

## MPC迅速簡介

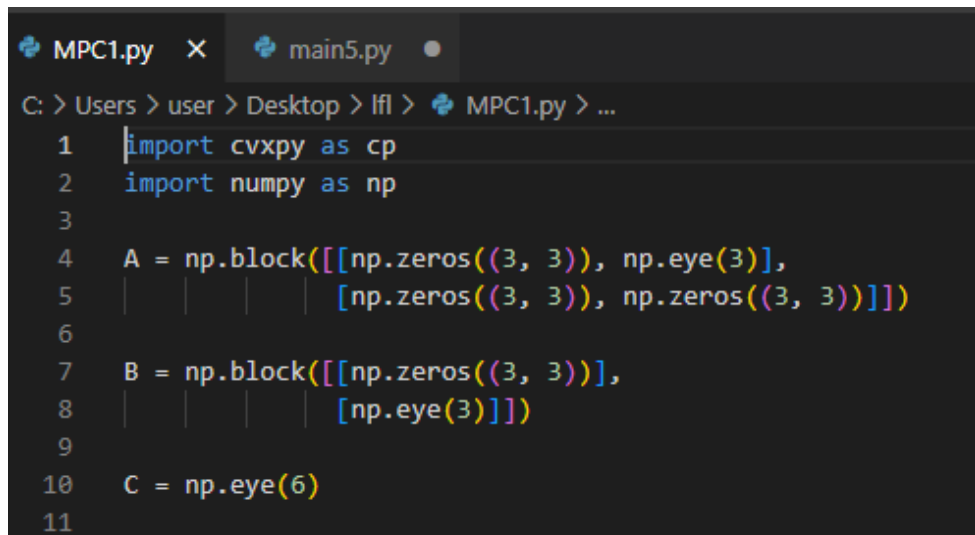
這個部分就可以完成MPC模型的控制

MPC又稱為模型預測控制我將分開為您簡介

MPC是一種控制器，可以將輸入模型的狀態，作為控制器的輸入，使用控制器的目標函數(公式)來對模型進行約束

模型=無人機 =  $x(t+1) = Ax(t) + Bu(t)$  = 這個數學式，我們會將無人機變成一個線性方程，稱為模型。

其中我們可以自行定義A B u等(我已經在MPC1.py中完成模型建立也確認可行，此部分只需要複製MPC1.py

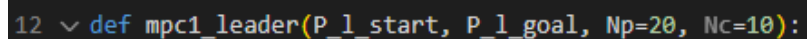


```
1 import cvxpy as cp
2 import numpy as np
3
4 A = np.block([[np.zeros((3, 3)), np.eye(3)],
5               [np.zeros((3, 3)), np.zeros((3, 3))]])
6
7 B = np.block([[np.zeros((3, 3))],
8               [np.eye(3)]])
9
10 C = np.eye(6)
11
```

模型說完後接著說預測

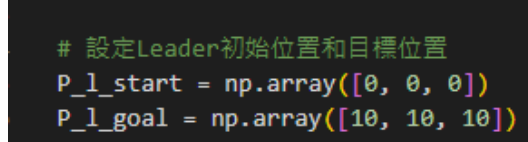
MPC控制器會有兩個時間控制區間Nc(控制器要運算的時間)和預測區間Ns(實際整個控制器運行的時間)

可以看到目標函數中有對應的 Nc 和 Np



```
12 def mpc1_leader(P_l_start, P_l_goal, Np=20, Nc=10):
```

在定義 MPC 控制器的初期就要將會用到的參數加入其中 ( 在 main 中定義如 P\_l\_start 和 P\_l\_goal ) 以及定義該 MPC 控制器的 Nc 和 Np



```
# 設定Leader初始位置和目標位置
P_l_start = np.array([0, 0, 0])
P_l_goal = np.array([10, 10, 10])
```

飛機的狀態因為是不斷更新的，所以要在main中定義，再於mpc控制器中呼叫做控制

```

13     # 定義狀態
14     u = cp.Variable((3, Np)) # 控制輸入 (速度)
15     P_l = cp.Variable((3, Np+1)) # Leader 位置 (x, y, z)
16
17     # 最小化 Leader 到目標點的距離跟權重
18     Q = np.eye(3)
19     R = 0.1 * np.eye(3)
20

```

這部分一樣是可以照抄的。

接著才是MPC不一樣且重要的地方目標函數

只需要將我給的數學式用程式語言表示即可

```

# 目標函數
cost = 0
for k in range(Np):
    cost += cp.quad_form(P_l[:, k] - P_l_goal, Q) + cp.quad_form(u[:, k], R)

```

MPC1 的目標函數為  $J_{l,N}(P_{l,G}, u_l, k) = \sum_{s=1}^{N_P} (P_{l,G}^T(k+s|k) Q P_{l,G}(k+s|k)) +$

$\sum_{m=1}^{N_c} (u_l(k+m|k) R u_l(k+m|k))$

這個部分每一個MPC控制器不一樣，所以要根據我給的數學式下去編寫

```

# 約束條件
constraints = [P_l[:, 0] == P_l_start]
for k in range(Np):
    constraints += [P_l[:, k+1] == P_l[:, k] + u[:, k]]

# 求解最佳化
prob = cp.Problem(cp.Minimize(cost), constraints)
prob.solve()

return P_l.value, u.value # 回到位置和控制輸入

```

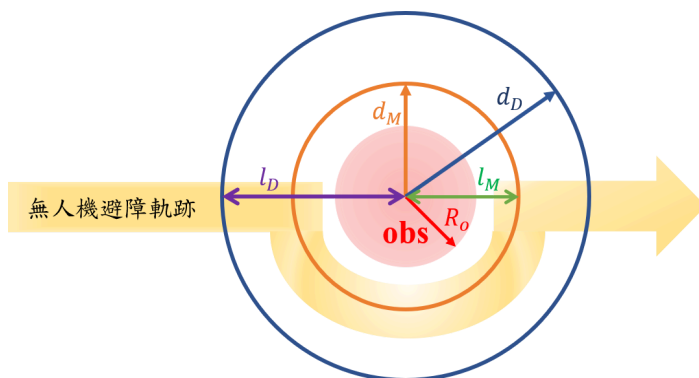
約束條件根據每一個控制器要做的事情來編寫

像我第一个控制器的功能是让Leader的起始點作為目標點的起始點，在模型內PL每一次更新狀態都會回傳後再次求解。

以爬樓梯作為比喻，每爬一格他就會回報他在第幾格(X,Y,Z)三維座標

以上部分是我搭配程式對 MPC 的簡介，希望對您有幫助。

### MPC3 領導者和障礙物(obs)避障



以下為對於無人機進行障礙物避障時的條件設定

UAV示意為Leader無人機

$l$ ：障礙物安全半徑(危險區域)

$l$ ：無人機避障理想軌跡半徑

$d$ ：障礙物的安全半徑的長度

$R$ ：障礙物半徑

設：障礙物的座標為  $(x,y,z)$

領導者無人機的座標為 (,,)

$$(k) = \sqrt{\left((k) - x_o\right)^2 + \left((k) - y_o\right)^2 + \left((k) - z_o\right)^2} - R_o$$

$$(k+s|k) = \sqrt{\left((k+s|k) - x_o\right)^2 + \left((k+s|k) - y_o\right)^2 + \left((k+s|k) - z_o\right)^2} - R_o$$

$$\text{目標函數 } L_o(P_l, k) = \begin{cases} 0 & (k) > l_D - (1) \\ \sum_{s=1}^{N_c} -a \left( (k+sk) - l_m \right) & (k) < l_D - (2) \end{cases}$$

：Leader的座標和obs的相對距離-障礙物半徑

障礙物的避障範圍沒有無人機(領導者)

障礙物的避障範圍出現無人機

功能：當障礙物範圍內出現UAV，目標函數最小化Leader和障礙物間的距離，但保持一個 $l_m$ 的關係Leader避障

### MPC3的解釋

一開始你要在main中定義一個三維的球體作為障礙物( (x,y,z) )，這個障礙物本身的長度就是 $R_o$ ，然後他的x座標是要隨機生成的(一開始可以先自行設定，確定避障功能後再用隨機生成他的xyz座標)，預期成果是至少有兩個障礙物(可以先生成一個在leader必經的路線(n,n,n), n=3~7(因最短路徑必為直線，所以生成在直線路徑中一定需要進行避障)，後續避障功能順利再隨機生成障礙物位置即可(main中定義後要在MPC3定義時呼叫障礙物)

MPC3的目標函數裡面寫(2) a可以自行選擇如0.5-1.5就可以，在main中要做的事情如下

定義領導者位置(已完成)

定義障礙物位置(尚未完成)

主要中的判斷式

判斷領導者和障礙物之間的相對距離式

的情況:leader跟障礙物的距離大於 $d_D = l_D$ ，數字相同但定義不同，在main(自行定義數字，如3 則main正常運行

的情況領導者跟障礙物的距離小於 $D_d$ ，則要啟動MPC3，公式中的 $l_m = d_M$ 可定義為2，本身的定義也就照著

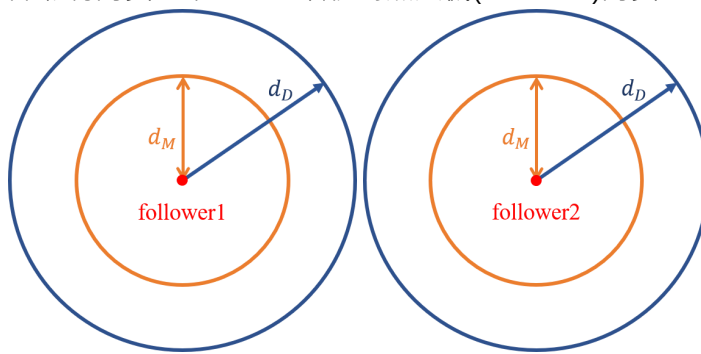
$$d(k) = \sqrt{\left((k) - x_o\right)^2 + \left((k) - y_o\right)^2 + \left((k) - z_o\right)^2} - R_o$$
 定義寫就好了， $R_o$ 就是障礙物的半徑，上面在定義障礙物就會用到

這些東西定義好後在定義MPC3時呼叫就可以。

MPC3的約束條件參考MPC1，用到的是Leader 大致上應該是這樣。

#### MPC4 follower1和follower2之間的避障

障礙物的安全半徑也可以視為無人機(follower)的安全半徑(防碰撞距離)。



$d_M$ ：無人機安全距離

以  $P$  表示追隨者1，中心點座標為  $(x, y)$

以  $P$  表示追隨者2，中心點座標為  $(x, y)$

如果follower1和follower2之間的距離小於  $d$ ，則兩者間需要避障

$$d_{12}(k) = \sqrt{(-)^2 + (-)^2 + (-)^2}$$

$$\text{目標函數 } L_p(P_1, P_2, k) = \begin{cases} 0 & d_{12}(k) > 2d_D - (1) \\ \sum_{s=1}^{NS} -b(d_{12}(k+sk) - 2d_M) & d_{12}(k) < 2d_D - (2) \end{cases}$$

follower1和follower2的距離大於安全半徑，不需避障

follower1和follower2的距離小於安全半徑，需要避障

功能 follower1 和 follower2 之間的距離最少要保持兩個  $d_M$  的距離，以免碰撞

#### MPC4的介紹

這裡是為了避免追隨者在避障過程中會有產生碰撞的機會，所以要有MPC4 (MPC4在main中的權重要比MPC2大)(因為我的MPC2是以向量跟隨leader，有可能leader避障成功但follower相撞或有相關問題)，可將mpc2中的約束條件更改為軟性約束(先完成避障後若有發生運行不了的問題再更動此部分)

MPC4的功能是防止碰撞

目標函數一樣是輸入(2)，b一樣為自行定義

其余dD和dM在main中階已經定義過，引用即可

在mian中要定義d12，後於MPC4定義時呼叫即可 $d(k) = \sqrt{(-)^2 + (-)^2 + (-)^2}$

主中若f1和f2的相對距離小於2個dM則需要啟動MPC4

MPC4中的約束條件則參考MPC2，約束Follower

以上希望能幫助到您謝謝

