Optimal Tracking Control Scheme for a Quadrotor based on Off-policy Reinforcement Learning

Pham Dinh Duong

Assoc. Prof. Dao Phuong Nam

Hanoi University of Science and Technology

Outlines

Introduction (1)

Why Quadrotor

- The advantages and disadvantages of quadrotor compared to another UAV
- The application of quadrotor

Figure: Flycam of DJI

Figure: Flying Taxi

Introduction (2)

Advantages of Quadrotor

- Vertical Take-off and Landing (VTOL) Ability
- The ability of staying at a specific position
- More flexible than Helicopter

Disadvantages of Quadrotor

- The maximum speed is less than that of other types of UAV

Figure: Helicopter

Figure: Hybrid UAV

Introduction (3)

Application of Quadrotor

Figure: In agriculture

Figure: In filming, surveillance

In addition, quadrotors also have many other applications in military, delivery and natural disaster forecasting

The problem statement and Contributions of the Project (1)

The problem statement

There have been many outstanding studies on quadrotors in the past 10 years:

- Generating trajectory using the *Minimum-snap* method and using conventional PD controller (Vijay-2011 [3]).
- Robust heading angles control with an uncertain quadrotor model based on Quaternion (Hao Liu-2015 [2])

The above methods partly (or fully) require knowledge about the model. But recently, with more application of adaptive algorithms and reinforcement learning, some articles solve the problem of *parameter uncertainty using nonlinear controllers* (ChenLiu-2021 [5]) or *optimal control of quadrotor formation with no knowledge of model* (as in Zhao-2021 [6]). The problem of [6] is that although formation control is mentioned, the trajectory is quite simple.

The problem statement and Contributions of the Project (2)

Contributions of the Project

As in [6], the project solves the control problem completely without knowledge of the model (apply only to a quadrotor). However:

- Solve the Optimal Tracking Control Problem (OTCP) with complex trajectory using Data-driven PI algorithm (An Off-policy Reinforcement Learning algorithm)
- The results of the proposed method are equivalent to the optimal solution in the case where the model is known
- Evaluate the influence of the decay factor λ in the cost function

In general, the quadrotor model is quite complicated but can still be reduced to affine nonlinear form and solve the optimal tracking problem with an unknown model (Model-free).

RL for Optimal tracking control problem (1)

Non-linear affine system:

$$\dot{x}(t) = f(x(t)) + g(x(t))u(t) \tag{1}$$

- $x = [x_1 ... x_n]^T \in \mathbb{R}^n$
- $u = [u_1 ... u_m] \in \mathbb{R}^m$
- $f(x(t)) \in \mathbb{R}^n$
- $g(x(t)) \in \mathbb{R}^{n \times m}$

Assumption

- $f(0) = 0$
- $f(x(t)) + g(x(t))u(x(t))$ satisfy the Lipschitz condition for continuity in a domain $\Omega \subseteq \mathbb{R}^n$

RL for Optimal tracking control problem (2)

Feed-forward Scheme

If $f(x)$, $g(x)$ is known and there exists $g^{-1}(x)$:

$$u_d(t) = g^{-1}(x_d(t))(\dot{x}_d(t) - f(x_d(t))) \tag{2}$$

In [Kamalapurkar – 2015] [1] presented the problem of solving the optimal tracking control problem using the Feed-Forward method, in which the optimal feedback signal $u_e(t) = u(t) - u_d(t)$ is used to optimize the cost function:

$$V_e(e_d(t)) = \int_t^{\infty} (e_d(\tau)^T Q_e e_d(\tau) + u_e(\tau)^T R u_e(\tau)) d\tau \tag{3}$$

And can be obtained by solving the HJB equation:

$$u_e^* = -\frac{1}{2} R^{-1} g(x(t))^T \frac{\partial V_e(e_d(t))}{\partial e_d} \tag{4}$$

RL for Optimal tracking control problem (3)

Definition

$$\begin{cases} e_d(t) = x(t) - x_d(t) \\ V(e_d(t), x_d(t)) = \int_t^{\infty} e^{-\lambda(\tau - t)}(e_d(\tau)^T Q_e e_d(\tau) + u(\tau)^T R u(\tau))d\tau \end{cases} \tag{5}$$

- $Q_e \in \mathbb{R}^{n\times n} \succ 0$
- $R \in \mathbb{R}^{m\times m} \succ 0$

The desired trajectory

$$\dot{x}_d(t) = r_d(x_d(t)), \quad r_d(0) = 0$$

- $x_d(t)$ is bounded
- $r_d(x_d(t)) \in \mathbb{R}^n$ is continuous Lipschitz function

RL for Optimal tracking control problem (4)

Data-driven PI algorithm for OTCP

Data-driven PI algorithm solves the problem in the case with unknown-dynamics system
Set $X(t) = [e_d(t)^T \; x_d(t)^T]^T \in \mathcal{X} \subset \mathbb{R}^{2n}$ and from (1) we have:

$$\dot{X}(t) = F(X(t)) + G(X(t))u(t) \tag{6}$$

where:

$$\begin{cases} F(X(t)) = \begin{bmatrix} f(e_d(t) + x_d(t)) - r_d(x(t)) \\ r_d(x_d(t)) \end{bmatrix} \\ G(X(t)) = \begin{bmatrix} g(e_d(t) + x_d(t))) \\ 0 \end{bmatrix} \end{cases}$$

RL for Optimal tracking control problem (5)

Data-driven PI algorithm for OTCP

The cost function is chosen as follow:

$$V(X(t)) = \int_t^\infty e^{-\lambda(\tau - t)}[X(\tau)^T Q X(\tau) + u(\tau)^T R u(\tau)] d\tau \tag{7}$$

$$Q = \begin{bmatrix} Q_e & 0 \\ 0 & 0 \end{bmatrix} \tag{8}$$

The decay factor λ in the cost function is necessary because: if the desired trajectory does not go to zero when $t \to \infty$, the cost function with $\lambda = 0$ will *not is bounded* because the feedforward component $u_d(t)$ depends on the desired state trajectory.

RL for Optimal tracking control problem (6)

Summary of Data-driven PI algorithm

1. **Init**
   Start with a stable control signal u^0 and add a noise component u_e to ensure the PE condition. Collect data and determine threshold ϵ
2. **Policy Evaluation and Policy Improvement**
   With the signal $u^i(X)$ solved from the previous loop, solve $V^{i+1}(X)$ and $u^{i+1}(X)$ from the equation:

$$\begin{aligned} V^{i+1}(X(t+\delta t)) - V^{i+1}(X(t)) = -\int_t^{t+\delta t} & [X(\tau)^T Q X(\tau) \\ & + [u^i(X(\tau))]^T R u^i(X(\tau))] d\tau + \int_t^{t+\delta t} \lambda V^{i+1}(X(\tau)) d\tau \\ & + 2\int_t^{t+\delta t} [u^{i+1}(X(\tau))]^T R [u^i(X(\tau)) - u(\tau)] d\tau \end{aligned} \tag{9}$$

3. **Checking convergence**
   Stop iterating if $\|u^{i+1} - u^i\| < \epsilon$, otherwise: update $u^i = u^{i+1}$, return to step (2).

RL for Optimal tracking control problem (7)

Data-driven PI

The Actor-Critic Neural Network structure is introduced to estimate $V(X)$ and $u(X)$ as follows:

$$\begin{aligned}\hat{V}^i(X) &= [w_V^i]^T\varphi(X)\\ \hat{u}^i(X) &= [w_u^i]^T\psi(X)\end{aligned} \tag{10}$$

- $\varphi(X) \in \mathbb{R}^{l_\varphi}$ and $\psi(X) \in \mathbb{R}^{l_\psi}$
- $w_V \in \mathbb{R}^{l_\varphi}$ and $w_u \in \mathbb{R}^{l_\psi \times m}$

Apply Data-driven PI for Quadrotor (1)

3.1. Kinematic model

1. Coordinate frame attached to the Earth, denoted α with axes $E_x E_y E_z$
2. Coordinate frame fixed to the Quadrotor, denoted β with axes $B_x B_y B_z$
3. State variables: (position and Euler angles)
   - $p = [p_x, p_y, p_z]^T \in \mathbb{R}^3$
   - $\Theta = [\phi, \theta, \psi] \in \mathbb{R}^3$

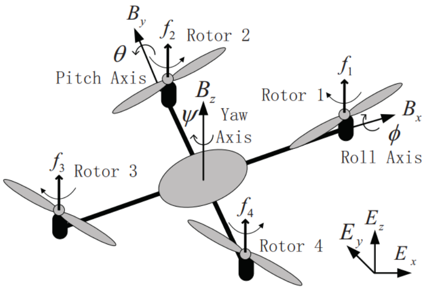


Figure: Kinematic model

Apply Data-driven PI for Quadrotor (2)

3.2. Dynamic model

$$m\ddot{p} = fRe_3^3 - mge_{3,3} \tag{11}$$

$$J\ddot{\Theta} = \tau - C(\Theta, \dot{\Theta})\dot{\Theta} \tag{12}$$

- $T_p \in \mathbb{R}$ is the total lift force of the blades in the β coordinate system, $T_p = T_1 + T_2 + T_3 + T_4$. $T_p = k_w u_z$ and $\tau = [l_\tau k_w u_\phi,\ l_\tau k_w u_\theta,\ k_t u_\psi]^T$ Where k_w and k_t are aerodynamic constants
- m is the mass of the quadrotor, g is the gravitational acceleration
- $J = diag(J_x, J_y, J_z)$

[1]

[1] $e_{(N,j)}$ is the column vector $N \times 1$ with the jth element equal to 1 and the remaining elements equal to 0

Apply Data-driven PI for Quadrotor (3)

3.2. Dynamic model

The input signals $u_z, u_\phi, u_\theta, u_\psi$ depend on the rotation speed of the blades as follows:

$$\begin{aligned} u_z &= \omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2 \\ u_\phi &= \omega_2^2 - \omega_4^2 \\ u_\theta &= \omega_1^2 - \omega_3^2 \\ u_\psi &= \omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2 \end{aligned} \tag{13}$$

Where $\omega_j (j = 1, 2, 3, 4)$ are the rotational velocities of the corresponding rotor blades j.

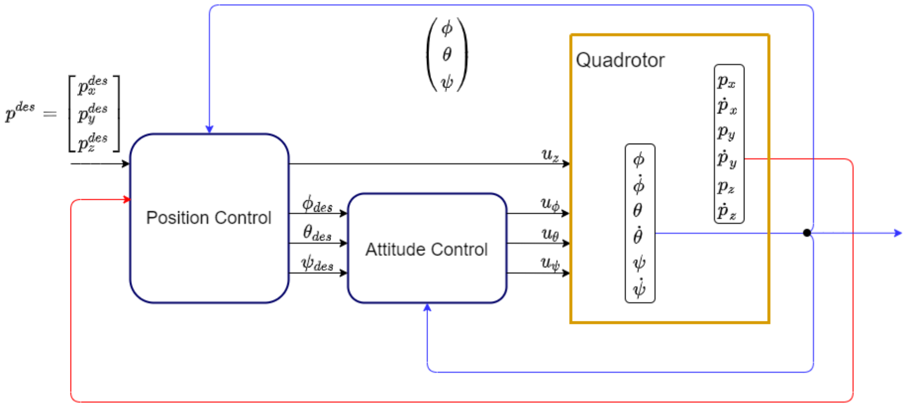
Summary

Quadrotor is a system with:

- 6 degrees of freedom $[p_x, p_y, p_z, \phi, \theta, \psi]^T$
- 4 control inputs $[u_z, u_\phi, u_\theta, u_\psi]^T$

Apply Data-driven PI for Quadrotor (4)

3.3. Typical control scheme for Quadrotor



$p^{des} = \begin{bmatrix} p_x^{des} \\ p_y^{des} \\ p_z^{des} \end{bmatrix}$
$\begin{pmatrix} \phi \\ \theta \\ \psi \end{pmatrix}$
Position Control
Attitude Control
Quadrotor
u_z
ϕ_{des}
θ_{des}
ψ_{des}
u_ϕ
u_θ
u_ψ
ϕ $\dot\phi$ θ $\dot\theta$ ψ $\dot\psi$
p_x $\dot p_x$ p_y $\dot p_y$ p_z $\dot p_z$

Figure: A typical control scheme for Quadrotor

Apply Data-driven PI for Quadrotor (5)

3.3. Typical control scheme for Quadrotor

Control structure

- *Position Controller*: generates a trajectory that sets the desired angles for the Attitude Controller
- *Attitude Controller*: tracks the desired angles set generated by the Position Controller

Feedback measurement

- *Position, velocity and acceleration*: Indoor - Vicon System
- *Euler Angles, Angular Velocity and Angular Acceleration*: IMU Sensor and AHRS Method to estimate Euler angles.
- In addition: the pressure sensor (barometer), position measurements using GPS (Global Positioning System), and saves parameters via black box.

Apply Data-driven PI for Quadrotor (6)

3.4. Data-driven PI algorithm for Quadrotor

Let $x_p = [p_x, \dot{p}_x, p_y, \dot{p}_y, p_z, \dot{p}_z]^T$ and the equation (11) is transformed to:

$$\begin{aligned} \dot{x}_p &= A_p x_p + B_p u_p \\ \dot{x}_{pd} &= A_{pd} x_{pd} \end{aligned} \tag{14}$$

We have:

$$\dot{X}_p = \begin{bmatrix} \dot{e}_p \\ \dot{x}_{pd} \end{bmatrix} = \begin{bmatrix} A_p & A_p - A_{pd} \\ 0_{6\times6} & A_{pd} \end{bmatrix} X_p + \begin{bmatrix} B_p \\ 0_{6\times3} \end{bmatrix} u_p \tag{15}$$

Based on that, define the cost function as follows:

$$\begin{cases} V_p(X_p(t)) = \int_t^\infty e^{-\lambda(\tau-t)}(X_p(\tau)^T Q_p X_p(\tau) + u_p(\tau)^T R_p u_p(\tau))d\tau \\ Q_p = \begin{bmatrix} Q_{ep} & 0_{6\times6} \\ 0_{6\times6} & 0_{6\times6} \end{bmatrix} \in \mathbb{R}^{12\times12}, \quad R_p \in \mathbb{R}^{3\times3} \end{cases}$$

Apply Data-driven PI for Quadrotor (7)

3.4. Data-driven PI algorithm for Quadrotor

After the optimal signal $u_p = [u_{px}, u_{py}, u_{pz}]^T$ is derived from Off-Policy PI algorithm, we can obtain the control signal u_z and the desired heading angles set for the internal control loop as follows (the Yaw angle ψ_d is chosen fixedly):

$$\begin{aligned}
u_z &= \sqrt{u_{px}^2 + u_{py}^2 + (u_{pz} + u_b)^2} \\
\psi_d &= 0 \\
\phi_d &= arcsin(\frac{u_{px}sin(\psi_d) - u_{py}cos(\psi_d)}{u_z}) \\
\theta_d &= arctan(\frac{u_{px}cos(\psi_d) + u_{py}sin(\psi_d)}{u_{pz} + u_b})
\end{aligned} \tag{16}$$

Continue solving similarly for the Attitude Controller after obtaining the desired heading angles set.

Apply Data-driven PI for Quadrotor (8)

3.4. Data-driven PI algorithm for Quadrotor

Let $x_\Theta = [\phi, \dot{\phi}, \theta, \dot{\theta}, \psi, \dot{\psi}]^T$ and the equation (12) is transformed to:

$$\begin{aligned} \dot{x}_\Theta &= F_\Theta x_\Theta + B_\Theta u_\Theta \\ \dot{x}_{\Theta d} &= F_{\Theta d} x_{\Theta d} \end{aligned} \tag{17}$$

Similarly, we have:

$$\dot{X}_{\Theta d} = \begin{bmatrix} \dot{e}_\Theta \\ \dot{x}_{\Theta d} \end{bmatrix} = \begin{bmatrix} F_\Theta & F_\Theta - F_{\Theta d} \\ 0_{6\times 6} & F_{\Theta d} \end{bmatrix} X_{\Theta d} + \begin{bmatrix} B_\Theta \\ 0_{6\times 3} \end{bmatrix} u_\Theta \tag{18}$$

Based on that, define the cost function as follows:

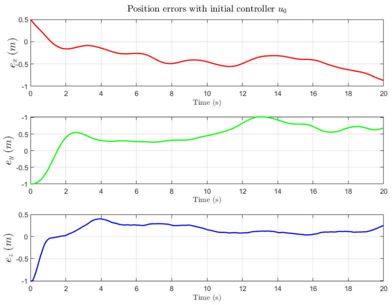
$$\begin{cases} V_\Theta(X_\Theta(t)) = \int_t^\infty e^{-\lambda(\tau - t)}(X_\Theta(\tau)^T Q_e X_\Theta(\tau) + u_\Theta(\tau)^T R_\Theta u_\Theta(\tau))d\tau \\ Q_\Theta = \begin{bmatrix} Q_{e\Theta} & 0_{6\times 6} \\ 0_{6\times 6} & 0_{6\times 6} \end{bmatrix} \in \mathbb{R}^{12\times 12}, \quad R_\Theta \in \mathbb{R}^{3\times 3} \end{cases}$$

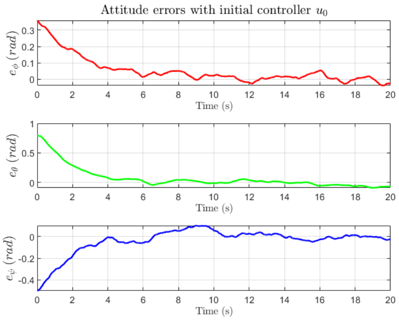
Apply Data-driven PI for Quadrotor (9)

3.5. Results

Simulation parameters

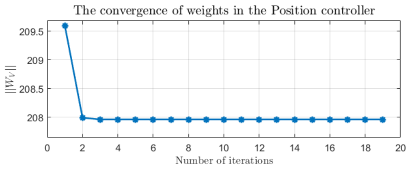
- $m = 2.0\,(kg),\ k_w = 1\,(Ns^2),\ k_t = 1\,(Ns^2/m),\ g = 9.8\,(m/s^2), l_\tau = 0.2\,(m)$ $J = diag(5.1, 5.1, 5.2)\ \ (10^{-3} kg.m^2)$
- The desired spiral trajectory: $p_d = [2sin(at),\ 2cos(at),\ 0.8t]^T$, $a = 0.5$
- The initial admissible controller u_0 is a PID controller with arbitrary K_p, K_i, K_d.

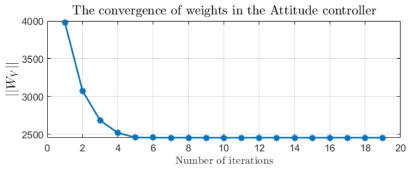


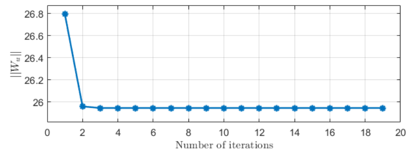


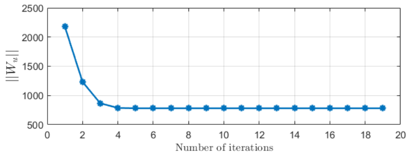
Apply Data-driven PI for Quadrotor (10)

3.5. Results









Apply Data-driven PI for Quadrotor (11)

3.5. Results

The similarity between the result $W_{up}*$ derived from the proposed method and the solution $K*$ solved from the ARE equation:

$$W_{up}^* = \begin{bmatrix} -9.9410 & -0.0000 & 0.0000 \\ -11.8122 & -0.0000 & 0.0000 \\ 0.0000 & -9.9410 & -0.0000 \\ 0.0000 & -11.8122 & 0.0000 \\ -0.0000 & -0.0000 & -9.9410 \\ -0.0000 & -0.0000 & -11.8122 \\ -2.3833 & -4.6813 & 0.0000 \\ -9.3901 & 3.7299 & 0 \\ 4.6813 & -2.3831 & 0.0000 \\ -3.7302 & -9.3941 & 0 \\ -0.0000 & -0.0000 & -0.0000 \\ 0.0000 & 0.0000 & -0.0006 \end{bmatrix}$$

$$K^* = \begin{bmatrix} -9.9401 & 0 & 0 \\ -11.8113 & 0 & 0 \\ 0 & -9.9401 & 0 \\ 0 & -11.8113 & 0 \\ 0 & 0 & -9.9401 \\ 0 & 0 & -11.8113 \\ -2.3843 & -4.6842 & 0 \\ -9.3934 & 3.7319 & 0 \\ 4.6842 & -2.3843 & 0 \\ -3.7319 & -9.3934 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Apply Data-driven PI for Quadrotor (12)

3.5. Results

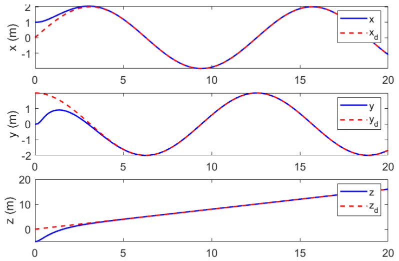


Figure: Tracking position w.r.t desired trajectory

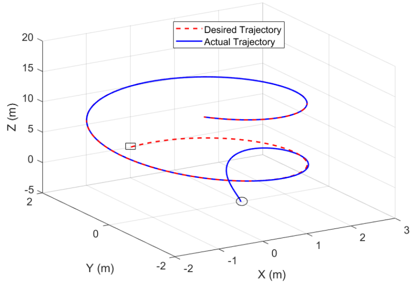


Figure: 3D tracking for spiral trajectory

Apply Data-driven PI for Quadrotor (13)

3.5. Results

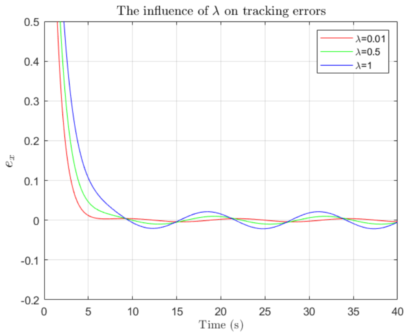


Figure: The influence of λ on position tracking errors

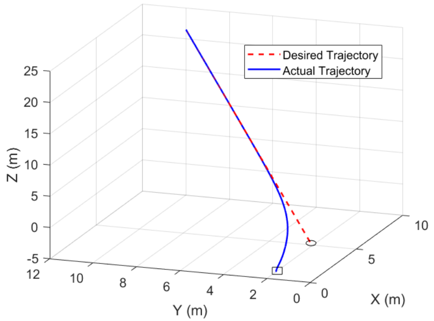


Figure: 3D tracking for a straight trajectory

Conclusion

Drawbacks of the project

- The problem does not consider the influence of noise
- The problem does not consider control input constraints

Development direction

- Using additional observers and noise compensation, research to solve the problem with the HJI equation [4] (Hamilton- Jacobi- Isaac equation).
- Consider the influence of input constraints and handle them by adjusting the cost function.
- Combine with multi-quadrotors formation control to solve many more highly applicable tasks, in which knowing the mathematical model of the controlled object is difficult to achieve

References

Rushikesh Kamalapurkar, Huyen Dinh, Shubhendu Bhasin, and Warren E. Dixon.
Approximate optimal trajectory tracking for continuous-time nonlinear systems.
Automatica, 51:40–48, 2015.

Hao Liu, Xiafu Wang, and Yisheng Zhong.
Quaternion-based robust attitude control for uncertain robotic quadrotors.
IEEE Transactions on Industrial Informatics, 11:1–1, 04 2015.

Daniel Mellinger and Vijay Kumar.
Minimum snap trajectory generation and control for quadrotors.
In *2011 IEEE International Conference on Robotics and Automation*, pages 2520–2525, 2011.

Hamidreza Modares, Frank L. Lewis, and Zhong-Ping Jiang.
H_∞ tracking control of completely unknown continuous-time systems via off-policy reinforcement learning.
IEEE Transactions on Neural Networks and Learning Systems, 26(10):2550–2562, 2015.

Tsung-Wei Ou Yen-Chen Liu.
Non-linear adaptive tracking control for quadrotor aerial robots under uncertain dynamics.
IET Control Theory and Applications, 2021.

Wanbing Zhao, Hao Liu, Frank L. Lewis, and Xinlong Wang.
Data-driven optimal formation control for quadrotor team with unknown dynamics.
IEEE Transactions on Cybernetics, pages 1–10, 2021.

Thank you