure.

B. Path Pruning

Even though RRT is an effective and computationally efficient tool for complex online motion planning, the solution is far from optimal due to its random exploration of the space. It is thus required to remove unnecessary waypoints. In [5], Dijkstra's algorithm is used to prune the waypoints to generate a near optimal path, but this method requires a bit large amount of computational time. Therefore it loses the strong advantage of RRT, namely fast planning. Instead, we use a simple but quite efficient method which can quickly find a path that eliminates most extraneous nodes.

Let the original path of nodes from start to goal point be denoted $\{x_1, \ldots, \boldsymbol{x}_N\}$, such that x_N is the goal location. Let the pruned path be initially an empty set, and let $j = N$. The pruning operation is as follows. First add x_j to the pruned path. Then for each $i \in [1..j-1]$, check the line between (x_i, x_j) for a collision, stopping on the first x_i without collision. Let $j = i$, add x_j to the pruned path, and repeat the process until a complete path is generated. Even though this path is not optimal, this method can get rid of unnecessary waypoints very quickly. Fig. 2 (b) shows the result of path pruning algorithm applied to the initial RRT path in Fig. 2 (a). The path has 45 nodes between starting and target point initially but the number of nodes is reduced to only 1 after the redundant waypoints are pruned.

C. Path Smoothing

The path in Fig. 2 (b) is piecewise linear and not followable for a UAV with kinematic and dynamic constraints. Therefore, it is necessary to make the path smooth in order to be suitable for UAVs.

A helicopter can fly in linear segments at low speed with the capability to stop and hover at each waypoint. However, at higher speeds, a helicopter would not be able to negotiate turns at smaller radii, which imposes the demand for a planner that accommodates for nonholonomic constraints [4].

An alternative is for the helicopter to reduce its speed in order to negotiate the path having large curvature. Since this increases the flight time, the aim is to generate a path which satisfies the maximum curvature constraint of the helicopter.

Two methods are considered for the comparison of path smoothing: C^1 Continuous Cubic Bézier Curve(C1CBC) and G^2 Continuous Cubic Bézier Spiral(G2CBS).

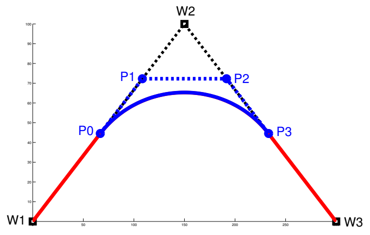


Fig. 3. C^1 Continuous Cubic Bézier Curve(C1CBC) Path Smoothing

1) C1CBC Path Smoothing: Let the degree n Bézier curve with $n+1$ control points $(P_0, P_1, \cdots, P_n)$ be defined as [30]

$$P(s) = \sum_{i=0}^{n} P_i B_{n,i}(s) \tag{1}$$

where the coefficients $B_{i,n}(s)$ are named Berstein polynomials and are defined as follows:

$$B_{n,i}(s) = \binom{n}{i} s^i (1-s)^{n-i} \tag{2}$$

Since the computational cost increases with degree n, a lower degree of Bézier curves is preferable. A cubic Bézier curve is the minimum degree curve in which a continuous curvature locus can be generated. Four control points (P_0, P_1, P_2, P_3) are needed to make a cubic Bézier curve, where P_0 and P_3 are the curve endpoints. Our concern is to connect two curves smoothly. A C^1 continuous curve between two curves may be obtained if the first derivative of the two curves at the junction point are the same. If we take the derivative of equation (1), the following first derivative of Bézier curves equation is obtained.

$$\dot{P}(s) = \sum_{i=0}^{n-1} n(P_{i+1} - P_i) B_{n-1,i}(s) \tag{3}$$

From this equation, we can see that consecutive segments in a composite Bézier curves can be made C^1 continuous simply by arranging that the first control vertex prior to endpoint of the first curve, the shared endpoint, and the second vertex of the next curve to be collinear and equally spaced [30]. Therefore the first two control points P_0, P_1 must be located between W_1 and W_2 and the last two control points P_2, P_3 must be located between W_2 and W_3 as can be seen in Fig. 3. The 4 control points can be decided by the following equation:

$$\begin{aligned} P_0 &= W_2 + d_1 \cdot u_1, \qquad P_1 = W_2 + \frac{d_1 \cdot u_1}{2}, \\ P_2 &= W_2 - \frac{d_2 \cdot u_2}{2}, \qquad P_3 = W_2 - d_2 \cdot u_2 \end{aligned} \quad (4)$$

where u_1 is a unit vector between W_2 and W_1 and u_2 is that of W_3 and W_2, and d_1 is a length between W_2 and P_0 and d_2 is that of W_2 and P_3 .

However, this C^1 continuous curve has a discontinuity of curvature at the joint. We want to generate a continuous curvature curve, namely a G^2 continuous curve. The curvature of a planar parametric curve is defined as follows [30]:

$$\kappa(s) = \frac{|\dot{P}(s) \times \ddot{P}(s)|}{|\dot{P}(s)|^3} \quad (5)$$

Therefore, the path must satisfy the condition of equation (5) at the junction points instead of equation (3).

III. G2CBS Path Smoothing

A. G^2 Continuity Condition

Generating a G^2 continuous curve is not as simple as making a C^1 continuous curve. Planar G^2 transition curves are generated, composing of cubic Bézier spiral segments [16]. Spiral segments are defined to have no interior curvature extrema. Therefore if the value of equation (6) is not zero, it becomes a cubic Bézier spiral.

$$\dot{\kappa} = \frac{(\dot{P} \cdot \dot{P}) \left(\dot{P} \times \dddot{P}\right) - 3 \left(\dot{P} \times \ddot{P}\right) \left(\dot{P} \cdot \ddot{P}\right)}{||\dot{P}||^5} \quad (6)$$

The solution of this equation is very complicated if a pure parametric approach is used. This is the significant disadvantage of a parametric curve. However, if the P_0 is moved to $(0, 0)$ and P_0, P_1, P_2 are aligned in x-axis after translation and rotation and a geometric property of the control polygon is used, it is possible to reduce the complexity of the curve.

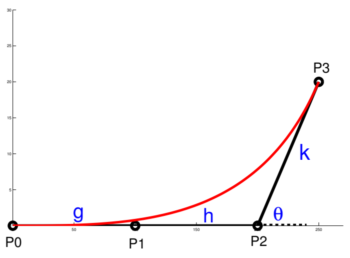


Fig. 4. Geometric Interpretation of Cubic Bézier curves.

Fig. 4 shows a planar cubic Bézier curve with 8 degrees of freedom, which may be described in just 4 degrees of freedom according to the three lengths (g, h, k) and a angle (θ). If we substitute these four variables into equation (6), three conditions for generating a spiral curve may be found.

$$\frac{g}{h} \geq 0.58, \qquad \frac{k}{h} \leq \frac{6\cos\theta}{\frac{g}{h} + 4} \qquad and \qquad 0 < \theta < \frac{\pi}{2} \quad (7)$$

If these conditions are applied to connect two straight lines, the following expression can be obtained which make a G^2 continuous composite curve.

$$\begin{aligned} F(\beta) &= \cos^2(\gamma - \beta)\sin\beta[(7.2364 + 6\cos^2\beta)D_y \\ &\quad -6D_x\cos\beta\sin\beta] + \cos^2\beta\sin(\gamma - \beta) \\ &\quad [7.2364(D_y\cos\gamma - D_x\sin\gamma) + \\ &\quad 6\cos(\gamma - \beta)(D_y\cos\beta - D_x\sin\beta)] \\ &= 0 \end{aligned} \quad (8)$$

where $D = (D_x, D_y) = E_0 - B_0$ and β and γ are angles as can be seen in Fig. 5 (a).

As the only known variable is γ, D_x and D_y need to be determined in order to solve equation (8). If we select d_1 to be the length between P_2 and B_0, and d_2 as the length between P_2 and E_0, the solution of equation (8) can be obtained by the bisection method. Having obtained the solution to equation (8), the eight control points can be determined which construct two cubic Bézier spiral curves. The first curve consist of following the four control points

$$\begin{aligned} B_0 &= P_2 + d_1 \cdot u_1, \qquad B_1 = B_0 + g_b \cdot u_1, \\ B_2 &= B_1 + h_b \cdot u_1, \qquad B_3 = B_2 + k_b \cdot u_d \end{aligned} \quad (9)$$

The second curve consists of the following four control points

$$\begin{aligned} E_0 &= E_1 - k_e \cdot u_d, \qquad E_1 = E_2 - h_e \cdot u_2, \\ E_2 &= E_3 - g_e \cdot u_2, \qquad E_3 = P_2 - d_2 \cdot u_2 \end{aligned} \quad (10)$$

where

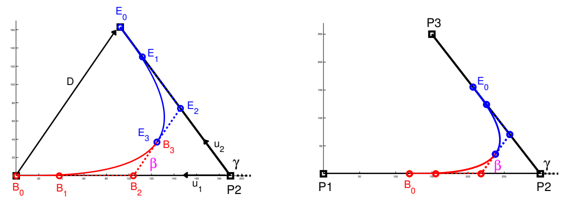
$$\begin{aligned} h_b &= \frac{4.58D_y\cos^2(\gamma - \beta)\sin\beta}{[1.2364\cos\beta\sin(\gamma - \beta) + 6\sin\gamma]\cos\beta\sin\gamma}, \\ h_e &= \frac{h_b\cos^2\beta\sin(\gamma - \beta)}{\sin\beta\cos^2(\gamma - \beta)}, \\ g_b &= 0.58 \cdot h_b, \\ g_e &= 0.58 \cdot h_e, \\ k_b &= 1.31 \cdot h_b\cos\beta, \\ k_e &= 1.31 \cdot h_e\cos(\gamma - \beta) \end{aligned} \quad (11)$$

and u_1 is a unit vector between P_2 and P_1, u_2 is that of P_3 and P_2 and u_d is a unit vector between B_2 and E_2.

B. Analytical Solution of G^2 Continuity

In order to determine the 8 control points, equation (8) must be solved. If there are many cusps to smooth, the computational time is increased.

Our goal is to find the solution of equation (8) without using numerical methods. The length of d_1 is assigned to be the same as d_2 $(d = d_1 = d_2)$, which does not change the maximum curvature of the curve. With this condition, an analytical solution of equation (8) can be obtained.



(a) (b)

Fig. 5. (a) Curvature continuous condition to connecting two lines. (b) G2CBS path smoothing result between two line segments.

Theorem 1. *If $d_1 = d_2$ and $\beta = \frac{\gamma}{2}$, then there exists curvature continuous curve which connects two straight lines.*

Proof. If one line is aligned to the x-axis and the same design variable d is applied, D_x and D_y can be decided as follows (see Fig. 5 (a)).

$$D_x = d + d\cos\gamma, \qquad D_y = d\sin\gamma \tag{12}$$

By substituting equation (12) into equation (8), the following equation is obtained.

$$\begin{aligned}F(\beta) = {} & \cos^2(\gamma-\beta)\sin\beta[(7.2364+6\cos^2\beta)d\sin\gamma- \\ & 6(d+d\cos\gamma)\cos\beta\sin\beta]+\cos^2\beta\sin(\gamma-\beta) \\ & [7.2364(d\sin\gamma\cos\gamma-(d+d\cos\gamma)\sin\gamma)+ \\ & 6\cos(\gamma-\beta)(d\sin\gamma\cos\beta-(d+d\cos\gamma)\sin\beta)]\end{aligned} \tag{13}$$

Using a trigonometrical function, equation (13) can be arranged as follows

$$\begin{aligned}F(\beta) = {} & d\cos^2(\gamma-\beta)\sin\beta[(7.2364+6\cos^2\beta)\sin\gamma- \\ & 6\cos^2\frac{\gamma}{2}\sin 2\beta]+d\cos^2\beta\sin(\gamma-\beta) \\ & [-7.2364\sin\gamma+6\cos(\gamma-\beta)(\sin(\gamma-\beta)-\sin\beta)] \\ = {} & 7.2364d\sin\gamma\,\underline{[\cos^2(\gamma-\beta)\sin\beta-\cos^2\beta\sin(\gamma-\beta)]}+ \\ & 6d\cos^2(\gamma-\beta)\sin\beta\,\underline{\left[\cos^2\beta\sin\gamma-\cos^2\frac{\gamma}{2}\sin 2\beta\right]}+ \\ & 6d\cos^2\beta\sin(\gamma-\beta)\cos(\gamma-\beta)\,\underline{[\sin(\gamma-\beta)-\sin\beta]}\end{aligned} \tag{14}$$

If β is chosen as a half of γ ($\beta = \frac{\gamma}{2}$), the spiral condition of equation (8) is always satisfied. □

If we use these conditions, all 8 control points can be easily determined. For example, the h_b and h_e values can be obtained as follows:

$$\begin{aligned}h_b &= \frac{4.58d\sin 2\beta\cos^2\beta\sin\beta}{[1.2364\cos\beta\sin\beta+6\sin 2\beta]\cos\beta\sin 2\beta} = 0.346d \\ h_e &= \frac{h_b\cos^2\beta\sin\beta}{\sin\beta\cos^2\beta} = h_b\end{aligned} \tag{15}$$

Therefore, equation (11) can be reduced as follows.

$$\begin{aligned} h_b &= 0.346d, & h_e &= h_b, \\ g_b &= 0.58\cdot h_b, & g_e &= 0.58\cdot h_b, \\ k_b &= 1.31\cdot h_b\cos\beta, & k_e &= 1.31\cdot h_b\cos\beta \end{aligned} \tag{16}$$

From equation (16), we can see that G2CBS path can be generated once we determine d.

C. Nonholonomic Constraint of G2CBS

In the G2CBS path, the only design variable is d if we use Theorem 1. Therefore, we have to decide the d value which satisfies this constraint.

Theorem 2. *If the maximum curvature of the UAV is given by κ_{max}, then the design variable d which satisfies this maximum curvature constraint can be decided as follows:*

$$d_{\kappa_{max}} = \frac{1.1228\sin\beta}{\kappa_{max}\cdot\cos^2\beta}$$

Proof. The maximum curvature of cubic Bézier spiral is located at ending point since its curvature monotonically increases from the starting point to the ending point.

The first and second derivative of a cubic Bézier curve are obtained such as equation (17) if the geometric property is used as shown in Fig. 4.

$$\dot{P}(s) = 3\overrightarrow{k}, \quad \ddot{P}(s) = 6\overrightarrow{k} - 6\overrightarrow{h} \tag{17}$$

Substituting these values into equation (5), the maximum curvature of the cubic Bézier spiral can be obtained as follows:

$$\begin{aligned}\kappa_{max} &= \frac{|3\overrightarrow{k}\times(6\overrightarrow{k}-6\overrightarrow{h})|}{|3\overrightarrow{k}|^3} = \frac{18kh\sin\theta}{27k^3} \\ &= \frac{2h\sin\theta}{3k^2}\end{aligned} \tag{18}$$

The G2CBS consists of two curves and they have the same curvature profiles. Without loss of generality, we can use the first curve to find out the maximum curvature. Since $k_b = 1.31h_b\cos\beta$ and $h_b = 0.346d$ from equation (16), the maximum curvature can be obtained as follows:

$$\kappa_{max} = \frac{2(0.346d)\sin\beta}{3[1.31(0.346d)\cos\beta]^2} = \frac{1.1228\sin\beta}{d\cos^2\beta} \tag{19}$$

Therefore the design variable d which satisfies the maximum curvature constraint can be obtained as follows.

$$d = \frac{1.1228\sin\beta}{\kappa_{max}\cdot\cos^2\beta} \quad \square \tag{20}$$

D. Collision Checking

The collision checking algorithm is executed after G2CBS path smoothing. If there is no collision, d will be applied but if there is collision, d will be reduced and checked again to see if collision still exist. To make the path smoother, d can be increased until collision is detected. The maximum length of d is dependent on the angle between the waypoints and it is not the scope of this research.

IV. 3D Path Planning

This section extends the path planning algorithms to 3D. The 2D Biased-Greedy RRT algorithm to be extended to 3D easily as well as pruning algorithm. However, to apply a G2CBS path smoothing algorithm to 3D is not as straightforward since it is based on a 2D plane.

A. G2CBS Path Smoothing

There are two approaches to solve 3D G2CBS path smoothing. The first method is to directly solve the problem in 3D space. This is a quite complex problem and is not easy to find the analytical solution. The second method is to map 3D waypoints to 2D, apply the 2D G2CBS path smoothing method and then map these values back again into 3D. Since the path smoothing algorithm is applied on consecutive triplets of waypoints, these 3 waypoints form a plane. Therefore, this plane can be mapped from the 3D space plane (in global frame) on to a 2D plane using a homogenous coordinate transformation.

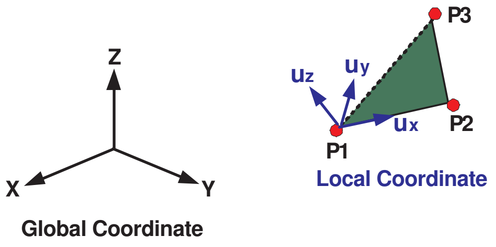


Fig. 6. Coordinate Transformation : Using coordinate transformation, the 3D space can be mapped to 2D plane

The 3D G2CBS path smoothing algorithm can be described in the following steps.

The first step is to find 3 unit vectors (u_x, u_y, u_z) in the $P_1P_2P_3$ plane. The u_x can be obtained by calculating the unit vector between P_1 and P_2.

$$u_x = \frac{P_2 - P_1}{||P_2 - P_1||} \tag{21}$$

Since the u_z is perpendicular to the $P_1P_2P_3$ plane, it can be obtained by the cross product of two vectors in this plane.

$$u_z = u_x \times u_y^{'} \tag{22}$$

where $u_y^{'}$ is the unit vector between P_2 and P_3.

$$u_y^{'} = \frac{P_3 - P_2}{||P_3 - P_2||} \tag{23}$$

The u_y can be obtained from the cross product between u_x and u_z.

$$u_y = u_z \times u_x \tag{24}$$

Finally the transformation matrix can be obtained, which maps the local (2D) frame to global (3D) frame by translating the P_1 to (0,0,0) point of global frame.

$$TM = \begin{bmatrix} u_x(x) & u_y(x) & u_z(x) & P_1(x) \\ u_x(y) & u_y(y) & u_z(y) & P_1(y) \\ u_x(z) & u_y(z) & u_z(z) & P_1(z) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{25}$$

This matrix transforms a coordinate in the local frame to the global frame.

$$P_{3D} = TM \cdot P_{2D} \tag{26}$$

Since the z value is zero in the local plane, it is expressed by P_{2D}. Therefore the 3D waypoints can be mapped to the 2D plane using the following equation.

$$P_{2D} = TM^{-1} \cdot P_{3D} \tag{27}$$

The second step is to apply a 2D G2CBS path smoothing algorithm to the 3 waypoints obtained by the equation (27). The final step is to map this value to the original global 3D space using the equation (26).

V. Simulation Results

The comparison of the performance of DC, C1CBC and G2CBS path with regard to fixed wing aircraft is shown in [29]. In this paper, only path smoothing results are compared.

A. 2D path planning navigating 3 waypoints

We first compared the three path smoothing methods connecting three waypoints in a 2D plane. Fig. 7 (a) shows the path smoothing results of using the Dubins Curve(DC), C1CBS and G2CBS algorithms. There is seemingly little differences among the three paths but the curvature of these paths are very different as can be seen in Fig. 7 (b). The curvature of DC and C1CBS are discontinuous at the junction points while the curvature of G2CBS is continuous over the whole path.

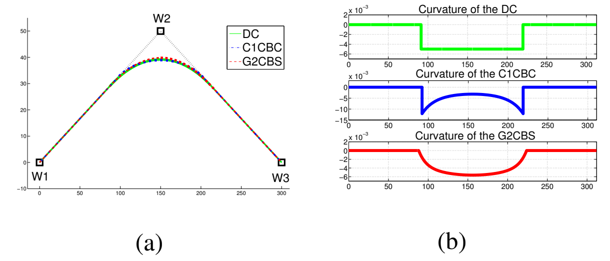
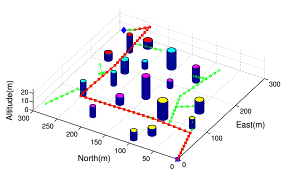


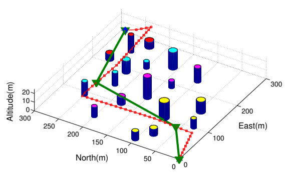
Fig. 7. Comparison of Path Smoothing Methods : (a) shows the path smoothing output of DC, C1CBC and G2CBS and (b) is the curvature of these three paths.

B. 3D Path Planning in Cluttered Environment

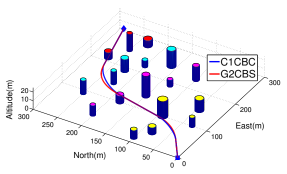
Fig. 8 shows the whole 3D path planning algorithms: Biased-Greedy RRT, path pruning and path smoothing. The altitude varies between 0 to 30m. The difference in 2D and 3D path planning is that the UAV can fly over the trees in 3D planning as can be seen in Figure 8. Results show that there are step changes of curvature at the joint in the case of C1CBC, while curvatures of G2CBS are continuous at these points.



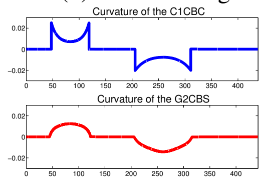
(a) RRT Path Planning



(b) Path Pruning



(c) Path Smoothing



(d) Curvature of the Path

Fig. 8. 3D Path Planning in Cluttered Environment: (a) is the path planning output from the Biased-Greedy RRT, (b) shows the path pruning output, (c) are the path smoothing results of C1CBS (blue line) and G2CBS (red line), (d) shows the curvature of C1CBS and G2CBS paths.

VI. CONCLUSIONS AND FUTURE WORKS

A UAV path planner has been developed using a modified RRT algorithm, combining biased sampling and the greedy extension of nodes. Since this path consists of piecewise linear segments, a smoothing process is needed.

A continuous curvature path smoothing algorithm has been developed satisfying nonholonomic constraints. The angle between two wapoints is the only required information for the generation of continuous curvature path. Therefore, the suggested algorithm is very simple and fast enough to be executed online path planning. Finally this algorithm is extended to 3-dimensional space.

We are currently doing system identification of our helicopter and will apply the algorithm to the real system.

VII. ACKNOWLEDGMENTS

This work is supported in part by the ARC Centre of Excellence programme, funded by the Australian Research Council (ARC) and the New South Wales State Government.

REFERENCES

[1] Choset, H., Lynch, K., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L., and Thrun, S., "*Principles of Robot Motion: Theory, Algorithms, and Implementations*," MIT Press, Cambridge, MA, 2005.

[2] E. Frazzoli, M. Dahleh and E.Feron , "Real-Time Motion Plannig for Agile Autonomous Vehicles", *American Control Conference*,Arlington, Virginia, June, 2001.

[3] J. Saunders, B. Call, A. Curtis, R. Beard and T. McLain, "Static and Dynamic Obstacle Avoidance in Miniature Air Vehicles", *AIAA Infotech@Aerospace*, Arlington, Virginia, Sep, 2005.

[4] M.Wzorek, P.Doherty, "Reconfigurable Path Planning for an Autonomous Unmanned Aerial Vehicle", *IEEE International Conference on Hybrid Information Technology*, Cheju, Korea, Nov. 2006.

[5] Jayesh N. Aminy, Jovan D. Boskovic and Raman K. Mehra, "A Fast and Efficient Approach to Path Planning for Unmanned Vehicle", *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Keystone, Colorado, August 2006.

[6] E. Magid, D. Keren, E. Rivlin and I. Yavneh, "Spline-Based Robot Navigation", *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, October, 2006,

[7] J. Barraquand and J.-C. Latombe, "On nonholonomic mobile robots and optimal maneuvering," *Revue dIntell. Artif*, vol. 3, no. 2, 1989.

[8] P. Jacobs and J. Canny, "Planning smooth paths for mobile robots," *IEEE Int. Conf. Robotics and Automation*, Scottsdale, AZ, May 1989.

[9] T. Fraichard, "Smooth trajectory planning for a car in a structured world," *IEEE Int. Conf. Robotics and Automation*, vol. 1, Sacramento, CA, Apr. 1991.

[10] L. E. Dubins, "On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal positions and tangents". *Amer. J. Math*, 1957.

[11] T. Liang and J. Liu, "Collision-free Path Planning for Mobile Robot Using Cubic Spiral", *IEEE International Conference on Robotics and Biomimetics*, Shenyang, China, August, 2004.

[12] T. Kito, J. Ota, R. Katsuiu, T.Mizuta, T. Arai, T. Ueyama and T. Nishiyama, "Smooth Path Planning by Using Visibility Graph-like Method", *IEEE International Conference on Robotics and Automation*, Taipei, Taiwan, September, 2003.

[13] T. Fraichard and A. Scheuer , "From Reeds and Shepps to Continuous-Curvature Paths", *IEEE Transactions on Robotics*, VOL. 20, NO. 6, December, 2004.

[14] W. Nelson, "Continuous-Curvature Paths for Autonomous Vehicles", *IEEE International Conference on Robotics and Automation*, Scottsdale, AZ, USA, 1989.

[15] O. Pinchard, A. LiCgeois and F. Pougnet, "Generalized Polar Polynomials for Vehicle Path Generation with Dynamic Constraints", *IEEE International Conference on Robotics and Automation*, Minneapolis, Minnesota, April, 1996.

[16] D.J. Walton, D.S. Meek, J.M. Ali, "Planar G2 transition curves composed of cubic Bezier spiral segments", *Journal of Computational and Applied Mathematics*, VOL 157, NO 2, August 2003.

[17] F. Lamiraux and J.-P. Laumond, "Smooth Motion Planning for Car-Like Vehicles", *IEEE Transactions on Robotics and Automation*,VOL. 17, NO. 4, August 2001.

[18] T. Berglund, H. Jonsson, and I. Sderkvist, "An obstacle-avoiding minimum variation b-spline problem", *International Conference on Geometric Modeling and Graphics*, 2003.

[19] J. Connors and G. Elkaim, "Analysis of a Spline Based, Obstacle Avoiding Path Planning Algorithm", *IEEE 65th Vehicular Technology Conference*, April, 2007.

[20] A. Piazzi, C. Bianco, and M. Romano , "η^3-Splines for the Smooth Path Generation of Wheeled Mobile Robots", *IEEE Transactions on Robotics*, VOL. 23, NO. 5, October, 2007.

[21] Anderson, E.P. Beard, R.W. McLain, "Real-time dynamic trajectory smoothing for unmanned air vehicles", *IEEE Transactions on Control System Technology*, VOL. 13, NO. 3, MAY, 2005.

[22] P. R. Chandler, S. Rasumussen, M. Pachter, "UAV cooperative path planning", *AIAA Guidance, Navigation, Control Conference*, Denver, CO, Aug. 2000.

[23] G. Yang and V. Kapila, "Optimal path planning for unmanned air vehicles with kinematic and tactical constraints", *IEEE Conf. Decision Control*, Las Vegas, 2002.

[24] C. Liu, W. Cheng and Z. Hong, "A Trajectory Generation for a Mobile Robot in 3D Path Planning", *IEEE International Conference on Automation and Logistics*, Jinan, China, August, 2007.

[25] D. J. Walton and D. S. Meek, "Curvature extrema of planar parametric polynomial cubic curves ", *Journal of Computational and Applied Mathematics*, VOL 134, Issues 1-2, September, 2001

[26] B. A. Barsky and T. D. DeRose, "Geometric continuity of parametric curves: constructions ofgeometrically continuous splines", *IEEE Computer Graphics and Applications*, VOL. 10, Issues 1, Jan, 1990.

[27] H. Deddi, H. Everett and S. Lazard, "Interpolation with curvature constraints", Dec, 2000. http://hal.inria.fr/docs/00/07/25/72/PDF/RR-4064.pdf

[28] Steven M. LaValle, "Rapidly-exploring random trees: A new tool for path planning", *TR 98-11, Computer Science Dept, Iowa State University*, 1998.

[29] Kwangjin Yang and Salah Sukkarieh, "Real-time continuous curvature path planning of UAVs in cluttered environment ", *5th International Symposium on Mechatronics and its Application*, Jordan, May 2008.

[30] Richard H.Bartels, John C. Beatty, Brian A. Barsky, *An Introduction to Splines for use in Computer Graphics and Geometric Modeling*, Morgan Ksufmann Publishers, Los Altos, California; 1986.