J Intell Robot Syst (2010) 57:101–122
DOI 10.1007/s10846-009-9359-1

An Efficient Path Planning and Control Algorithm for RUAV's in Unknown and Cluttered Environments

Kwangjin Yang · Seng Keat Gan · Salah Sukkarieh

Received: 1 February 2009 / Accepted: 1 August 2009 / Published online: 27 August 2009

Abstract This paper presents an efficient planning and execution algorithm for the navigation of an autonomous rotary wing UAV (RUAV) manoeuvering in an unknown and cluttered environment. A Rapidly-exploring Random Tree (RRT) variant is used for the generation of a collision free path and linear Model Predictive Control(MPC) is applied to follow this path. The guidance errors are mapped to the states of the linear MPC structure by using the nonlinear kinematic equations. The proposed path planning algorithm considers the run time of the planning stage explicitly and generates a continuous curvature path whenever replanning occurs. Simulation results show that the RUAV with the proposed methodology successfully achieves autonomous navigation regardless of its lack of prior information about the environment.

Keywords Dynamic path planning · Model predictive control · Rapidly-exploring random trees · Small-scale helicopter

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.

K. Yang (✉) · S. K. Gan · S. Sukkarieh
Australian Centre for Field Robotics, University of Sydney,
Sydney, New South Wales 2006, Australia
e-mail: k.yang@acfr.usyd.edu.au

S. K. Gan
e-mail: s.gan@acfr.usyd.edu.au

S. Sukkarieh
e-mail: salah@acfr.usyd.edu.au

1 Introduction

Navigation through an unknown environment has been of great interest in field robotics [1–6]. To complete the mission successfully in such environments, functionalities such as obstacle map building, dynamic path planning, and path following need to be implemented. 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 [7]. For UAV applications computational complexity is the most important requirement. If previously unknown obstacles are detected, the UAV has to replan the path in real time to avoid these obstacles. The computational time of deterministic and complete algorithms grows exponentially with the dimension of the configuration space, and hence these algorithms do not provide an adequate solution for real-time UAV path planning in unknown natural environments [8]. Recently, sample-based motion planning has gained much interest as an alternative to complete motion planning methods. Among them Rapidly-exploring Random Trees (RRT) has been used successfully to demonstrate efficient autonomous navigation in unknown environments [4, 6, 9, 11]. In this research RRT is used for the generation of a collision free piecewise linear path and a path smoothing algorithm is applied which satisfies curvature continuity and non-holonomic constraints.

Accurate path following control is important, especially when the RUAV flies in an environment cluttered with obstacles. In addition, it is also important to guarantee the smooth transition between the previous path and the new path when replanning occurs. However, it is difficult to design an accurate path following controller of a helicopter because it is a poorly damped system and its dynamics are highly nonlinear and unstable. A cascaded type controller is favored for helicopter control [12, 13]. The rule of thumb of this controller is that the inner loop control system must have sufficiently large bandwidth to track the commands that are generated from the outer loop guidance system. However, since the two systems are effectively coupled, stability and adequate performance of the combined systems can not be guaranteed [14].

Instead of this bandwidth separation scheme, an integrated guidance and control strategy such as Model Predictive Control (MPC) can achieve greater performance and stability. MPC employs an explicit model of the plant to predict the future output behavior by minimizing tracking error over a future horizon [15]. However standard linear MPC is not applicable for trajectory tracking since the position states are not included in the state space model. In this paper a novel explicit linear MPC path following control algorithm is used. The main advantage of this approach is that an explicit state feedback law avoids the need for on-line optimization [16]. Due to its low complexity and high reliability this method provides a good alternative for UAV control. To solve this path following control problem in a linear MPC framework we map the guidance errors to the states of the linear MPC using nonlinear kinematic equations. With this technique the guidance errors are converted to the reference values of the states which are controllable in a linear MPC formulation.

If new obstacles are detected and if they are located within the safety zone then the RRT path planner must replan the path to generate a collision free path. In this research the run time of the path planning algorithm is considered explicitly when replanning occurs. Furthermore the replanned path we propose also preserves the continuous curvature property which smoothes the transition of the UAV motion from the old path to the new path.

The paper is organized as follows: Section 2 describes the path planning algorithm, and the MPC path following control is presented in Section 3. Section 4 illustrates dynamic path replanning and simulation results are given in Section 5. Conclusion is presented in Section 6.

2 Path Planning

2.1 RRT Path Planning

RRT was first suggested in [17] as an alternative to complete path planning in high degree of freedom situations. The 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 are no collisions in the interval between x_{near} and x_{new} then the latter is added to the tree.

Figure 1 (Left) shows the path planning result of the RRT Algorithm. The thin lines are all trees generated by the algorithm and the thick line is the shortest path which connects the starting point and the target point. Even though RRT is an effective and computationally efficient tool for complex online motion planning, the solution produced is simply a random exploration of the space. We extend upon the standard approach to find a path that eliminates most extraneous nodes produced by the standard RRT algorithm.

Let the original path of nodes from start to goal point be denoted $\{x_1, \ldots, x_N\}$, 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

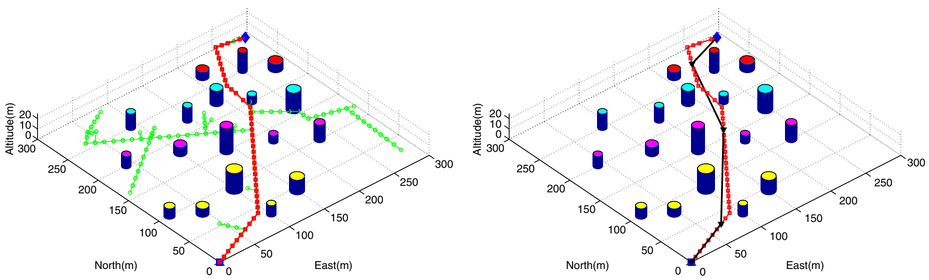

Fig. 1 (*Left*) Path planning using the RRT algorithm. (*Right*) Path pruning algorithm: the path consists of 43 nodes but after removing redundant waypoints, the number of nodes is reduced to only 3

until a complete path is generated. This method will remove unnecessary waypoints very quickly. Figure 1 (Right) shows the result of the path pruning algorithm applied to the initial RRT path. The path has 43 nodes between the starting and end points initially but the number of nodes is reduced to only 3 after the redundant waypoints are pruned.

2.2 Path Smoothing

The path in Fig. 1 (Right) is piecewise linear and not suitable for a RUAV with kinematic and dynamic constraints. To achieve curvature continuity the position, heading, and the curvature values of connecting linear paths must be the same at the joints.

Let the degree n Bézier curve with $n+1$ control points $(P_0, P_1, \cdots, P_n)$ be defined as [18]

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

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

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

The continuous curvature curve which satisfies the maximum curvature constraint can be obtained using the two spiral curves as shown in Fig. 2.

The first curve consists of following the four control points:

$$\begin{aligned} B_0 &= P_2 + d \cdot u_1, \quad & B_1 &= B_0 - g_b \cdot u_1, \\ B_2 &= B_1 - h_b \cdot u_1, \quad & B_3 &= B_2 + k_b \cdot u_d \end{aligned} \tag{3}$$

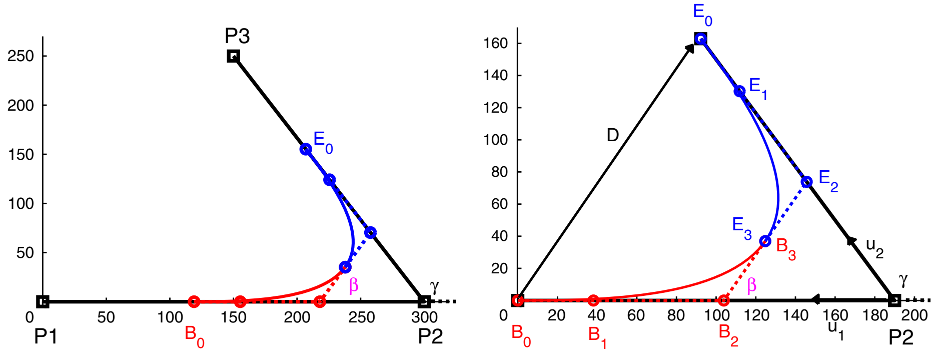

Fig. 2 (*Left*) G2CBS path smoothing result between two line segments. (*Right*) Curvature continuity condition to connecting two lines

The second curve consists of the following four control points:

$$\begin{aligned} E_0 &= P_2 + d \cdot u_2, \qquad & E_1 &= E_0 - g_e \cdot u_2, \\ E_2 &= E_1 - h_e \cdot u_2, \qquad & E_3 &= E_2 - k_e \cdot u_d \end{aligned} \tag{4}$$

where

$$\begin{aligned} h_b &= h_e = c_3 \cdot d, \\ g_b &= g_e = c_1 c_3 \cdot d, \\ k_b &= k_e = \frac{6 c_3 \cos\beta}{c_1 + 4} \cdot d \end{aligned} \tag{5}$$

Here 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, and d is a length between B_0 and P_2, and $\beta = \frac{\gamma}{2}$, and $c_1 = \frac{2}{5}(\sqrt{6} - 1)$, and $c_2 = 7.2364$, and $c_3 = \frac{c_1+4}{c_2+6}$.

The only design variable to generate a continuous curvature path is d as can be seen in Eqs. 3–5. If d is selected as Eq. 6 [19], then the path generates a G^2 path which satisfies the maximum curvature constraint.

$$d = \left(\frac{(c_1 + 4)^2}{54 c_3} \right) \cdot \frac{\sin\beta}{\kappa_{max} \cdot \cos^2\beta} \tag{6}$$

where κ_{max} is a maximum curvature value.

Figure 3 shows smoothed path result (Left) and the curvature of the path (Right).

3 Path Following Control

3.1 Helicopter Model

In this research a six DOF rigid body model suggested by Mettler et al. [20] was used. It expresses a small-size unmanned helicopter with system equations of first order. This model captures the nonlinear helicopter dynamics with very high accuracy and has been applied successfully in real applications [21].

The system equation is given as

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) \tag{7}$$

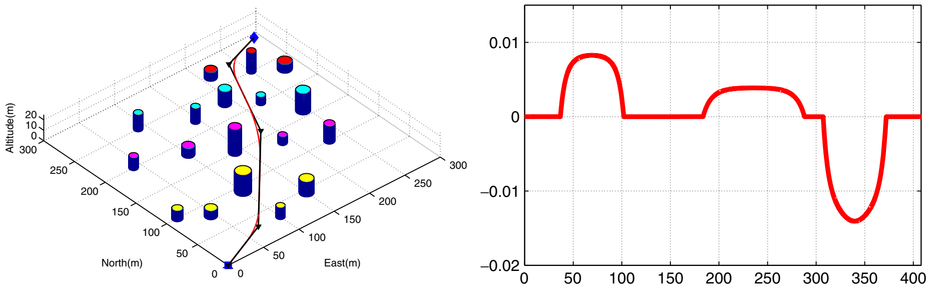

Fig. 3 (*Left*) Continuous curvature path smoothing. (*Right*) Curvature of the path

This equation can be separated into the dynamics and kinematics of the helicopter.

$$\begin{bmatrix} \dot{\mathbf{x}}^D \\ \dot{\mathbf{x}}^A \\ \dot{\mathbf{x}}^S \end{bmatrix} = \begin{bmatrix} \mathbf{A}\mathbf{x}^D + \mathbf{B}\mathbf{u} \\ C_B^S \omega \\ T_B^S \dot{\mathbf{x}}^B \end{bmatrix} \tag{8}$$

where

$$\mathbf{x}^D = [u,\ v,\ p,\ q,\ a,\ b,\ w,\ r,\ r_{fb}]^T$$
$$\mathbf{x}^A = [\phi,\ \theta,\ \psi]^T, \quad \mathbf{x}^S = [x^S,\ y^S,\ z^S]^T$$
$$\omega = [p,\ q,\ r]^T, \quad \mathbf{u} = [u_{lat},\ u_{lon},\ u_{col},\ u_{tr}]$$
$$C_B^S = \begin{bmatrix} 1, & \sin\phi\tan\theta, & \cos\phi\tan\theta \\ 0, & \cos\phi, & -\sin\phi \\ 0, & \sin\phi\sec\theta, & \cos\phi\sec\theta \end{bmatrix}$$
$$T_B^S = \begin{bmatrix} c\theta c\psi, & s\phi s\theta c\psi - c\phi s\psi, & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi, & s\phi s\theta c\psi + c\phi c\psi, & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta, & s\phi c\theta, & c\phi c\theta \end{bmatrix}$$

Here $\mathbf{x}^D$ represents the dynamic states and $\mathbf{x}^A$, $\mathbf{x}^S$ represents the kinematic states of the helicopter dynamics. The superscript S and B denote spatial and body coordinates, (u, v, w) are translational body velocities, (p, q, r) are body rotational roll, pitch, yaw rates, (ϕ, θ, ψ) are roll, pitch, yaw angles, (a, b) are the longitudinal, and lateral flapping angles of the main rotor, r_{fb} is a yaw rate gyro feedback state, and $c\theta, s\theta$ denotes $\cos\theta$ and $\sin\theta$.

There are four control inputs. u_{lon} and u_{lat} are cyclic pitch controls which are used to change the direction of the thrust vector controlling the pitch and roll, and x, y position and velocity. u_{col} is used for the generation of the thrust controlling the heave velocity and position. u_{tr} is used to counteract the torque effect of the main rotor and heading control.

3.2 PID Control

To judge the performance of the MPC implementation a PID control method is developed based on the work in [21]. The first step of designing the controller is to stabilize the attitude dynamics. Small scale helicopters are usually fitted with a stabilizer bar and an active yaw damping system. These two systems act as dynamic stability augmentations and have major effects on the vehicle response. The stabilizer bar acts as a lagged rate feedback in the pitch and roll axes and the angular rate gyro acts as a negative feedback of the helicopter heading rate. Therefore we can assume that small scale helicopters are naturally equipped with roll-pitch-yaw rate feedback controllers. Because of this property the attitude dynamics can be stabilized with only proportional feedback of the roll-pitch-yaw angle.

After stabilizing the attitude, the longitudinal and lateral velocity dynamics can be also stabilized by proportional control. The proportional control is sufficient for the heave velocity dynamics control because it is intrinsically stable.

Finally, position control also can be achieved by proportional and derivative control after stabilizing the velocity dynamics. To remove steady state error integral controls are added to the position control. The PID control laws for path following control are shown in Eq. 9.

$$\begin{aligned}
u_{lon} &= -K_\theta e_\theta - K_u e_u - K_x e_x - K_{Ix} \int e_x \\
u_{lat} &= -K_\phi e_\phi - K_v e_v - K_y e_y - K_{Iy} \int e_y \\
u_{col} &= -K_w e_w - K_z e_z - K_{Iz} \int e_z \\
u_{tr} &= -K_\psi e_\psi
\end{aligned} \tag{9}$$

3.3 Model Predictive Control

To formulate the explicit MPC structure consider the following linear system:

$$\begin{aligned}
\dot{\mathbf{x}} &= \mathrm{A}\mathbf{x} + \mathrm{B}\mathbf{u} \\
\mathbf{y} &= \mathrm{C}\mathbf{x} + \mathrm{D}\mathbf{u}
\end{aligned} \tag{10}$$

The states of the system at time t is obtained by the following convolution integral,

$$\mathbf{x}(t) = \Phi(t)\,\mathbf{x}(0) + \int_{\tau=0}^{t} \Phi(t-\tau) B\mathbf{u}(\tau)\, d\tau \tag{11}$$

where

$$\Phi(t) = e^{At} \tag{12}$$

The states at n prediction time step later is,

$$\mathbf{x}(t_n) = \Phi(t_n)\,\mathbf{x}(0) + \sum_{k=1}^{n} \Phi(t_n - t_k) \times A^{-1}\left[\Phi(t_k - t_{k-1}) - I\right] B\mathbf{u}(t_{k-1}) \tag{13}$$

By substituting this to the state space output equation in Eq. 10 yields,

$$\begin{aligned}
\mathbf{y}(t_n) =& C\Phi(t_n)\,\mathbf{x}(0) + \sum_{k=1}^{n} C\Phi(t_n - t_k) \\
& \times A^{-1}\left[\Phi(t_k - t_{k-1}) - I\right] B\mathbf{u}(t_{k-1}) + D\mathbf{u}(t_n)
\end{aligned} \tag{14}$$

This simplifies to,

$$\mathbf{y}(t_n) = \theta_n \mathbf{x}(0) + \sum_{k=1}^{n} K_{n,k}\mathbf{u}_{k-1} + D\mathbf{u}_n \tag{15}$$

where

$$\begin{aligned}
\theta_n &= C\Phi(t_n) \\
K_{n,k} &= C\Phi(t_n - t_k)\, A^{-1}\left[\Phi(t_k - t_{k-1}) - I\right] B
\end{aligned} \tag{16}$$

By combining $K_{i,j}$ and D, we can obtain the following equation

$$\mathcal{Y} = \Theta\mathbf{x}(0) + \Psi\mathcal{U} \tag{17}$$

where $\mathcal{Y} = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n]^T$ and $\Theta = [\theta_1, \theta_2, \cdots, \theta_n]^T$ and $\mathcal{U} = [\mathbf{u}_0, \mathbf{u}_1, \cdots, \mathbf{u}_n]^T$ and Ψ is,

$$\begin{bmatrix} K_{1,1} & D & 0 & 0 & \cdots & \cdots & 0 \\ K_{2,1} & K_{2,2} & D & 0 & \cdots & \cdots & 0 \\ K_{3,1} & K_{3,2} & K_{3,3} & D & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ K_{n,1} & K_{n,2} & K_{n,3} & \cdots & K_{n,n-1} & K_{n,n} & D \end{bmatrix}$$

For path following control, we define the following cost function,

$$J = \sum_{k=1}^{n} \left(\mathbf{y}_d(k)^T Q \mathbf{y}_d(k) + \mathbf{u}(k)^T R \mathbf{u}(k) \right) \tag{18}$$

where $\mathbf{y}_d(k) = \mathbf{y}_r(k) - \mathbf{y}(k)$. Here $\mathbf{y}_d(k)$ represents an error between the reference states $(\mathbf{y}_r)$ and the current helicopter states $(\mathbf{y})$. Q and R are positive definite diagonal weighting matrices. This cost function can be simplified as follows:

$$J = Y_d^T \mathcal{Q} Y_d + U^T \mathcal{R} U \tag{19}$$

The explicit form of the optimal control input which minimizes the above cost function can be obtained by the partial derivative of J with respect to control input, $\frac{\partial J}{\partial U} = 0$, which yields:

$$U = \left(\Psi^T \mathcal{Q} \Psi + \mathcal{R} \right)^{-1} \Psi^T \mathcal{Q} E \tag{20}$$

where E is the error between system future response of the RUAV and the reference path.

$$E = Y_r - \mathcal{Q} X \tag{21}$$

This MPC tracking controller minimizes the tracking errors and control inputs. However this linear MPC can not be directly applicable to path following control since the states of the MPC do not contain the position and heading states. Therefore to control the position and heading of the helicopter these guidance errors have to be appropriately mapped into the MPC controllable states. It is reasonable to select the u, v, w and r states to control x, y, z position and heading (ψ). We convert the guidance errors (e_x, e_y, e_z, e_ψ) to the reference states of the MPC (u_r, v_r, w_r, r_r). Due to the mapping of these guidance errors to the reference states of the MPC the position and heading errors can be optimally controlled.

3.4 Path Following Simulations

In this section, we compare the path following performance between the MPC algorithm presented here and the multi-loop PID control presented above. Initially the helicopter is hovering at position (0,0,5). The mission is to follow the path with a constant forward velocity of 5m/s. In this simulation a 3 prediction horizon is used in the MPC path following control since it is sufficient to get a accurate tracking performance.

Figure 4 shows the path tracking simulation result of the PID and the MPC control and Fig. 5 shows the guidance error. As we can see in Fig. 5, the position and altitude

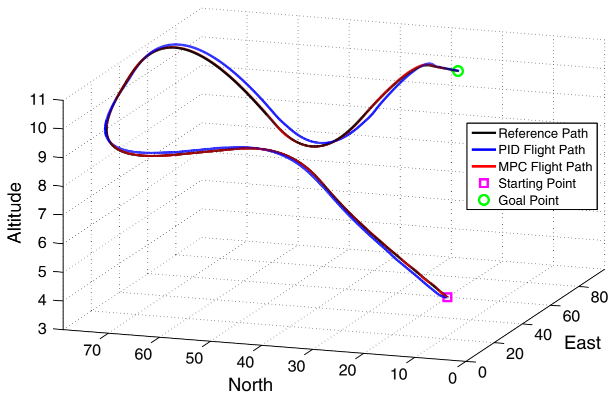

Fig. 4 3D path following results of the PID control and the MPC

errors of the MPC is almost zero but in the PID control case there exists relatively large errors. The heading error of MPC is also smaller than PID but the difference is not so big as the position errors. This is because the path has a smooth curvature profile and so the PID control can follow the reference heading well.

To compare the performance quantitatively the $\mathcal{L}_2$ norm of the guidance errors is calculated and presented in Table 1. The position error $\mathcal{L}_2$ norms of the PID control is more than 8 times larger than that of MPC with little difference in the heading error. Since the reference path is originally generated in a cluttered environment [19], a larger error increases the possibility of a collision. Therefore the maximum tracking error is also crucial in path following. Table 2 shows the $\mathcal{L}_\infty$ norms of the tracking error of the MPC and the PID control. Similar to the error $\mathcal{L}_2$ norms, the maximum error of PID control is more than 8 times larger than MPC except for the heading error.

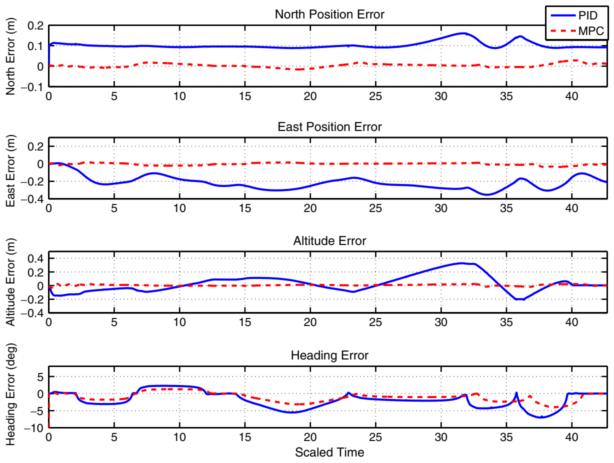

Fig. 5 Position and heading errors of the PID control and the MPC

Table 1 Comparison of the $\mathcal{L}_2$ norm path tracking errors

	North error(m)	East error(m)	Height error(m)	Heading error(rad)
PID control	4.79	10.59	5.81	2.41
MPC	0.60	0.81	0.68	1.86

4 Dynamic Path Planning

If new obstacles are detected within the safety zone the RRT path planner replans the path to generate a collision free path. Moreover, the replanned path preserves the continuous curvature property which makes for smooth transition of the UAV motion from the old path to the new path.

4.1 Decreasing Curvature Path Generation

The replanning can be triggered at an arbitrary point on the path whenever the helicopter detects new obstacles. Therefore the curvature of the path at that point may also have an arbitrary value. If the path planner does not consider the curvature at the replanning point then there will exist a curvature discontinuity between the old path and the new replanned path. This will cause a discontinuity in the centripetal acceleration which will result in rapid change of heading control and an increase in tracking error. To generate a continuous curvature path when replanning occurs a decreasing curvature path is proposed which monotonically increases or decreases the curvature to zero at the end of the path segment.

In [10] the condition proposed to generate a cubic Bézier spiral curve is as follows:

$$\frac{k}{h} \geq c_1, \qquad \frac{g}{h} \leq \frac{6\cos\theta}{\frac{g}{h}+4} \qquad \textit{and} \qquad 0 < \theta < \frac{\pi}{2} \tag{22}$$

where g, h, k are the lengths between two points and θ is the angle between the $\overline{P_0P_1}$ line and the $\overline{P_1P_3}$ line as is shown in Fig. 6.

Also the conditions to generate a decreasing curvature curve is suggested as follows:

$$P_{3x} > 0, \qquad P_{3y} \leq c_y \cdot \kappa_0 \cdot P_{3x}^2 \tag{23}$$

where $c_y = 0.4869$, and P_{3x}, P_{3y} are the x and y positions of P_3 and κ_0 is the curvature at P_0.

Table 2 Comparison of the $\mathcal{L}_\infty$ norm path tracking errors

	North error(m)	East error(m)	Height error(m)	Heading error(rad)
PID control	0.161	0.354	0.327	0.122
MPC	0.028	0.036	0.030	0.071

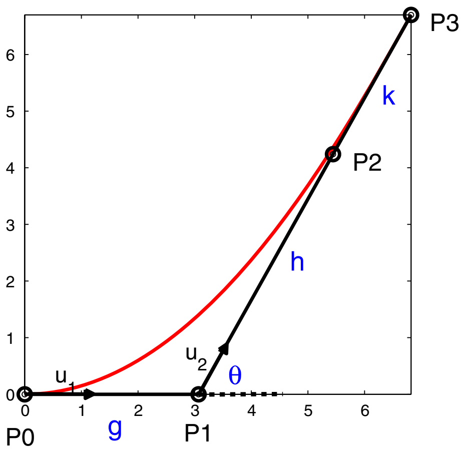

Fig. 6 Decreasing curvature path generation

To decide on the four control points for generating the decreasing curvature path both Eqs. 22 and 23 must be satisfied. The curvature at P_0 can be calculated using Eq. 24 then

$$\kappa_0 = \frac{|3\overrightarrow{g} \times (6\overrightarrow{h} - 6\overrightarrow{g})|}{|3\overrightarrow{g}|^3} = \frac{2h\sin\theta}{3g^2} \tag{24}$$

h can be expressed by κ_0, g and θ.

$$h = \frac{3\kappa_0 g^2}{2\sin\theta} \tag{25}$$

Therefore from the geometry of Fig. 6, θ, h, k can be expressed as follows:

$$\theta = \text{atan}\left(\frac{P_{3y}}{P_{3x} - g}\right), \tag{26}$$

$$h = \frac{3\kappa_0 \cdot g^2\sqrt{P_{3y}^2 + (P_{3x} - g)^2}}{2 \cdot P_{3y}} \tag{27}$$

$$k = \frac{(2P_{3y} - 3\kappa_0 \cdot g^2)\sqrt{P_{3y}^2 + (P_{3x} - g)^2}}{2 \cdot P_{3y}} \tag{28}$$

There are an infinite number of paths which satisfy Eqs. 22 and 23. however by making $k = c_1 \cdot h$ then the shortest path is selected as seen in Eq. 22.

$$k = c_1 \cdot h \tag{29}$$

Since P_{3y} can be obtained using the geometry in Fig. 6,

$$P_{3y} = (h + k)\sin\theta = (c_1 + 1) \cdot h\sin\theta \tag{30}$$

h and k can be expressed by P_{3y} as follows:

$$h = \frac{P_{3y}}{(c_1 + 1) \cdot \sin\theta}, \quad k = \frac{c_1 \cdot P_{3y}}{(c_1 + 1) \cdot \sin\theta} \tag{31}$$

If we substitute Eqs. 27 and 28 into Eq. 29,

$$\frac{k}{h} = c_1 = \frac{\frac{(2P_{3y} - 3\kappa_0 \cdot g^2)\sqrt{P_{3y}^2 + (P_{3x} - g)^2}}{2P_{3y}}}{\frac{3\kappa_0 \cdot g^2 \sqrt{P_{3y}^2 + (P_{3x} - g)^2}}{2P_{3y}}} = \frac{2P_{3y} - 3\kappa_0 \cdot g^2}{3\kappa_0 \cdot g^2} \tag{32}$$

g can be obtained as follows:

$$g = \sqrt{\frac{2P_{3y}}{3\kappa_0(c_1 + 1)}} \tag{33}$$

Then the only variable remaining in g, h, and k (Eqs. 31 and 33) is P_{3y}. We can chose any value smaller than $c_y \cdot \kappa_0 \cdot P_{3x}^2$ as shown in Eq. 23 but it is better to use small P_{3y} to minimize the path length. However if P_{3y} is too small the curvature will change very sharply shown in Fig. 7. To avoid this situation we choose P_{3y} which is proportional to the curvature.

$$P_{3y} = c_4 \kappa_0 P_{3y}^2 = \left(\frac{\kappa_0}{\kappa_{max}} c_y\right) \kappa_0 P_{3y}^2 \tag{34}$$

where κ_{max} is the maximum allowable curvature of the path. If the curvature at P_0 is large as in Fig. 6 then P_{3y} will have a large value which will tend to bend the path and if the curvature is small then P_{3y} will also be small which in turn generates an almost a straight line.

Finally, four control points generating a decreasing curvature path can be obtained as follows:

$$\begin{aligned} P_0 &= (0, 0), & P_1 &= P_0 + g \cdot u_1, \\ P_2 &= P_1 + h \cdot u_2, & P_3 &= P_2 + k \cdot u_2 \end{aligned} \tag{35}$$

where

$$g = \sqrt{\frac{2P_{3y}}{3\kappa_0(c_1 + 1)}}, \quad h = \frac{P_{3y}}{(c_1 + 1) \cdot \sin\theta}, \quad k = c_1 h, \quad \mathit{and} \quad P_{3y} = c_4 \kappa_0 P_{3x}^2 \tag{36}$$

Therefore if P_{3x} is determined then the other parameters can be determined based on P_{3x} as can be seen in Eqs. 35 and 36. If the computational time of the path

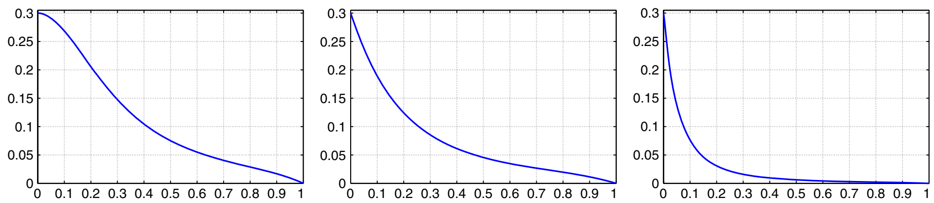

Fig. 7 Curvature comparison: the curvature at P_0 is 0.3 and c_4 is chosen with different values; $c_4 = 0.48$ (*Left*), $c_4 = 0.20$ (*Middle*), $c_4 = 0.05$ (*Right*)

replanning is specified as T_{rp} and the helicopter flies with Vm/s then the distance which helicopter flies during the replanning time is

$$D_{rp} = V \cdot T_{rp} \tag{37}$$

Therefore P_{3x} must be decided which makes the length of the planned path D_{rp}. Generally, a numerical method is used to calculate the path length in Bézier curves [22] but Gravesen suggested an algorithm which can approximate the length of the Bézier curves in [23].

$$L_{bc} = \frac{2 \cdot L_c + (n-1) \cdot L_p}{n+1} \tag{38}$$

where L_c is the length between P_0 and P_3, L_p is the sum of all polygon lengths, and n is the order of the Bézier curve. In our application $n = 3$ since the cubic Bézier curves is used and $L_p = g + h + k$. Therefore, Eq. 38 can be modified

$$L_{bc} = \frac{L_c + g + h + k}{2} \tag{39}$$

If we substitute the values obtained from Eqs. 36 into 39 then it can also be modified to:

$$L_{bc} = \frac{\sqrt{P_{3x}^2 + P_{3y}^2} + \sqrt{\frac{2P_{3y}}{3(c_1+1)\cdot\kappa_0}} + \frac{P_{3y}}{\sin\theta}}{2} \tag{40}$$

Since $P_{3y} = c_4 \cdot \kappa_0 \cdot P_{3x}^2$ then

$$\tan\theta = \frac{P_{3y}}{P_{3x} - g} = \left(\frac{c_4}{1 - \sqrt{\frac{2c_4}{3(c_1+1)}}} \right) \kappa_0 P_{3x} = c_5 \kappa_0 P_{3x} \tag{41}$$

and therefore

$$\sin\theta = \frac{c_5 \kappa_0 P_{3x}}{\sqrt{1 + (c_5 \kappa_0 P_{3x})^2}} \tag{42}$$

Finally Eq. 40 becomes:

$$L_{bc} = \frac{\sqrt{1 + (c_4 \kappa_0 P_{3x})^2} + \sqrt{\frac{2c_4}{3(c_1+1)}} + \frac{c_4 \sqrt{1+(c_5\kappa_0 P_{3x})^2}}{c_5}}{2} P_{3x} \tag{43}$$

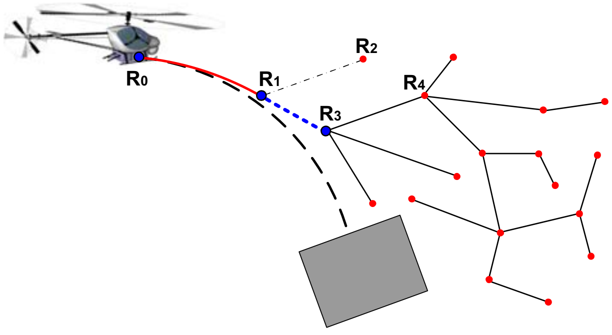

Fig. 8 Path replanning: if the RRT path planner starts the replanning at R_1, it may cause a heading discontinuity at this point even though it satisfies the position and curvature continuity. However, if the heading at R_1 is stored and another linear path is created which holds the same heading as R_1, the continuous curvature path can be generated

Table 3 Flight time comparison

	Known E	Partially known E	Unknown E
Flight time (seconds)	36.8	38.0	38.2
Flight distance (m)	184.2	189.9	191.2

The goal is to decide on a value for P_{3x} which makes $L_{bc} = D_{rp}$. It is difficult to find the exact solution to Eq. 43. To solve this problem we assume that P_{3x} located in inside the square-root has a constant value D_p. Using this assumption Eq. 43 becomes

$$P_{3x} = f(V, T_{rp}, \kappa_0) \approx \frac{2D_{rp}}{\sqrt{1 + (c_4\kappa_0 D_p)^2} + \sqrt{\frac{2c_4}{3(c_1+1)}} + \frac{c_4\sqrt{1+(c_5\kappa_0 D_p)^2}}{c_5}} \tag{44}$$

Since Eq. 43 is highly nonlinear the assumption in Eq. 44 is not valid. In our application the helicopter flies slower than 10m/s and the allowable maximum curvature of the helicopter is 0.25. The maximum run time for path planning is 1 second. Within these constraints the D_p function can be approximated as follows:

$$D_p = D_{rp}\left[e^{-\kappa_u} + (0.9 - 0.12 \cdot D_{rp}) \cdot \kappa_u\right] \tag{45}$$

where $D_{rp} = V \cdot T_{rp}$ is the distance which the helicopter flies during the run time of path generation and κ_u is an unsigned curvature. Therefore D_p changes with respect to the helicopter velocity, the replanning run time, and the curvature at the point which the replanning is triggered. If P_{3x} is decided using Eq. 44 which considers the helicopter velocity, the replanning run time, and the curvature of the path, then the decreasing curvature path is generated from Eqs. 35 and 36.

Figure 8 shows the proposed dynamic path replanning algorithm. When the helicopter flies at R_0, a new obstacle is detected located in the reference path(dashed line). The helicopter starts the replanning at R_0 point. First the decreasing curvature

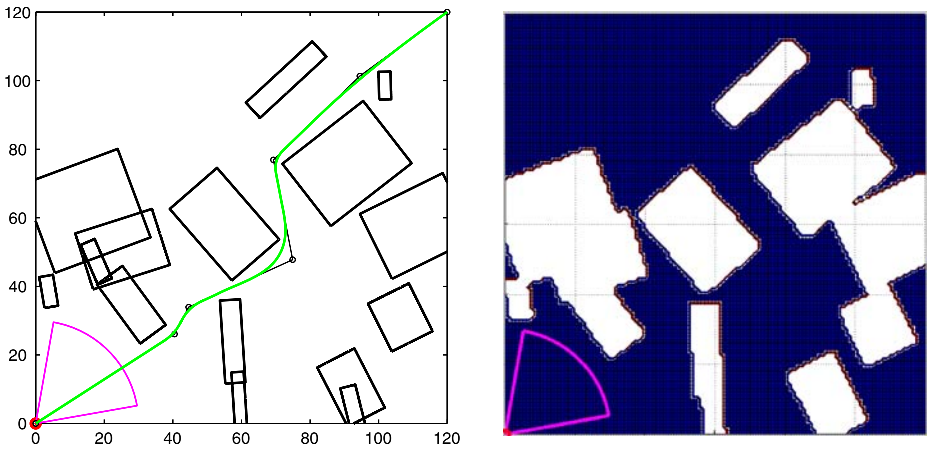

Fig. 9 On-line dynamic planning and control in known environment: start

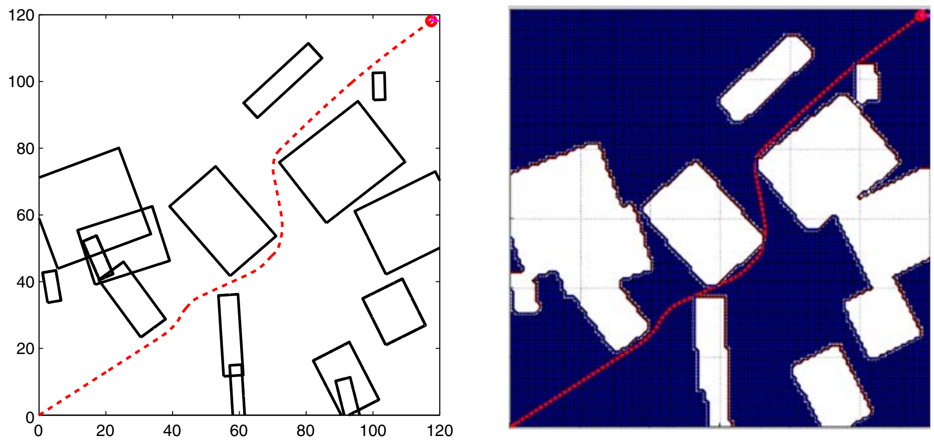

Fig. 10 On-line dynamic planning and control in known environment: finish (36.84 s)

path is generated which ends at R_1. If the RRT path planner starts the path replanning at R_1, it may cause a heading discontinuity at this point even though it satisfies the position and curvature continuity (They share the same position value at R_1 and the curvature of both path are zero at this point but the headings are different). To solve this problem, the heading at R_1 is stored and another linear path is created which holds the same heading as R_1. R_3 represents this linear line. The role of R_3 is to offer an length for path smoothing which preserves the heading continuity. Therefore even though the RRT path planner starts the replanning at R_3, the new smooth path will start at R_1.

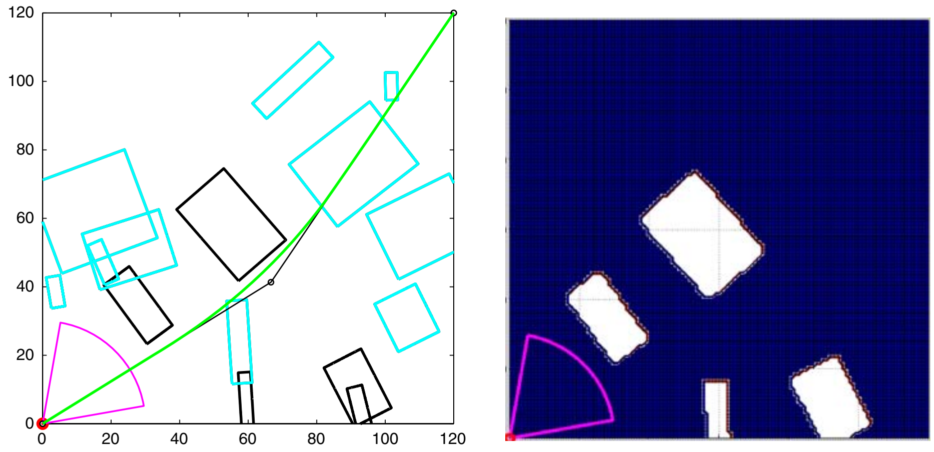

Fig. 11 On-line dynamic planning and control in partially known environment: start

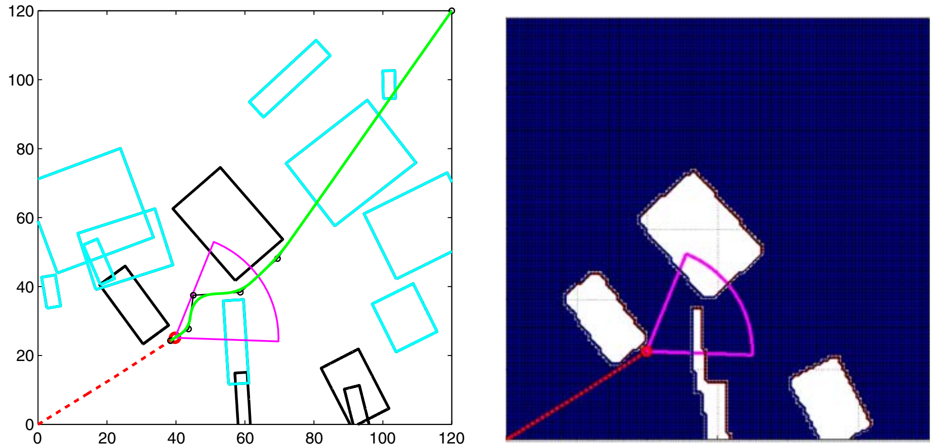

Fig. 12 On-line dynamic planning and control in partially known environment: 11 s

5 Simulation Experiments

In this section we will demonstrate the effectiveness of our on-line path planning algorithm for three different scenarios: a fully known environment; partially known environment; and a totally unknown environment. The helicopter flies with 5m/s velocity and 15 obstacles are generated randomly. An obstacle map is built by using a laser scanner as in [24, 25]. The laser scanner has a 70° field of view and returns the range information within 30m. The obstacle size in the map is larger than the real size since a safety margin is included when generating the map. Table 3 shows the flight time and distance comparison among three different environments. It takes the shortest flight time and distance in the fully known environment case and it takes the longest time and distance in the totally unknown environment case.

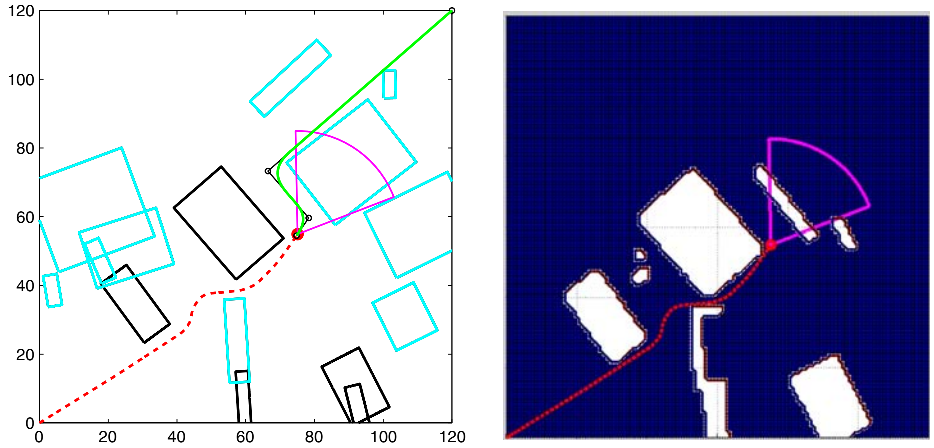

Fig. 13 On-line dynamic planning and control in partially known environment: 21 s

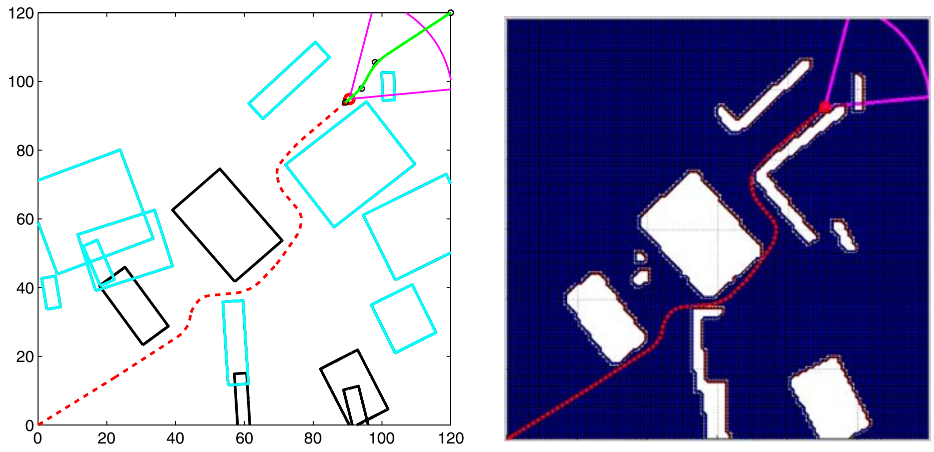

Fig. 14 On-line dynamic planning and control in partially known environment: 31 s

5.1 Fully Known Environment

In this scenario all obstacle locations are known before flight so the helicopter holds perfect information about the environment. Figure 9 (left) shows the path planning result: the black line represents a polygonal path and the green line represents a continuous curvature path. Figure 9 (right) shows an obstacle map of the environment. The obstacle size in the map is larger than the real obstacle as already mentioned based on the safety margin applied.

Figure 10 shows the flight path (dotted line) and the map. The helicopter navigates successfully in the cluttered environment without collision. The obstacle map (right) is not changed since the helicopter already has a perfect map of the environment before flight.

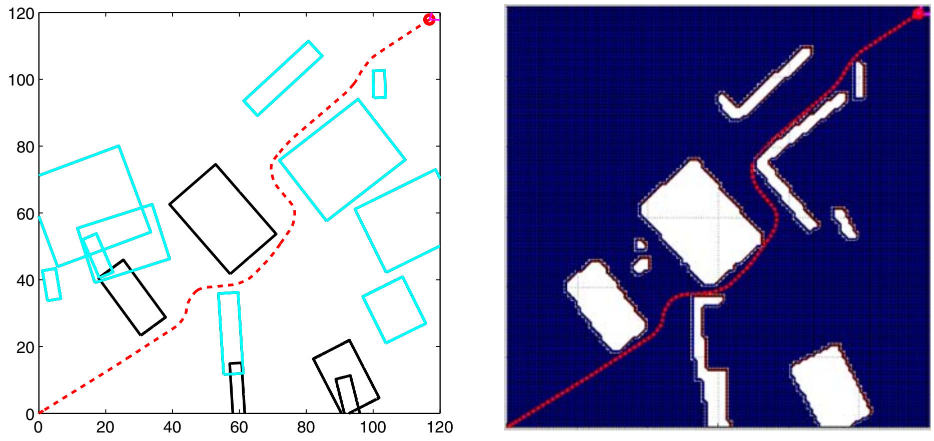

Fig. 15 On-line dynamic planning and control in partially known environment: finish (37.98 s)

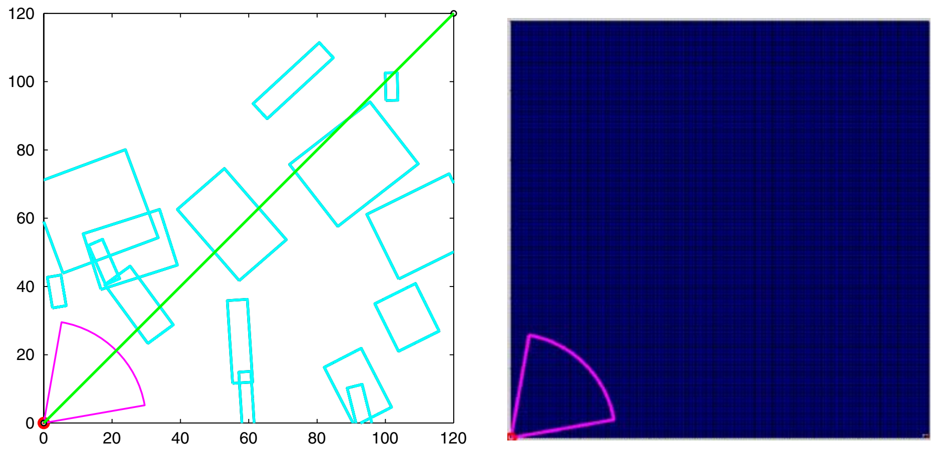

Fig. 16 On-line dynamic planning and control in unknown environment: start

5.2 Partially Known Environment

In this case the helicopter only has information about five obstacles (black color) and there are unknown ten obstacles (crayon color). Figure 11 shows the initial path (left) and the map (right). Since the helicopter generates a path based on partial information then collisions will occur.

Figure 12 shows the path and the obstacle map when the flight time is 11 seconds. A new obstacle located at (60,30) is detected and using the updated map the RRT path planner replans the path. It generates a path which passes through the narrow passage since the shortest length metric is used to generate a path.

Figure 13 shows the path and the obstacle map when the flight time is 21 seconds. Even though the obstacle located at (90,75) is large, the size of this obstacle in the

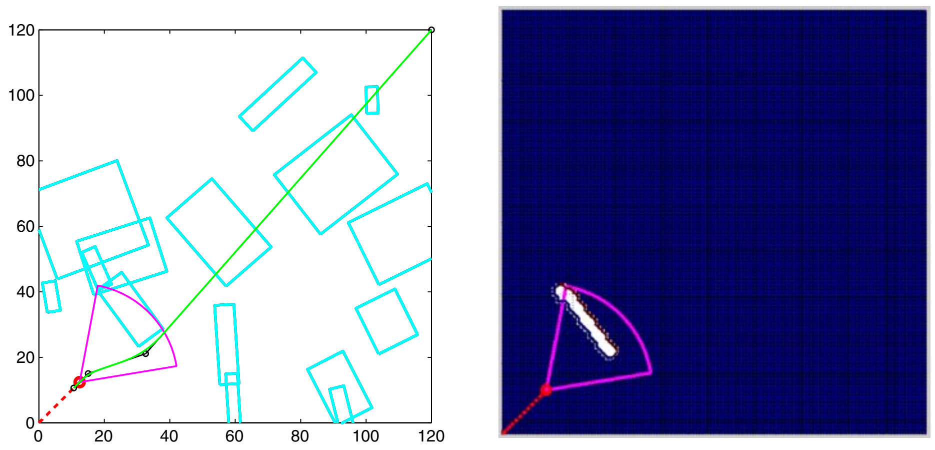

Fig. 17 On-line dynamic planning and control in unknown environment: 5 s

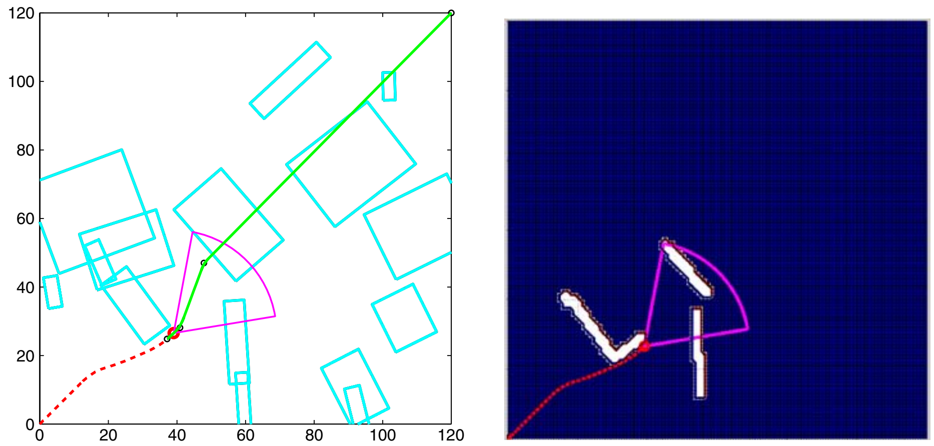

Fig. 18 On-line dynamic planning and control in unknown environment: 11 s

map is small due to the occlusion of the sensor. However the planner generates a collision free path successfully based on this map.

Figure 14 shows the path and the obstacle map at 31 seconds. The sensor detects the last obstacle at (105,95) and replans the path again to avoid the obstacle.

Figure 15 shows the path and the obstacle map when the helicopter arrives at the target point. The final path has a similar shape to that of the fully known environment except for a slight change near (50,35).

5.3 Totally Unknown Environment

In this case, the helicopter has no information about the obstacles when it starts its mission. Figure 16 shows the initial path and the obstacle map. Since the helicopter

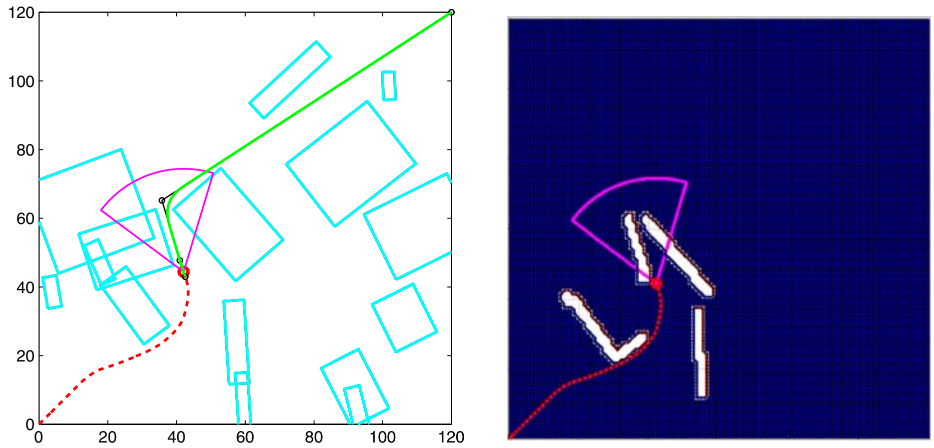

Fig. 19 On-line dynamic planning and control in unknown environment: 15 s

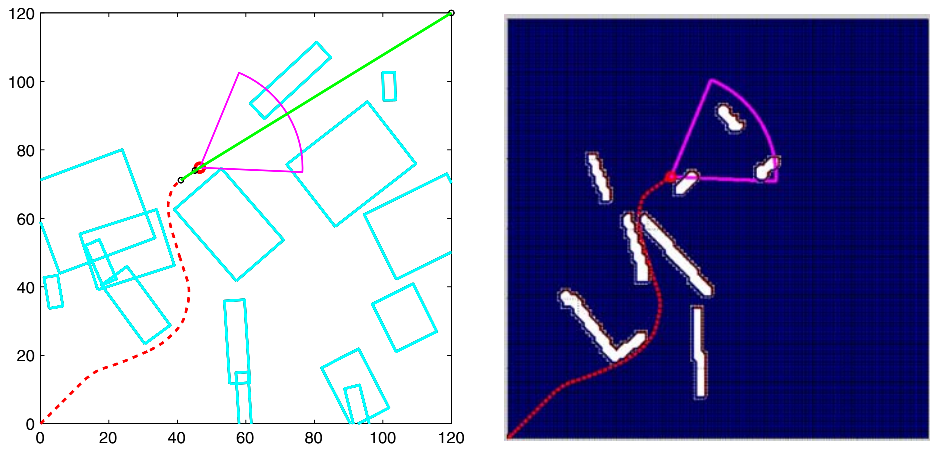

Fig. 20 On-line dynamic planning and control in unknown environment: 22 s

assumes that there are no obstacles in the environment it initially generates a straight line path.

Figure 17 shows the path and the obstacle map when the flight time is 5 seconds. The previous two cases have the information about the obstacle located in (30,30) so the initial heading is about 30° but the initial heading of the unknown environment case is 45° which causes replanning at 5 seconds.

Figure 18 shows the path and the obstacle map when the flight time is 11 seconds. The previous two cases have the information about the obstacle located in (60,60) so they plan the path which pass the narrow passage since it is shortest path based on their map. However in this case the helicopter senses the obstacle located in (60,60) gradually when it approaches this obstacle. In this case the shortest path is to travel to the left side of the obstacle when the replanning occurs.

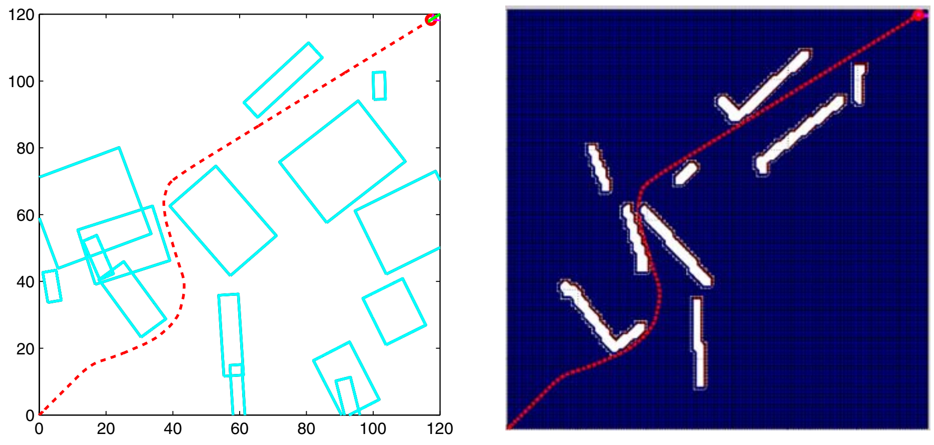

Fig. 21 On-line dynamic planning and control in unknown environment: finish (38.24 s)

Figure 19 shows the path and the obstacle map when the flight time is 15 seconds. The flight route of the helicopter is different from the previous two cases. The planner selects the left side of the obstacle located in (60,60).

Figure 20 shows the path and the obstacle map when the flight time is 22 seconds. Even though the helicopter detects 3 obstacles in front, it does not change the path since they do not cause a collision.

Figure 21 shows the path and the obstacle map when the helicopter arrives at the target point. It shows that the RRT planner successfully generates a collision free path even though it has no a-priori information about the environment.

6 Conclusions

An on-line dynamic path planning algorithm for a UAV navigating in an unknown cluttered environment is presented. The RRT path planner generates a collision-free path and cubic Bézier spiral curves are used to smoothen the path guaranteeing the curvature continuity. This path can then be tracked accurately by the novel MPC path following controller presented in this paper. We also consider the run time of the path planning process explicitly when replanning occurs. The proposed method generates a curvature continuous path whenever the path replanning is triggered. The simulation results show that the proposed method can achieve autonomous navigation successfully regardless of the level of a-priori information about environment.

Acknowledgements 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. Stentz, A., Hebert, M.: A complete navigation system for goal acquisition in unknown environments. Auton. Robots **2**(2), 127–145 (1995)
2. Brock, O., Khatib, O.: High-speed navigation using the global dynamic window approach. In: IEEE International Conference on Robotics and Automation, Detroit, Michigan (1999)
3. Ersson, T., Hu, X.: Path planning and navigation of mobile robots in unknown environments. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, Maui, Hawaii (2001)
4. Bruce, J., Veloso, M.: Real-time randomized path planning for Robot navigation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, Lausanne, Switzerland (2002)
5. Philippsen, R., Kolski, S., Macek, K., Siegwart, R.: Path planning, replanning, and execution for autonomous driving in urban and offroad environments. In: Proceedings of the ICRA Workshop on Planning, Perception and Navigation for Intelligent Vehicles (2007)
6. Griffiths, S., Saunders, J., Curtis, A., Barber, B., McLain, T., Beard, R.: Maximizing miniature aerial vehicles. IEEE Robot. Autom. Mag. **13**, 34–43 (2006)
7. Choset, H., Lynch, K., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L., Thrun, S.: Principles of Robot Motion: Theory, Algorithms, and Implementations. MIT, Cambridge (2005)
8. Frazzoli, E., Dahleh, M., Feron, E.: Real-Time Motion Plannig for Agile Autonomous Vehicles. American Control Conference, Arlington, Virginia (2001)
9. Espinoza, J., Sánchez, A., Osorio, M.: Exploring unknown environments with randomized strategies. In: MICAI 2006: Advances in Artificial Intelligence (2006)
10. Walton, D.J., Meek, D.S., Ali, J.M.: Planar G2 transition curves composed of cubic Bezier spiral segments. J. Comput. Appl. Math. **157**(2), 453–476 (2003)
11. Wzorek, M., Doherty, P.: Reconfigurable path planning for an autonomous unmanned aerial vehicle. In: IEEE International Conference on Hybrid Information Technology (2006)
12. Sanders, C.P., DeBitetto, P.A., Feron, E., Vuong, H.F., Leveson, N.: Hierarchical control of small autonomus helicopters. In: IEEE Conference on Decision and Control, Tempa, Florida (1998)

13. Muratet, L., Doncieux, S., Briere, Y., Meyer, J.A.: A contribution to vision-based autonomous helicopter flight in urban environments. Robot. Auton. Syst. **50**, 195–209 (2005)
14. Silvestre, C., Pascoal, A., Kaminer, I.: On the design of gain-scheduled trajectory tracking controllers. Int. J. Robust Nonlinear Control **12**(9), 797–839 (2002)
15. Kim, H., Shim, D., Sastry, S.: Nonlinear model predictive tracking control for rotorcraft-based unmanned aerial vehicles. In: American Control Conference, Anchorage, AK (2002)
16. Grancharova, A., Johansen, T.A., Kocijan, J.: Explicit model predictive control of gas-liquid separation plant via orthogonal search tree partitioning. Comput. Chem. Eng. **28**, 2481–2491 (2004)
17. LaValle, S.M.: Rapidly-exploring random trees: a new tool for path planning. TR 98-11, Computer Science Dept, Iowa State University (1998)
18. Bartels, R.H., Beatty, J.C., Barsky, B.A.: An Introduction to Splines for use in Computer Graphics and Geometric Modeling. Morgan Ksufmann Publishers, Los Altos, California (1986)
19. Yang, K., Sukkarieh, S.: 3D smooth path planning for a UAV in cluttered natural environments. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, 22–26 September 2008
20. Mettler, B., Tischler, M.B., Kanade, T.: System identification of small-size unmanned helicopter dynamics. In: American Helicopter Society 55th Forum, Montreal, Quebec (1999)
21. Shim, D.H., Kim, H.J., Sastry, S.: Hierarchical control system synthesis for rotorcraft-based unmanned aerial vehicles. In: AIAA Guidance, Navigation and Control Conference, Denver (2000)
22. Hwang, J., Arkin, R., Kwon, D.: Mobile robots at your fingertip: Bezier curve on-line trajectory generation for supervisory control. In: IEE/RSJ Int. Conference on Intelligent Robots and Systems, Las Vegas, Nevada (2003)
23. Gravesen, J.: Adaptive subdivision and the length and energy of Bezier curves. Comput. Geom. **8**(1), 13–31 (1997)
24. Shim, D., Chung, H., Sastry, S.: Conflict-free navigation in unknown urban environments. IEEE Robot. Autom. Mag. **13**, 27–33 (2006)
25. Scherer, S., Singh, S., Chamberlain, L., Elgersma, M.: Flying fast and low among obstacles: methodology and experiments. Int. J. Rob. Res. **27**(5), 549–574 (2008)