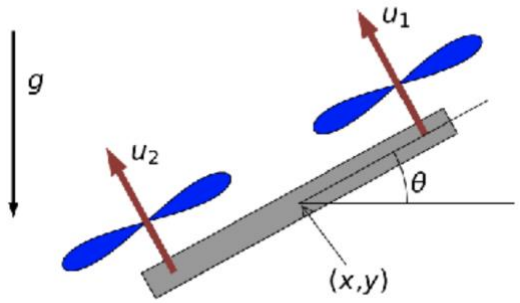
N16708895 MENGYA QIU

2D Quadrotor Trajectory Planning

1. The 2d quadrotor model



The dynamics of a 2D quadrotor can be written as:

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}v_x\\ v_y\\ \omega\end{bmatrix}, \quad \begin{bmatrix}\ddot{x}\\ \ddot{y}\\ \ddot{\theta}\end{bmatrix} = \begin{bmatrix}0\\ -g\\ 0\end{bmatrix} + \begin{bmatrix}\frac{-1}{m}\sin\theta & 0\\ \frac{1}{m}\cos\theta & 0\\ 0 & \frac{1}{I_{xx}}\end{bmatrix} \cdot \begin{bmatrix}u_1\\ u_2\end{bmatrix}$$

$$\dot{z} = \begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\\ \ddot{x}\\ \ddot{y}\\ \ddot{\theta}\end{bmatrix} = \begin{bmatrix}v_x\\ v_y\\ \omega\\ -\frac{u_1}{m}\sin\theta\\ -g + \frac{u_1}{m}\cos\theta\\ \frac{u_2}{mr^2}\end{bmatrix} \qquad (1)$$

There are 6 states: the position on the plane (x, y), the linear velocity (v_x, v_y) , the angle of the quadrotor θ and its angular velocity ω. There are two control inputs (u_1, u_2) which are the thrust forces applied to the quadrotor. Thus, the state vector can be written as: $z = \lceil x \;\; y \;\; \theta \;\; v_x \;\; v_y \;\; \omega \rceil^T$.

We know that the quadrotor's dynamics is nonlinear. We start with an equilibrium hover configuration: $x_0, y_0, \theta_0 = 0, u_{1,0} = mg, u_{2,0} = 0$. That is, we need to stabilize the nonlinear system around the fixed point $\bar{z} = \lceil \bar{x} \;\; \bar{y} \;\; \bar{\theta} \;\; \bar{v}_x \;\; \bar{v}_y \;\; \bar{\omega} \rceil^T = \lceil 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \rceil^T, \bar{u} = \lceil mg \;\; 0 \rceil^T$.

We will first linearize the system around $\bar{z}$ (with $\bar{u} = \lceil mg \;\; 0 \rceil^T$) and then compute a LQR control law to stabilize it. The control law will be approximately optimal for the nonlinear system but should be a good enough approximation when sufficiently close to the fixed point.

2. Linearizing the dynamics

To linearize the dynamics, here we replace all non-linear functions of the state and control variables with their first order Taylor expansion at the equilibrium location $\bar{z}$. In this case, the non-linear functions are $\sin\theta$ and $\cos\theta$. Near $\theta = 0$, $\sin\theta \approx \theta$ and $\cos\theta \approx 1$. Hence the linearized dynamics are given by:

$$\begin{bmatrix}\ddot{x}\\ \ddot{y}\\ \ddot{\theta}\end{bmatrix} = \begin{bmatrix}-g\theta\\ -g\\ 0\end{bmatrix} + \begin{bmatrix}0 & 0\\ \frac{1}{m} & 0\\ 0 & \frac{1}{I_{xx}}\end{bmatrix} \cdot \begin{bmatrix}u_1\\ u_2\end{bmatrix}$$

$$\Longrightarrow \dot{z} = \begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\\ \ddot{x}\\ \ddot{y}\\ \ddot{\theta}\end{bmatrix} = \begin{bmatrix}v_x\\ v_y\\ \omega\\ -g\theta\\ -g + \frac{u_1}{m}\\ \frac{u_2}{mr^2}\end{bmatrix} \tag{2}$$

We linearize $\dot{z} = f(z,u) \simeq f(\bar{z},\bar{u}) + \frac{\partial f}{\partial z}|_{\substack{z=\bar{z}\\ u=\bar{u}}}(z-\bar{z}) + \frac{\partial f}{\partial u}|_{\substack{z=\bar{z}\\ u=\bar{u}}}(u-\bar{u})$.
Note that $f(\bar{z},\bar{u}) = 0$.

Next, we compute the partial derivatives of the dynamics with respect to all state and control variables.

First for states:

$\frac{\partial}{\partial \mathrm{x}}\mathrm{v_x} = 0, \frac{\partial}{\partial \mathrm{y}}\mathrm{v_x} = 0\ , \frac{\partial}{\partial \theta}\mathrm{v_x} = 0\ , \frac{\partial}{\partial \mathrm{v_x}}\mathrm{v_x} = 1\ , \frac{\partial}{\partial \mathrm{v_y}}\mathrm{v_x} = 0\ , \frac{\partial}{\partial \omega}\mathrm{v_x} = 0$

$\frac{\partial}{\partial \mathrm{x}}\mathrm{v_y} = 0, \frac{\partial}{\partial \mathrm{y}}\mathrm{v_y} = 0\ , \frac{\partial}{\partial \theta}\mathrm{v_y} = 0\ , \frac{\partial}{\partial \mathrm{v_x}}\mathrm{v_y} = 0\ , \frac{\partial}{\partial \mathrm{v_y}}\mathrm{v_y} = 1\ , \frac{\partial}{\partial \omega}\mathrm{v_y} = 0$

$\frac{\partial}{\partial \mathrm{x}}\omega = 0, \frac{\partial}{\partial \mathrm{y}}\omega = 0\ , \frac{\partial}{\partial \theta}\omega = 0\ , \frac{\partial}{\partial \mathrm{v_x}}\omega = 0\ , \frac{\partial}{\partial \mathrm{v_y}}\omega = 1\ , \frac{\partial}{\partial \omega}\omega = 1$

$\frac{\partial}{\partial \mathrm{x}}\ddot{\mathrm{x}} = 0, \frac{\partial}{\partial \mathrm{y}}(-\mathrm{g}\theta) = 0\ , \frac{\partial}{\partial \theta}(-\mathrm{g}\theta) = -\mathrm{g}\ , \frac{\partial}{\partial \mathrm{v_x}}(-\mathrm{g}\theta) = 0\ , \frac{\partial}{\partial \mathrm{v_y}}(-\mathrm{g}\theta) = 0\ , \frac{\partial}{\partial \omega}(-\mathrm{g}\theta) = 0$

$\frac{\partial}{\partial \mathrm{x}}\ddot{\mathrm{y}} = 0, \frac{\partial}{\partial \mathrm{y}}(-\mathrm{g} + \frac{\mathrm{u_1}}{\mathrm{m}}) = 0\ , \frac{\partial}{\partial \theta}\left(-\mathrm{g} + \frac{\mathrm{u_1}}{\mathrm{m}}\right) = 0\ , \frac{\partial}{\partial \mathrm{v_x}}(-\mathrm{g} + \frac{\mathrm{u_1}}{\mathrm{m}}) = 0\ , \frac{\partial}{\partial \mathrm{v_y}}(-\mathrm{g} + \frac{\mathrm{u_1}}{\mathrm{m}}) = 0\ , \frac{\partial}{\partial \omega}(-\mathrm{g} + \frac{\mathrm{u_1}}{\mathrm{m}}) = 0$

$\frac{\partial}{\partial \mathrm{x}}\ddot{\theta} = 0, \frac{\partial}{\partial \mathrm{y}}(\frac{\mathrm{u_2}}{\mathrm{mr^2}}) = 0\ , \frac{\partial}{\partial \theta}\left(\frac{\mathrm{u_2}}{\mathrm{mr^2}}\right) = 0\ , \frac{\partial}{\partial \mathrm{v_x}}(\frac{\mathrm{u_2}}{\mathrm{mr^2}}) = 0\ , \frac{\partial}{\partial \mathrm{v_y}}(\frac{\mathrm{u_2}}{\mathrm{mr^2}}) = 0\ , \frac{\partial}{\partial \omega}(\frac{\mathrm{u_2}}{\mathrm{mr^2}}) = 0$

Then for control command:

$\frac{\partial}{\partial u_1}v_x = 0, \frac{\partial}{\partial u_1}v_y = 0\ , \frac{\partial}{\partial u_1}\omega = 0\ , \frac{\partial}{\partial u_1}(-g\theta) = 0\ , \frac{\partial}{\partial u_1}(-g + \frac{u_1}{m}) = \frac{1}{m}, \frac{\partial}{\partial u_1}(\frac{u_2}{mr^2}) = 0$

$\frac{\partial}{\partial u_2}v_x = 0, \frac{\partial}{\partial u_2}v_y = 0\ , \frac{\partial}{\partial u_2}\omega = 0\ , \frac{\partial}{\partial u_2}(-g\theta) = 0\ , \frac{\partial}{\partial u_2}(-g + \frac{u_1}{m}) = 0\ , \frac{\partial}{\partial u_2}(\frac{u_2}{mr^2}) = \frac{1}{mr^2}$

$$\Longrightarrow\ \dot{z} - f(\bar{z},\bar{u}) = \begin{bmatrix}0 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0 & 0 & 1\\ 0 & 0 & -g & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix}(z-\bar{z}) + \begin{bmatrix}0 & 0\\ 0 & 0\\ 0 & 0\\ 0 & 0\\ \frac{1}{m} & 0\\ 0 & \frac{1}{mr^2}\end{bmatrix}(u-\bar{u}) \tag{3}$$

Finally, we change coordinates to center around the linearized point, we set $\tilde{z} = z - \bar{z}$ and $\tilde{u} = u - \bar{u}$ and get the following dynamics.

$$\Longrightarrow \dot{\tilde{z}} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -g & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tilde{z} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{m} & 0 \\ 0 & \frac{1}{mr^2} \end{bmatrix} \tilde{u} \tag{4}$$

3. Discretizing the dynamics

Using Δt as the integration step, we have:

$$\tilde{z}_{n+1} = \tilde{z}_n + \Delta t \left(\begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -g & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tilde{z}_n + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{m} & 0 \\ 0 & \frac{1}{mr^2} \end{bmatrix} \tilde{u}_n \right)$$

Or equivalently,

$$\tilde{z}_{n+1} = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & -g\Delta t & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tilde{z}_n + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{\Delta t}{m} & 0 \\ 0 & \frac{\Delta t}{mr^2} \end{bmatrix} \tilde{u}_n \tag{5}$$

We now have an approximation of the dynamics in a form that can be used to compute linear optimal controllers (i.e. LQR). The controllers we will compute will be of the form: $\tilde{u}_n = K_n \tilde{z}_n + k_n$; and they will need to be transformed into $u_n = \bar{u}_n + K_n(z_n - \bar{z}_n) + k_n$ to be used on the original system.

For original discretized system dynamics, it has iteration:

$$z_{n+1} = z_n + \dot{z}_n \cdot \Delta t$$

More specifically:

$$\begin{aligned} x_{n+1} &= x_n + \Delta t \cdot v_{x_n} \\ y_{n+1} &= y_n + \Delta t \cdot v_{y_n} \\ \theta_{n+1} &= \theta_n + \Delta t \cdot \omega_n \\ v_{x_{n+1}} &= v_{x_n} + \Delta t \cdot \left(-\frac{u_1}{m} \sin\theta\right) \\ v_{y_{n+1}} &= v_{y_n} + \Delta t \cdot \left(-g + \frac{u_1}{m} \cos\theta\right) \\ \omega_{n+1} &= \omega_n + \Delta t \cdot \left(\frac{u_2}{mr^2}\right) \end{aligned} \tag{6}$$

4. Tracking problems

Basically, if we want a system with linear dynamics and quadratic cost function to move along a certain path, which is also known as Linear-Quadratic tracking problems. A very straightforward and often used method for LQ problems is to construct an LQR controller with feedforward, which is useful but it can only derive a global optimal control solution without any restrictions. To deal with LQ problems with constraints, a variation of LQR is derived as Quadratic Programs (QP solver) that could solve a set of equalities and inequalities at the same time and converge relatively fast. Therefore, in this experiment merely 2 methods are used to track the 2d quadrotor model along the designated paths.

The tracked states are selected as $z_{des} = \left[\mathrm{x_n},\ \mathrm{y_n},\ 0,\ \mathrm{v_{x_n}},\ \mathrm{v_{y_n}},\ 0\right]^{\mathrm{T}}$, that is, plane positions and velocities of the quadrotor will be emphasized and regulated. Hence, the corresponding $x_n, y_n, v_{x_n}, v_{y_n}$ will be calculated and given for each time step. Here I created 4 different trajectories and the time step is del_t = 0.01 for each trajectory.

4.1 LQR feedforward controller for tracking problems

Given a planned trajectory z_{des}, the cost function could be rewritten as:

$$\min_{\mathrm{u_n}}(\mathrm{z_N}-\mathrm{z_{des}})^{\mathrm{T}}\mathrm{Q_N}(\mathrm{z_N}-\mathrm{z_{des}})+\sum_{\mathrm{n=0}}^{\mathrm{N-1}}(\mathrm{z_n}-\mathrm{z_{des}})^{\mathrm{T}}\mathrm{Q_n}(\mathrm{z_n}-\mathrm{z_{des}})+\mathrm{u_n}\mathrm{R_n}\mathrm{u_n}$$

$$\Longrightarrow \min_{\mathrm{u_n}} \mathrm{z_N}^{\mathrm{T}}\mathrm{Q_N}\,\mathrm{z_N}+q_N^T\mathrm{z_N}+\sum_{\mathrm{n=0}}^{\mathrm{N-1}}\mathrm{z_n}^{\mathrm{T}}\mathrm{Q_n}\,\mathrm{z_n}+q_n^T\mathrm{z_n}+\mathrm{u_n}\mathrm{R_n}\mathrm{u_n}$$

where $\mathrm{q_n} = -\mathrm{Q_n}\,\mathrm{z_{des}}$

$$\text{s.t} \qquad \mathrm{z_{n+1}} = A_n\mathrm{z_n} + B_n\mathrm{u_n}$$

where A_n, B_n have given by formula (5)

Thus, solving the tracking problem is to solve the Riccati equations below:

First initialize $\mathrm{P_N} = \mathrm{Q_N}$, $\mathrm{p_N} = \mathrm{q_N}$

Then iterate from N-1 to 0：

$$\mathrm{K_n} = -(R_n + B_n^T P_{n+1} B_n)^{-1} B_n^T P_{n+1} A_n$$

$$P_n = \mathrm{Q_n} + A_n^T P_{n+1} A_n + A_n^T P_{n+1} B_n \mathrm{K_n}$$

$$k_n = -(R_n + B_n^T P_{n+1} B_n)^{-1} B_n^T p_{n+1}$$

$$p_n = \mathrm{q_n} + A_n^T p_{n+1} + A_n^T P_{n+1} B_n \mathrm{k_n}$$

Finally a global optimal policy： $\mu_n^*(z_n) = \mathrm{K_n} z_n + \mathrm{k_n}$.

Note that $\mathrm{K_n}$ are feedback gains while $\mathrm{k_n}$ are feedforward controls (only linear dynamics or linearized nonlinear dynamics could apply linear LQR controller.) Also note that the optimal controls $\mu_n^*(\mathrm{z_n})$ are derived using the linearized approximation system dynamics given by formula (5). However, when use $\mu_n^*(\mathrm{z_n})$ to simulate the original system dynamics it is formula (6) should be used. The actual control inputs for the original dynamics need to be transformed into the form (note that $\bar{\mathrm{z}}_\mathrm{n} = 0$)：

$$u_n = \bar{u}_n + K_n(z_n - \bar{z}_n) + k_n$$
$$= [mg \;\; 0]^T + K_n(z_n - z_{des}) + k_n$$

4.2 Solving quadratic programs with constraints

With constraints, the optimization problem can be constructed as the form:

$$\min_{u_n} z_N{}^T Q_N z_N + q_N^T z_N + \sum_{n=0}^{N-1} z_n{}^T Q_n z_n + q_n^T z_n + u_n R_n u_n$$

where $q_n = -Q_n z_{des}$

$$\text{s.t} \quad z_{n+1} = A_n z_n + B_n u_n$$
$$G_n z_n \le g_n$$
$$H_n u_n \le h_n$$
$$z_0 = z_{initial}$$

$$\Longrightarrow \quad \min_{\tilde{z},\tilde{u}} \tilde{z}^T \begin{bmatrix} Q_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & Q_N \end{bmatrix} \tilde{z} + \tilde{u}^T \begin{bmatrix} R_0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & R_{N-1} \end{bmatrix} \tilde{u} + \begin{bmatrix} q_1 \\ \vdots \\ q_N \end{bmatrix}^T \tilde{z} + (z_0^T Q_0 z_0 + q_0^T z_0)$$

$$\text{s.t} \quad \begin{bmatrix} -1 & 0 & \cdots & 0 & 0 \\ A_1 & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & A_{N-2} & -1 & 0 \\ 0 & 0 & \cdots & A_{N-1} & -1 \end{bmatrix} \tilde{z} + \begin{bmatrix} B_0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & B_{N-1} \end{bmatrix} \tilde{u} = \begin{bmatrix} -Az_0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} G_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & G_N \end{bmatrix} \tilde{z} \le \begin{bmatrix} g_1 \\ \vdots \\ g_N \end{bmatrix} ,$$

$$\begin{bmatrix} H_0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & H_{N-1} \end{bmatrix} \tilde{u} \le \begin{bmatrix} h_0 \\ \vdots \\ h_{N-1} \end{bmatrix}$$

$$\Longrightarrow \min_{\tilde{z},\tilde{u}} \begin{bmatrix} \tilde{z} \\ \tilde{u} \end{bmatrix}^T \begin{bmatrix} \tilde{Q} & 0 \\ 0 & \tilde{R} \end{bmatrix} \begin{bmatrix} \tilde{z} \\ \tilde{u} \end{bmatrix} + \begin{bmatrix} \tilde{q} \\ 0 \end{bmatrix}^T \begin{bmatrix} \tilde{z} \\ \tilde{u} \end{bmatrix}$$

$$\text{s.t} \quad [\tilde{A} \;\; \tilde{B}] \begin{bmatrix} \tilde{z} \\ \tilde{u} \end{bmatrix} = \tilde{a}$$

$$\begin{bmatrix} \tilde{G} & 0 \\ 0 & \tilde{H} \end{bmatrix} \begin{bmatrix} \tilde{z} \\ \tilde{u} \end{bmatrix} \le \begin{bmatrix} \tilde{g} \\ \tilde{h} \end{bmatrix}$$

Thus, the tracking problem turns to a simple Quadratic Program that can be solved with any QP solver of the form:

$$\min_x x^T Q x + q^T x$$
$$\text{s.t} \quad Ax = b$$
$$Gx \le h$$

Note that in this experiment A, B, Q, R are all constant where Q > 0, R > 0. A QP solver of this form is also known as a direct transcription approach.

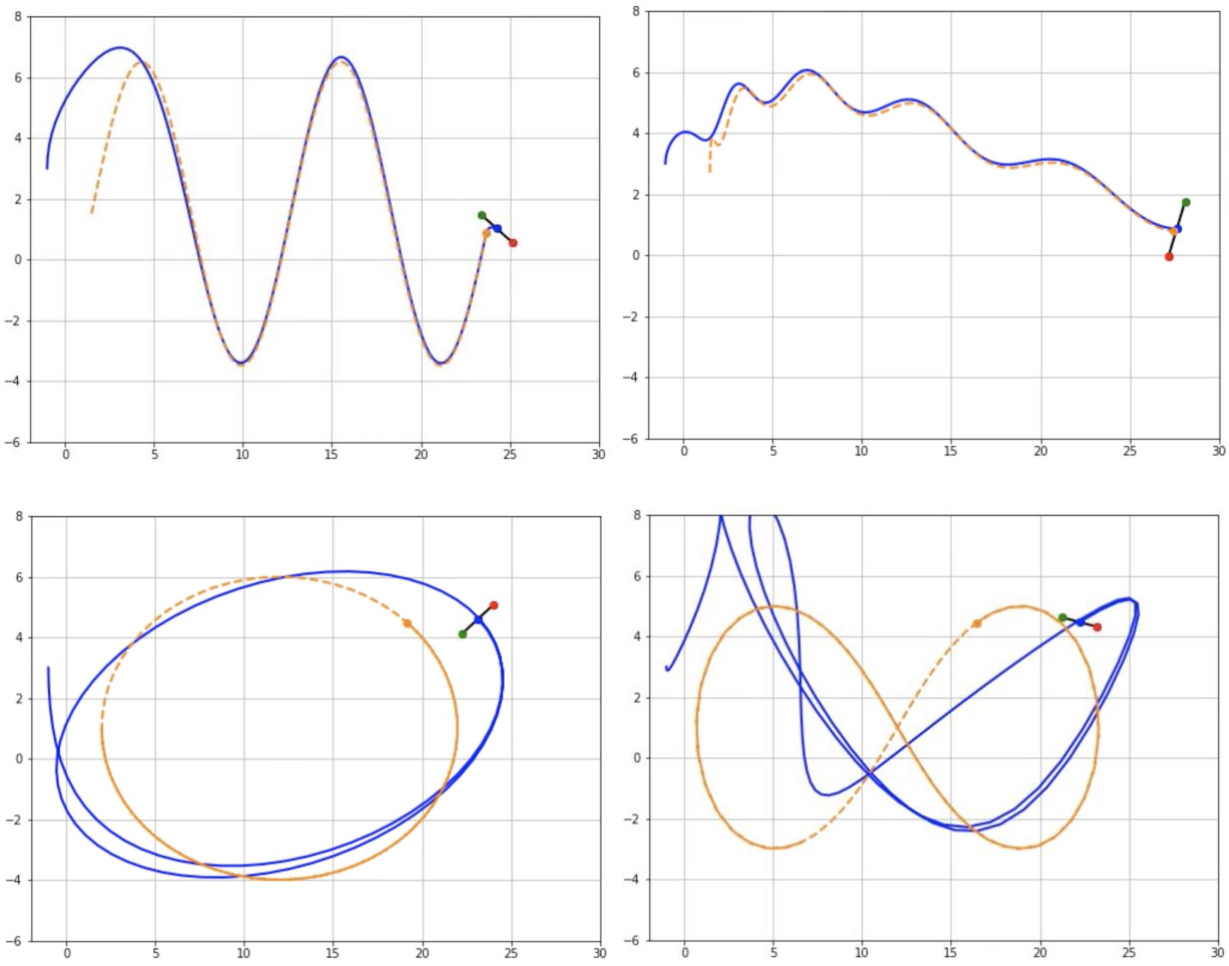
5. Simulation Results

Here I apply LQR and QP algorithms respectively to 4 different trajectories and get results.

Four trajectories were created in total for this experiment:

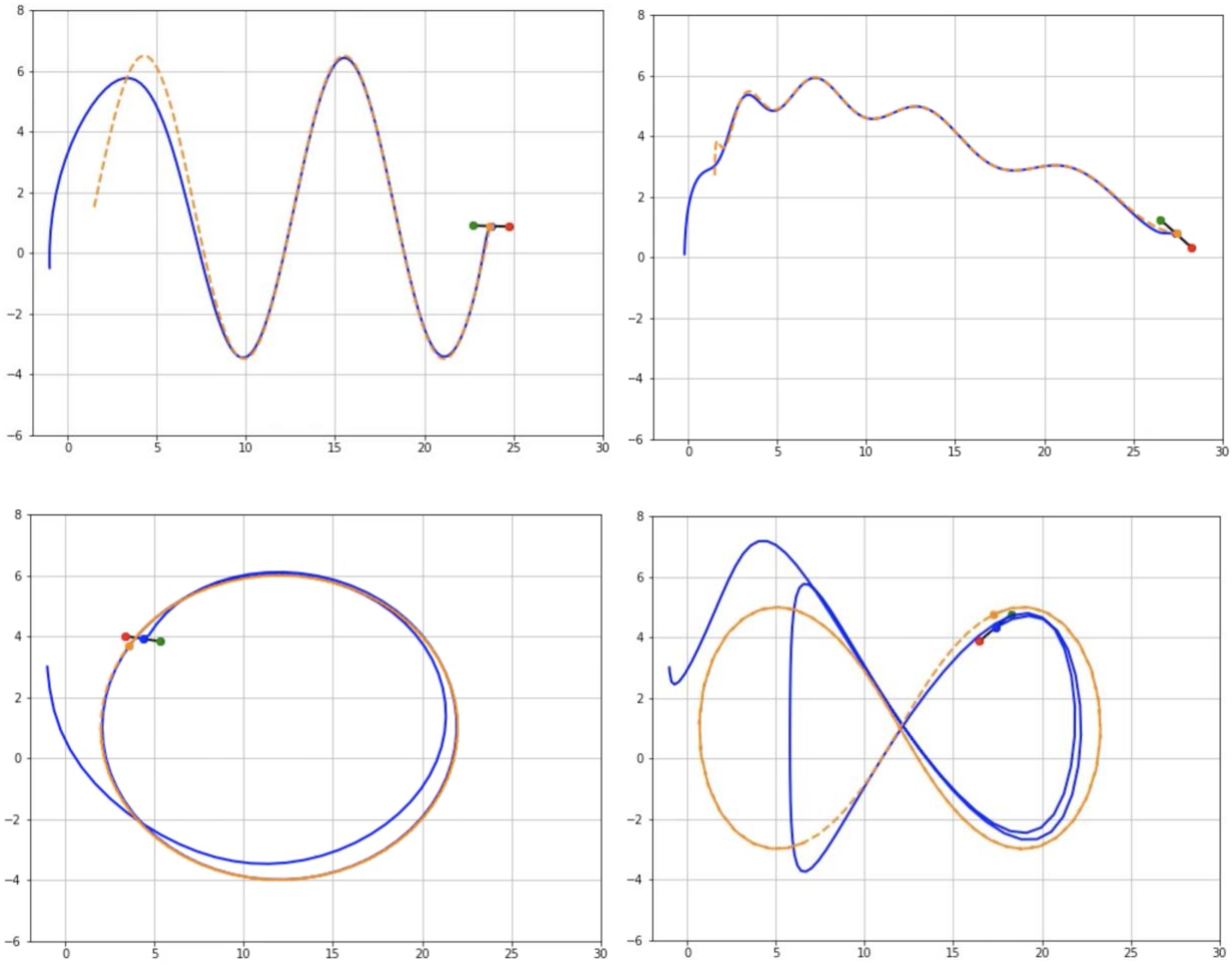
1. A sinusoidal curve
2. An unregular curve
3. An ellipse
4. A "8" curve (Lemniscate/Bernoulli curve)

5.1 Linear Quadratic Regulator



Note that in this experiment the LQR controllers with only feedback controls had much better performance than LQR controllers with feedforward gains. We can observe that the computed feedforward gains had a magnitude of 10^3 -- a 2d quadrotor with a light weight can easily be “throwed” away from its original trajectory – therefore they need to be scaled down and tuned by multiplying a discount factor.

5.2 Quadratic Program Solver



The QP solver achieved even better tracking performance than LQR controllers despite under more constrictions.

As the planned trajectories gets more complex the tracking performance faded for both linear solutions.

(More details see 2dQuadrotor_1.ipynb & 2dQuadrotor_2.ipynb)