DESIGN OF PATH PLANNING AND TRACKING CONTROL OF QUADROTOR

YI GAO, RUI LI, YINGJING SHI AND LI XIAO

School of Automation Engineering
University of Electronic Science and Technology of China
Chengdu 611731, China

(Communicated by Bin Li)

Abstract. In this paper, we first design a motion planning system based on the Batch Informed Trees (BIT*) algorithm for quadrotor and a linear model predictive control (LMPC) is applied to solve the path tracking problem for a quadrotor. BIT* algorithm is used to plan a barrier-free trajectory quickly in an obstructed environment. Then we apply linear model predictive control for the full state quadrotor system model to track the generated trajectory. Finally, the BIT* algorithm simulation case is presented using RVIZ visual interface and some simulation cases are presented using MATLAB / Simulink. The results demonstrate the capability and the effectiveness of the control strategy in fast path tracking and the quadrotor stability, while the desired performance is achieved.

1. Introduction. In recent years, since the unmanned quadrotor helicopter has agility and good ability, they have attracted the attention of many researchers and are involved in a lot of applications such as objective tracking [16], agricultural services [10], and marine operations [5]. However, the environments of these scenarios are sometimes obstacle-cluttered. The trajectories which quadrotors followed should be safe, smooth, and dynamic feasible. In addition, a precise path tracking controller is needed to achieve good tracking performance.

Path planning algorithm has been widely investigated in literature including [4, 3, 13, 8, 2, 6, 20] and so on, which mainly involves two types: Graph-search and sampling-based methods. Graph-based methods, such as Dijkstra algorithm [8], D* algorithm [2], and A*algorithm [6], using graph traversal to search for collision-free paths, are mainly applied in two-dimensional space. Graph traversal increases the quality of the solution but also increases the computational effort to find it. This would greatly increase the computation time, and a large number of search results would be useless for robot motion. Sampling-based methods, such as Probabilistic
Roadmaps (PRM) algorithm [20], Rapidly-exploring Random Trees (RRT) algorithm [11], and RRT* algorithm [12], are widely used in different dynamic models. These methods have higher search efficiency when the map dimension is higher or the size is larger, and conveniently carry on the real-time on-line application. BIT* algorithm is able to combine the efficient search of algorithms, such as A* algorithm, with the anytime scalability of sampling-based algorithms, such as RRT* algorithm. As a sampling-based planning algorithm, BIT* algorithm performs better than graph-based search algorithm in a high dimensional environment [9]. Adding heuristic ordered search makes the convergence rate faster than the sampling-based algorithm. Therefore, the BIT* algorithm is suitable for the front-end path search of quadrotor planning.

On the other hand, researchers put forward a variety of control methods and control concepts in the field of flight control [18]. However, most control techniques, such as PID [15], linear quadratic methods [17], robust linear control [1], adaptive control [21], iterative learning control [14], neural networks [7], and sliding mode [19], apply the input constraints as post-facto, while model predictive control imposes input constraints as a part of the control synthesis. Since the quadrotor is a complex system with characteristics of underdrive, nonlinear, strong coupling, and constrained input, it is difficult to design the control algorithm. At each sampling instant, model predictive control obtains the current control state by optimizing a prespecified objective function over a finite horizon. Model predictive control can effectively integrate the state, input constraints, the goal state, and the trajectory constraints (equality and inequality) into the MPC framework as part of the comprehensive control, and lower requirement of the model. It can overcome the influence of various uncertain factors, and has a good control effect and robustness. Therefore, model predictive control has a great advantage in dealing with nonlinearly constrained problems and is more suitable for practical complex systems with time variability and uncertainty, such as quadrotors.

In this paper, we consider a real-time motion planning system and a precise path tracking controller for quadrotor autonomous navigation in unknown environments. First, the Batch Informed Trees algorithm is employed to find a barrier-free trajectory in an obstructed environment, where batches of samples are used to perform an ordered search on a continuous planning domain while maintaining anytime performance. Then, LMPC is applied for the path tracking controller of a quadrotor. Finally, we verify the system’s path planning and tracking performance in the simulation system, which illustrates that the BIT* algorithm finds better solutions with faster anytime convergence towards the optimum and the LMPC scheme has an effective performance on tracking reference paths.

The remainder of this paper is organized as follows: Section 2 presents the system structure. Section 3 presents an initial theoretical analysis of the motion planning system. Section 4 presents the model of the quadrotor dynamics. Section 5 presents a path tracking controller based on model predictive control, including the system model linearization and the establishment of objective functions. Section 6 presents the experimental results in detail. Section 7 provides a conclusion.

2. Problem formulation. The real-time trajectories which quadrotors followed in obstacle-cluttered environments should be safe, smooth and dynamic feasible. A precise path tracking controller should also be designed to achieve good tracking performance. First, we design a motion planning system based on the BIT*
algorithm to generate these qualified trajectories in real-time. Note that these discrete path points are relatively sparse, not smooth, and do not contain any time information. Then these sparse path points are changed into continuous smooth curves through the minimum-snap polynomial trajectory. Finally, we apply LMPC to design the rotational dynamics control subsystem and translational dynamics control subsystem of quadrotor separately. The system structure diagram of this paper is shown in Fig. 1.

3. **Motion planning navigation algorithm.** In this section, in order to realize the real-time collision-free path planning of a quadrotor in an unknown environment, we design a real-time motion planning system. The system is divided into two parts. First, we design Batch Informed Trees (BIT*) algorithm for the quadrotor to quickly find a non-collision continuous path from the starting point to the endpoint in the map. Then, we consider the dynamics of the quadrotor, building a Minimum Snap trajectory optimization scheme to ensure the safety of the trajectory by iteratively inserting intermediate points.

3.1. **BIT* path planning.** BIT* algorithm uses batches of samples to perform an ordered search on a continuous planning domain while maintaining anytime performance. Multiple samples are generated in each batch and iterated multiple times. By processing samples in batches, its search can be ordered around the minimum solution proposed by a heuristic. The batch ends and the extension stops when it finds a solution, then add a new batch of samples and start searching again. By processing multiple batches of samples, it converges asymptotically towards the global optimum with anytime resolution. This is done efficiently by using incremental search techniques to maintain asymptotic optimality.

BIT* algorithm is presented in Algorithm 1. $X_{obs}, X_{free}, X_{goal}$ represent obstacle space, obstacle-free space, and target space, respectively. Therefore, the programming problem is described as finding a sequence of states, $x(t) \in X_{free}$. The algorithm starts with a given initial state $x_{start}$ and ends with the goal state $x_{goal}$ in time interval $t \in [0, \tau]$. Then solve the optimal problem of the minimized cost function to find the path, for which the tree $T = (V, E)$ is constructed and maintained. Here, $V$ represents the set of state vertices from $X_{free}$ and $E \subseteq V \times V$ is the edges that connect the vertices.
Algorithm 1 BIT* Algorithm ($x_{start} \in X_{free}; x_{goal} \in X_{goal}$)

1: $V \leftarrow \{x_{start}\}; E \leftarrow \emptyset; X_{samples} \leftarrow \{x_{goal}\};$
2: $Q_E \leftarrow \emptyset; Q_V \leftarrow \emptyset; r \leftarrow \infty;$
3: (BatchCreation)
4: repeat
5:   if $Q_E \equiv \emptyset$ and $Q_V \equiv \emptyset$ then
6:     Prune ($g_\tau(x_{goal}))$; (GraphPruning)
7:     $X_{samples} \leftarrow \text{Sample}(m, g_\tau(x_{goal})$);
8:     $V_{old} \leftarrow V$;
9:     $Q_V \leftarrow V$;
10: $r \leftarrow \text{radius}(|V| + |X_{samples}|)$;
11: end if
12: (EdgeSelection)
13: while $\text{BestQueueValue}(Q_V) \leq \text{BestQueueValue}(Q_E)$ do
14:     ExpandVertex($\text{BestInQueue}(Q_V)$); (VertexExpansion)
15: end while
16: $(v_m, x_m) \leftarrow \text{BestInQueue}(Q_E)$;
17: (EdgeProcessing)
18: $Q_E \leftarrow \{(v_m, x_m)\}$
19: if $g_\tau(v_m) + \hat{c}(v_m, x_m) + \hat{h}(x_m) < g_\tau(x_{goal})$ then
20:   if $\hat{g}(v_m) + c(v_m, x_m) + \hat{h}(x_m) < g_\tau(x_{goal})$ then
21:     if $g_\tau(v_m) + c(v_m, x_m) < g_\tau(x_m)$ then
22:       if $x_m \in V$ then
23:         $E \leftarrow \{(v, x_m) \in E\}$;
24:       else
25:         $X_{samples} \leftarrow \{x_m\}$;
26:         $V \leftarrow \{x_m\}; Q_V \leftarrow \{x_m\}$;
27:         $Q_V \leftarrow V$;
28:         $r \leftarrow \text{radius}(|V| + |X_{samples}|)$;
29:       end if
30:     end if
31:   end if
32: end if
33: end if
34: end if
35: else
36:   $Q_E \leftarrow \emptyset; Q_V \leftarrow \emptyset;$
37: end if
38: until STOP
39: return $\tau$;

3.2. Trajectory optimization. We obtain a series of discrete path points representing the shortest path through the path search in the previous section, which is relatively sparse, not smooth, and do not contain any time information. Therefore, it cannot be directly converted into the operation control quantity of the actuator. In order to control the quadrotor movement, these sparse path points need to be changed into continuous and smooth curves. Then we generate a minimum-snap polynomial trajectory through a sequence of path points.
Dividing the discrete trajectories into $\lambda$-segments polynomials, and decoupling each of the polynomials on the $X, Y$-axis, we obtain a set of polynomial trajectory sequences respectively, denoted as $f_x(t), f_y(t)$. Here, $f_x(t)$ and $f_y(t)$ have the same form. Let’s take the $X$-axis as an example, polynomial trajectory sequence is expressed as:

$$
f_x(t) = \begin{cases} 
  f_1(t) = \sum_{j=0}^{N} p_{1,j} t^j, & T_0 \leq t \leq T_1 \\
  f_2(t) = \sum_{j=0}^{N} p_{2,j} t^j, & T_1 \leq t \leq T_2 \\
  \vdots \\
  f_\lambda(t) = \sum_{j=0}^{N} p_{\lambda,j} t^j, & T_{\lambda-1} \leq t \leq T_\lambda
\end{cases} \quad (3.1)
$$

where $i$ represents the $i$th segment, $i = 1, 2, ..., \lambda$; $j$ represents the order of each polynomial, $j = 0, 1, 2, ..., N$, ($N$ is the total order of polynomial trajectories); $p(i,j)$ represents the $j$th order coefficient of the $i$th polynomial trajectory.

It is assumed that the motions between each polynomial trajectory in $f_x(t)$ are uniformly accelerated, uniformly velocity and uniformly decelerated. In any section of polynomial trajectory, according to the Euclidean distance $d_{i,i-1}$ between two discrete points $i$ and $i-1$, given the maximum acceleration $a_{\text{max}}$ and maximum velocity $v_{\text{max}}$, the time interval $\Delta T$ between two discrete points $i$ and $i-1$ is calculated:

$$
\Delta T = T_i - T_{i-1} = \frac{d_{i,i-1}}{v_{\text{max}}} + \frac{v_{\text{max}}}{a_{\text{max}}} \quad (3.2)
$$

The cost function is constituted by the integral of the fourth derivative squared of each segment of the polynomial trajectory in the corresponding time period, which guaranteed that the polynomial trajectory has a solution. The cost function is expressed as the quadratic form:

$$
J_{\text{total}} = \begin{cases} 
  J_1(t) \int_{T_0}^{T_1} (f_1^{(4)}(t))^2 dt = p_1^T Q_1 p_1 \\
  J_2(t) \int_{T_1}^{T_2} (f_2^{(4)}(t))^2 dt = p_2^T Q_2 p_2 \\
  \vdots \\
  J_\lambda(t) \int_{T_{\lambda-1}}^{T_\lambda} (f_\lambda^{(4)}(t))^2 dt = p_\lambda^T Q_\lambda p_\lambda
\end{cases} \quad (3.3)
$$

where $p_i$ corresponds to the coefficient of $f_x(t)$ on segment $i$; $Q_i$ is the quadratic matrix of the cost function of the polynomial trajectory on segment $i$.

By solving for the minimum cost function, we obtain the optimal parameter of the optimal polynomial trajectory on the $X$-axis.

Similarly, we obtain the optimal polynomial trajectory and the corresponding optimal parameters in the $Y$-axes respectively. Finally, we obtain the optimal trajectory in the two-dimensional space.
4. **Quadrotor dynamics.** In this section, we present the dynamic model of a quadrotor employed in the controller’s formulation. We assume that the structure of the model is rigid and symmetrical, the propellers are rigid and the thrust and drag forces are proportional to the square of propellers’ speed.

The quadrotor structure is presented in Fig. 2, let \( \{O_e, X_e, Y_e, Z_e\} \) denote the earth-fixed inertial frame and \( \{O_b, X_b, Y_b, Z_b\} \) denote the body-fixed frame whose origin \( O_b \) is at the center of mass of the quadrotor. The inertial position of the quadrotor is defined by \( P = [x, y, z]^T \) and the attitude is defined by three Euler angles: roll \( \phi \), pitch \( \theta \), and yaw \( \psi \) (\( \Theta = [\phi, \theta, \psi]^T \)). \( V_b = [v_x, v_y, v_z]^T \) and \( \Omega = [p, q, r]^T \) represents for the inertial velocity in the body-fixed frame and body angular velocities, respectively.

According to Newton’s second law, we have:

\[
F = m \frac{dV}{dt} \tag{4.1}
\]

\[
M = \frac{dH}{dt} \tag{4.2}
\]

where \( F \) is the external force, \( M \) is the external torque of the quadrotor, and \( H \) is the moment of momentum.

The main force of the quadrotor during the flight is as following:

The gravity \( F_g \) is:

\[
F_g = [0 \ 0 \ mg]^T \tag{4.3}
\]

The lift force \( F_b \) provided by the four motors in the body coordinate system can be expressed as:

\[
F_b = [0 \ 0 \ \sum_{i=0}^{4} F_i]^T = b_i \omega_i^2 (i = 1, 2, 3, 4) \tag{4.4}
\]

where \( F_i \) is the lift of each rotor, \( b_i \) is the lift coefficient, and \( \omega_i \) is the speed of each motor.
The rotation matrix between the earth-fixed inertial frame \( \{O_eX_eY_eZ_e\} \) and the body-fixed frame \( \{O_bX_bY_bZ_b\} \) coordinates is expressed as:

\[
\begin{bmatrix}
\cos \theta \cos \psi & \cos \theta \sin \psi & -\sin \theta \\
\sin \phi \sin \theta \cos \psi - \cos \phi \sin \psi & \sin \phi \sin \theta \sin \psi + \cos \phi \cos \psi & \sin \phi \cos \theta \\
\cos \phi \sin \theta \cos \psi + \sin \phi \sin \psi & \cos \phi \sin \theta \sin \psi - \sin \phi \cos \psi & \cos \phi \cos \theta
\end{bmatrix}
\]

Therefore, according to the coordinate transformation matrix, thrust \( F_T \) can be expressed as:

\[
F_T = \hat{E}_B \mathbf{F}_b = (\sum_{i=0}^{4} F_i)
\]

The resultant force of the quadrotor in the ground coordinate system is given below:

\[
F = F_y + F_T
\]

Assuming the quality of the quadrotor is invariant during flight, the position equation of the quadrotor is obtained as:

\[
\begin{bmatrix}
p \\
q \\
r \\
v_x \\
v_y \\
v_z
\end{bmatrix} = \begin{bmatrix}
v_x \cos \theta \cos \psi + v_y (\sin \phi \sin \theta \cos \psi - \cos \phi \sin \psi) + v_z (\cos \phi \sin \theta \cos \psi + \sin \phi \cos \psi) \\
v_x \cos \theta \sin \phi \sin \theta \sin \psi + v_y (\cos \phi \sin \theta \sin \psi + \sin \phi \cos \psi) + v_z (\cos \phi \sin \theta \sin \psi - \sin \phi \cos \psi) \\
-\psi \sin \phi \sin \phi \cos \psi \sin \theta + v_y \sin \phi \cos \phi \cos \theta + v_z \sin \phi \cos \phi \sin \theta + \cos \phi \cos \psi \\
(\cos \phi \sin \theta \cos \psi + \sin \phi \sin \psi) \sum_{i=0}^{4} F_i / m \\
(\cos \phi \sin \theta \sin \psi - \sin \phi \cos \psi) \sum_{i=0}^{4} F_i / m \\
(\cos \phi \cos \theta) \sum_{i=0}^{4} F_i / m - g
\end{bmatrix}
\]

The rotation matrix for angular velocities from the earth-fixed inertial frame to the body-fixed frame is \( \hat{E}_B \), then we obtained:

\[
\dot{\mathbf{\Omega}} = \hat{E}_B \mathbf{\Omega}, \quad \begin{bmatrix}
\dot{\phi} \\
\dot{\theta} \\
\dot{\psi}
\end{bmatrix} = \begin{bmatrix}
1 & \sin \phi \tan \theta & \cos \phi \tan \theta \\
0 & \cos \phi & -\sin \phi \\
0 & \sin \phi \sec \theta & \cos \phi \sec \theta
\end{bmatrix} \begin{bmatrix}
p \\
q \\
r
\end{bmatrix}
\]

The moments of inertia matrix is given below:

\[
\mathbf{J} = \begin{bmatrix}
J_x & J_y \\
J_y & J_z
\end{bmatrix}
\]

where \( J_x \), \( J_y \), and \( J_z \) are the moments of inertia of the three-axis in the body coordinate system.

By the rigid body inertia calculation, the formula can be obtained as:

\[
\mathbf{M}_b = \mathbf{J} \begin{bmatrix}
\dot{p} \\
\dot{q} \\
\dot{r}
\end{bmatrix}
\]

Note that the following formula can be introduced in the body coordinate system of the quadrotor around the axis of the angular acceleration:

\[
\begin{bmatrix}
\dot{p} \\
\dot{q} \\
\dot{r}
\end{bmatrix} = \begin{bmatrix}
(J_y - J_z)qr / J_x + LU_2 / J_x \\
(J_z - J_x)pr / J_y + LU_3 / J_y \\
(J_x - J_y)pq / J_z + LU_4 / J_z
\end{bmatrix}
\]

where \( L \) is the length of the quadrotor arms.
is the state vector and the period vector. The linear discrete-time-varying state-space model assuming a sampling period \( T \) is needed. Then, the discretized linear dynamics around the reference point are achieved.

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{y}_1 \\
\dot{z}_1 \\
v_x \\
v_y \\
v_z \\
\psi \\
\theta \\
\psi \\
q \\
r 
\end{bmatrix} = \begin{bmatrix}
v_x \cos \theta \cos \psi + v_y (\sin \phi \sin \theta \cos \psi - \cos \phi \sin \psi) + v_z (\cos \phi \sin \theta \cos \psi + \sin \phi \sin \psi) \\
v_x \cos \theta \sin \psi + v_y (\sin \phi \sin \theta \sin \psi + \cos \phi \cos \psi) + v_z (\cos \phi \sin \theta \sin \psi - \sin \phi \cos \psi) \\
n_x \cos \psi + n_y \sin \phi \cos \psi + n_z \sin \phi \sin \psi \\
n_x \cos \psi + n_y \sin \phi \cos \psi + n_z \sin \phi \sin \psi \\
\psi \\
\theta \\
\psi \\
q \\
r 
\end{bmatrix} \begin{bmatrix}
u_x u_{1x}/m \\
u_y u_{1x}/m \\
u_z u_{1x}/m - g \\
p + q \sin \phi \tan \theta + \rho \cos \phi \tan \theta \\
\rho \\
\tan \phi \\
(J_y - J_1) q v / J_x + L U_{1x}/J_x \\
(J_x - J_1) p v / J_y + L U_{1y}/J_y \\
(J_x - J_2) p q / J_z + L U_{1z}/J_z 
\end{bmatrix}
\]

(4.13)

\[
\begin{bmatrix}
\begin{bmatrix}
u_x \\
u_y \\
u_z 
\end{bmatrix} = \begin{bmatrix}
\cos \phi \sin \theta \cos \psi + \sin \phi \sin \psi \\
\cos \phi \sin \theta \sin \psi - \sin \phi \cos \psi \\
\cos \phi \cos \psi 
\end{bmatrix}
\end{bmatrix}
\]

(4.14)

\[
U = \begin{bmatrix}
U_1 \\
U_2 \\
U_3 \\
U_4 
\end{bmatrix} = \begin{bmatrix}
b_1 (\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2) \\
b_1 (-\omega_2^2 + \omega_4^2) \\
b_1 (\omega_1^2 - \omega_3^2) \\
c (-\omega_1^2 + \omega_2^2 - \omega_3^2 + \omega_4^2) 
\end{bmatrix}
\]

(4.15)

where \( u_x, u_y, u_z \) are the direction cosines of the total thrust vector along the \( X_b, Y_b, Z_b \) axes, \( U \) is the input vector consisting of \( U_1 \) (total thrust), and \( U_2, U_3, U_4 \), which are related to the rotations of the quadrotor. \( c \) is drag coefficient.

5. **Model predictive trajectory tracking controller.** In this section, we apply LMPC strategy to design for translational and rotational subsystem separately to achieve path tracking.

In equations (4.13), the first six equations describe the translational quadrotor dynamics, while the other six equations represent the rotational dynamics. To facilitate the design of the MPC controller, linearization of the system model is required. Then, the discretized linear dynamics around the reference point are needed.

For the translational dynamics, \( x_\zeta(k) = [x(k), y(k), z(k)] \) is the state vector and \( u_\zeta(k) = [u_x U_1, u_y U_1, mg - u_z U_1]^T \) is the control input vector. The linear discrete-time-varying state-space model assuming a sampling period \( T_s^\zeta \) at the reference point \( \{x_\zeta, u_\zeta\} \) of the quadrotor can be written as below:

\[
\dot{x}_\zeta(k + 1) = A_\zeta \dot{x}_\zeta(k) + B_\zeta \tilde{u}_\zeta(k)
\]

(5.1)

Here, \( k \) is corresponding to the sample index, \( \tilde{x}_\zeta = x \zeta - \bar{x}_\zeta, \tilde{u} = u_\zeta - u_\zeta \). The matrices \( A_\zeta \) and \( B_\zeta \) can be written as below:

\[
A_\zeta = \begin{bmatrix}
1 & T_s^\zeta & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & T_s^\zeta & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & T_s^\zeta \\
0 & 0 & 0 & 0 & 0 & 1 
\end{bmatrix}
\]

\[
B_\zeta = \begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
0 \\
0
\end{bmatrix}
\]
For the rotational dynamics, $x_\eta(k) = [\phi(k) \ p(k) \ \theta(k) \ q(k) \ \psi(k) \ r(k)]^T$ is the state vector and $u_\eta(k) = [U_2 \ U_3 \ U_4]^T$ is the control input vector. The linear discrete-time-varying state-space model assuming a sampling period $T_s$ (with a sampling period $T_s \neq T_s^\phi$) can be written as below:

$$\dot{x}_\eta(k+1) = A_\eta \dot{x}_\eta(k) + B_\eta \ddot{u}_\eta(k)$$  \hspace{1cm} (5.2)

$$A_\eta = \begin{bmatrix} 1 & T_s^\phi & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \Delta rT_s^\phi(J_y - J_z)/J_z & 0 & \Delta qT_s^\phi(J_y - J_z)/J_z \\ 0 & 0 & 1 & \Delta rJ_s^\phi(J_z - J_x)/J_z & 0 & \Delta pT_s^\phi(J_z - J_x)/J_z \\ 0 & 0 & 0 & 1 & \Delta pJ_s^\phi(J_x - J_y)/J_y & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta qJ_s^\phi(J_y - J_x)/J_z & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$B_\eta = \begin{bmatrix} T_s^\phi L/J_x & 0 & 0 \\ 0 & T_s^\phi L/J_y & 0 \\ 0 & 0 & T_s^\phi L/J_z \end{bmatrix}$$

In order to ensure the quadrotor to track the desired trajectory quickly and smoothly, an objective function is required in the MPC algorithm. The objective function is necessary to add the optimization of system state variables and control variables. In order to avoid control variables change suddenly, we add the control increment change constraints during each sampling period to the objective function. Consequently, the objective function for translational motion MPC is expressed as:

$$J_\eta(k) = \sum_{i=1}^{N_p^\xi} ||\eta^c_i(k + i | k) - \eta^c_{ref}(k + i | k)||_Q^2 + \sum_{i=0}^{N_p^\xi - 1} ||\Delta \ddot{u}^c_i(k + i | k)||_R^2$$  \hspace{1cm} (5.3)

Subject to:

$$\tilde{u}^c_{min}(k) \leq \ddot{u}^c_i(k) \leq \tilde{u}^c_{max}(k), \ k = 0, 1, \cdots, N_p^\xi - 1$$  \hspace{1cm} (5.4)

$$\Delta \tilde{u}^c_{min}(k) \leq \Delta \ddot{u}^c_i(k) \leq \Delta \tilde{u}^c_{max}(k), \ k = 0, 1, \cdots, N_p^\xi - 1$$  \hspace{1cm} (5.5)

where $\eta^c_i(k + i | k)$ and $\eta^c_{ref}(k + i | k)$ denote the output state and the reference output state in the predicted horizon, respectively. In the objective function (5.3), the first item on the right side describes the capability of trajectory tracking and the second item reflects the constraint on the change of the control input. The weighting terms $Q$, $R$ are diagonal matrices. $N_p^\xi$ is the prediction horizon, and $N_p^\xi$ is the control horizon.

In order to solve the objective function (5.3), a variable is given as:

$$\xi(k | k) = \begin{bmatrix} x(k | k) \\ u(k - 1 | k) \end{bmatrix}$$  \hspace{1cm} (5.6)

Then we get a new state-space expression:

$$\xi(k + 1 | k) = \tilde{A}_\xi \xi(k | k) + \tilde{B}_\xi \Delta u^c_i(k | k)$$  \hspace{1cm} (5.7)
\[
\eta^\zeta(k | k) = \hat{C}_\zeta \xi(k | k) \tag{5.8}
\]

where \( \hat{A}_\zeta = \begin{bmatrix} A_\zeta & B_\zeta \\ 0_{m \times n} & I_m \end{bmatrix}, \hat{B}_\zeta = \begin{bmatrix} pB_\zeta \\ I_m \end{bmatrix} \), \( n \) is the dimension of the state variable, \( m \) is the dimension of control variable.

After analysis, prediction output expression of the system (5.8) is obtained in a matrix form that can be expressed as:

\[
Y^\zeta(k) = \Psi^\zeta \xi(k | k) + \Theta^\zeta \Delta U^\zeta(k) \tag{5.9}
\]

\[
\Psi^\zeta = \begin{bmatrix} \hat{C}_\zeta \hat{A}_\zeta \\ \hat{C}_\zeta \hat{A}_\zeta^2 \\ \vdots \\ \hat{C}_\zeta \hat{A}_\zeta^{N_p} \end{bmatrix}
\]

\[
\Theta^\zeta = \begin{bmatrix} \hat{C}_\zeta \hat{B}_\zeta & 0 & 0 & \cdots & 0 \\ \hat{C}_\zeta \hat{A}_\zeta \hat{B}_\zeta & \hat{C}_\zeta \hat{B}_\zeta & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \hat{C}_\zeta \hat{A}_\zeta^{N_p-2} \hat{B}_\zeta & \hat{C}_\zeta \hat{A}_\zeta^{N_p-3} \hat{B}_\zeta & \cdots & \hat{C}_\zeta \hat{B}_\zeta \\ \hat{C}_\zeta \hat{A}_\zeta^{N_p-1} \hat{B}_\zeta & \hat{C}_\zeta \hat{A}_\zeta^{N_p-2} \hat{B}_\zeta & \cdots & \hat{C}_\zeta \hat{A}_\zeta^{N_p-N_c-1} \hat{B}_\zeta \end{bmatrix}
\]

\[
\Delta U^\zeta(k) = \begin{bmatrix} \Delta u^\zeta(k | k) \\ \Delta u^\zeta(k + 1 | k) \\ \vdots \\ \Delta u^\zeta(k + N_c | k) \end{bmatrix}
\]

Let’s substitute equation (5.9) into equation (5.3), then the objective function (5.3) is rewritten as:

\[
J^\zeta(k) = \| \hat{Y}^\zeta(k) \|_Q^2 + \| \Delta U^\zeta(k) \|_R^2 \tag{5.10}
\]

where \( \hat{Q} = \bigoplus_{i=1}^{N_c} Q, \hat{R} = \bigoplus_{i=0}^{N_c-1} R \). \( \hat{Y}^\zeta(k) = Y^\zeta(k) - Y^r(k) \), \( Y^r(k) \) is the future reference output.

The objective function for rotational dynamics is consistent with translational motion.

After solve the above-mentioned optimization problem in each control period, we obtain a set of control input increment in the control horizon:

\[
\Delta U^*_k = [\Delta u^*_k, \Delta u^*_{k+1}, \ldots, \Delta u^*_{k+N-1}] \tag{5.11}
\]
Although the prediction and optimization are performed over a control horizon, only the first element of the control sequences \( (5.11) \) at the time instant \( k \) is applied to the system:

\[
    u(t) = u(t - 1) + \Delta u_t^* \quad (5.12)
\]

After starting the next control period, the above procedure is repeated. Then the optimal solution of the objective function drives the quadrotor to achieve path tracking.

6. Experimental studies. In this section, we first verify the timeliness and effectiveness of the BIT* path planning algorithm using RVIZ visual interface. Then, in order to test the LMPC controllers’ performance of path tracking, the BIT* algorithm is implemented in MATLAB to generate the reference trajectory. Besides, a comparative simulation with the PID tracking controller is carried out to verify the effectiveness.

6.1. Path planning test. A fundamental component of BIT* is the application of heuristic estimates to all aspects of path cost. For a given computational time, BIT* has a higher likelihood of finding a solution and generally finds solutions of equivalent quality sooner. To verify the path planning module of the quadrotor, we use RVIZ visual interface to test and evaluate the performance of the quadrotor path planning module. The solution time is set in the program as 0.3s. The result is shown in Fig. 3. In RVIZ visual interface, the red line represents the BIT* algorithm path search result, and the green line represents the optimization of the planning result.

![Path planning module test result.](image)

6.2. Path tracking test. To demonstrate the capabilities of the proposed linear model predictive controller, we carry out the simulation experiment on the MATLAB / Simulink platform. By creating a map with obstacles at first, we give the starting point (the red dot) and the endpoint (the green dot), then, the motion planning module is implemented for simulation in MATLAB 2018 with MATLAB programming language. The planned reference trajectory is shown in Fig. 4.
First scenario. For the path tracking problem, one quadrotor is considered with the nonlinear dynamics described in Section 4. The quadrotor relies on the control algorithm from Section 5. The initial conditions for translational and rotational states are set to \(x^{init}_\eta = [0 \ 0 \ 0 \ 0 \ 0 \ 0]^T\), and \(x^{init}_\xi = [22.87 \ 16.62 \ 2 \ 0 \ 0 \ 0]^T\). For all the optimization problems, the prediction and control horizon are set to \(N_p = 10\), \(N_c = 2\). \(Q_\eta = diag(1, 1, 100, 1, 1, 100)\), \(R_\eta = 20I_3\), \(Q_\zeta = diag(10, 10, 100, 1, 1, 100)\), and \(R_\zeta = 10I_3\). Due to rotor constraints, the practical restrictions on the states and the inputs of the controller are:

\[
-\frac{\pi}{2} < \phi, \theta < \frac{\pi}{2} \text{ (rad)}, \quad 0 < \psi < \pi \text{ (rad)}
\]

\[-0.8 < p, q < 0.8 \text{ (rad/s)}, \quad -1 < r < 1 \text{ (rad/s)}
\]

\[-2 < v_x, v_y, v_z < 2 \text{ (m/s)}
\]

\[0 \leq U_1 < 20(N),
\]

\[|U_2| \leq 18.5(N),
\]

\[|U_3| \leq 18.5(N),
\]

\[|U_4| \leq 1(N).
\]

Second scenario. A comparative simulation between the LMPC and the PID tracking controller is shown in this section. In the comparative simulation, the initial conditions of the quadrotor are the same as the first scenario, \([x(0) \ y(0) \ z(0)]^T = [22.87 \ 16.62 \ 2]^T\).

The comparative simulation results are shown in Figs. 5-7. Path tracking results of the two quadrotor controllers are shown in Fig. 5. The red line represents the reference path, the blue line represents path tracking results using the LMPC controller, and the green line represents the path tracking results using the PID controller. Fig. 6 presents the tracking results of the X-axis of two controllers. The red line represents the reference path of the X-axis, the blue line represents the path tracking results of the X-axis of the MPC controller using the LMPC controller, and the green line represents the path tracking results of the X-axis of the PID controller. Fig. 7 presents the tracking results of the Y-axis of two controllers. The red line represents the reference path of the Y-axis, the blue line represents
path tracking results of the Y-axis of the MPC controller using the LMPC controller, and the green line represents the path tracking results of the Y-axis of the PID controller. It can be seen from the results that the two controllers both can effectively track the reference path. However, the tracking performance using the LMPC controller is smoother than the PID controller, and the proposed LMPC controller has a small overshoot and a shorter adjustment time. The tracking errors using the LMPC controller asymptotically converges to the reference position more quickly than the PID controller. In general, LMPC has precise tracking and good control performance.

**Figure 5.** Path tracking of the quadrotor.

**Figure 6.** Path tracking of $x(t)$. 

7. Conclusion. In this paper, we design a motion planning system based on BIT* to generate trajectory in an obstacle environment and a cascaded control approach based on linear model predictive control to achieve the trajectory tracking for quadrotors. The simulations show that the BIT* algorithm finds better solutions more quickly with a given computational time and the tracking algorithm based on LMPC has good tracking performance. Our work in this paper is currently tested in the simulation environment, thus our future work will focus on applying these methods in real quadrotors and test the performance of the whole system in real environments.

Acknowledgments. We would like to thank you for following the instructions above very closely in advance. It will definitely save us lot of time and expedite the process of your paper’s publication.

REFERENCES

[1] Y. Bai, H. Liu, Z. Shi and Y. Zhong, Robust control of quadrotor unmanned air vehicles, in Proceedings of the 31st Chinese Control Conference, Hefei, (2012), 4462–4467.
[2] J. Carsten, D. Ferguson and A. Stentz, 3D field d: Improved path planning and replanning in three dimensions, in 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, (2006), 3381–3386.
[3] Y.-B. Chen, G.-C. Luo, Y.-S. Mei, J.-Q. Yu and X.-L. Su, UAV path planning using artificial potential field method updated by optimal control theory, Internat. J. Systems Sci., 47 (2016), 1407–1420.
[4] F. Chen, J.-Q. Yu, X.-L. Su and G.-C. Luo, Path planning for multi-UAV formation, Journal of Intelligent and Robotic Systems, 77 (2015), 229–246.
[5] N. R. Council, Autonomous Vehicles in Support of Naval Operations, The National Academies Press, Washington, DC, 2005, https://www.nap.edu/catalog/11379/autonomous-vehicles-in-support-of-naval-operations.
[6] L. de Filippis, G. Guglieri and F. Quagliotti, Path planning strategies for UAVs in 3D environments, Journal of Intelligent & Robotic Systems, 65 (2012), 247–264.
[7] T. Dierks and S. Jagannathan, Output feedback control of a quadrotor UAV using neural networks, IEEE Transactions on Neural Networks, 21 (2010), 50–66.
[8] E. W. Dijkstra, A note on two problems in connexion with graphs, Numer. Math., 1 (1959), 269–271.
[9] J. D. Gammell, S. S. Srinivasa and T. D. Barfoot, Batch informed trees (BIT*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs, 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, (2015), 3067–3074.

[10] S. R. Herwitz, L. F. Johnson, S. E. Dunagan, R. G. Higgins, D. V. Sullivan, J. Zheng, B. M. Lobitz, J. G. Leung, B. A. Gallmeyer, M. Aoyagi, R. E. Slye and J. A. Brass, Imaging from an unmanned aerial vehicle: Agricultural surveillance and decision support, in Computers and Electronics in Agriculture, 44 (2004), 49–61.

[11] Y. Kwangjin and S. Sukkarieh, Real-time continuous curvature path planning of UAVs in cluttered environments, in 2008 5th International Symposium on Mechatronics and its Applications, IEEE, (2008), 1–6.

[12] X. Q. Li, L. Qiu, S. Aziz, J. F. Pan, J. P. Yuan and B. Zhang, Control method of UAV based on RRT* for target tracking in cluttered environment, in 2017 7th International Conference on Power Electronics Systems and Applications - Smart Mobility, Power Transfer & Security (PESA), (2017), 1–4.

[13] R. Li, Y. Shi and K.-L. Teo, Coordination arrival control for multi-agent systems, Internat. J. Robust Nonlinear Control, 26 (2016), 1456–1474.

[14] Z. Ma, T. Hu, L. Shen, W. Kong, B. Zhao and K. Yao, An iterative learning controller for quadrotor UAV path following at a constant altitude, in 2015 34th Chinese Control Conference (CCC), Hangzhou, China, (2015), 4406–4411.

[15] M. Nguyen Duc, T. N. Trong and Y. S. Xuan, The quadrotor MAV system using PID control, in 2015 IEEE International Conference on Mechatronics and Automation (ICMA), Beijing, (2015), 506–510.

[16] R. Sharma, M. Kothari, C. N. Taylor and I. Postlethwaite, Cooperative target-capturing with inaccurate target information, in Proceedings of the 2010 American Control Conference, Baltimore, MD, (2010), 5520–5525.

[17] Z. Shulong, A. Honglei, Z. Daibing and S. Lincheng, A new feedback linearization LQR control for attitude of quadrotor, in 2014 13th International Conference on Control Automation Robotics Vision (ICARCV), Singapore, (2014), 1593–1597.

[18] M. Sumaila, Techniques for quadcopter modelling & design: A review, Journal of Unmanned System Technology.

[19] J. Xiong and G. Zhang, Sliding mode control for a quadrotor UAV with parameter uncertainties, in 2016 2nd International Conference on Control, Automation and Robotics (ICCAR), Hong Kong, (2016), 207–212.

[20] F. Yan, Y.-S. Liu and J.-Z. Xiao, Path planning in complex 3D environments using a probabilistic roadmap method, International Journal of Automation and Computing, 10 (2013), 525–533.

[21] B. Zhao, B. Xian, Y. Zhang and X. Zhang, Nonlinear robust adaptive tracking control of a quadrotor UAV via immersion and invariance methodology, IEEE Transactions on Industrial Electronics, 62 (2015), 2891–2902.

Received December 2020; revised January 2021.

E-mail address: 1049529375@qq.com
E-mail address: lirui@uestc.edu.cn
E-mail address: shiyijingjing@uestc.edu.cn
E-mail address: 526741308@qq.com