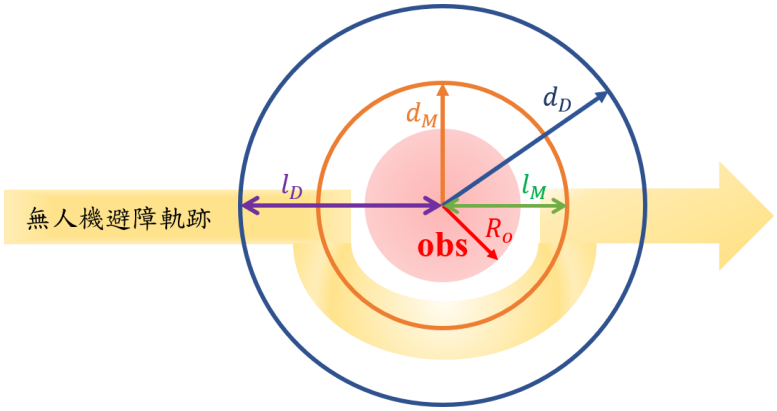
MPC3　Leader 和障礙物(obs)避障



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_D：障碍物安全半径(危险区域)

l_M：无人机避障理想轨迹半径

d_D：障碍物的安全半径的长度

R_o：障碍物半径

设：障碍物的座标为(x_o, y_o, z_o)

Leader 无人机的座标为(P_l^x, P_l^y, P_l^z)

$$l_l^S(k) = \sqrt{(P_l^x(k) - x_o)^2 + \left(P_l^y(k) - y_o\right)^2 + (P_l^z(k) - z_o)^2} - R_o$$

$$l_l^S(k+s|k)$$

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

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

l_i^S：Leader 的座標和 obs 的相對距離-障礙物半徑

(1) 障碍物的避障范围没有 UAV(Leader)

(2) 障碍物的避障范围出现 UAV

功能 • 当障碍物范围内出现 UAV，目标函数最小化 Leader 和障碍物间的距离，但保持一个 l_M 的关系 Leader 避障

MPC3 的解释

一开始你要在 main 中定义一个三维的球体作为障碍物((x_o, y_o, z_o)，这个障碍物本身的长度就是 Ro，然后他的 x 座标是要随机生成的(一开始可以先自行设定，确定避障功能后再用随机生成他的 xyz 座标)，预期成果是至少有两个障碍物(可以先生成一个在 leader 必经的路线(n,n,n),n=3~7(因最短路径必为直线，所以生成在直线路径中一定需要进行避障)，后续避障功能顺利再随机生成障碍物位置即可(main 中定义后要在 MPC3 定义时呼叫障碍物)

MPC3 的目标函数里面写(2) a 可以自行选择 如 0.5-1.5 就可以，在 main 中要做的事情如下

定义 Leader 位置(已经完成)

定义障碍物位置(尚未完成)

Main 中的判断式

判断 leader 和障碍物之间的相对距离式

的情况:leader 跟障碍物的距离大于 dD=ld，數字相同但定義不同，在 main(自行定义数字，如 3 则 main 正常运行

的情况 leader 跟障碍物的距离小于 Dd，则要启动 MPC3，公式中的 lm=dM 可定义为 2，l_i^s本身的定义也就照着

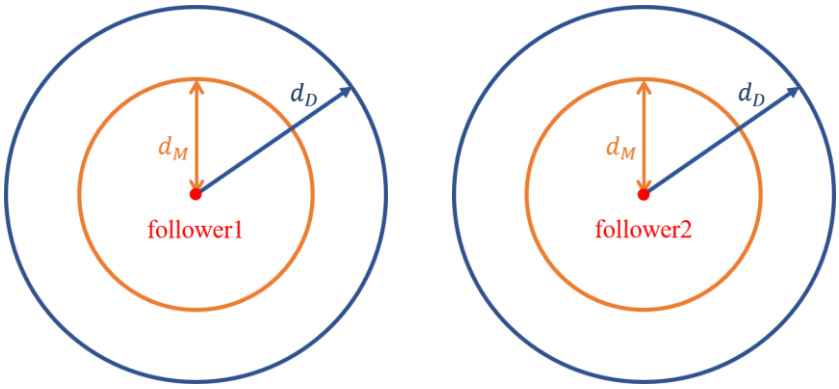
$l_i^S(k) = \sqrt{(P_l^x(k) - x_o)^2 + \left(P_l^y(k) - y_o\right)^2 + (P_l^z(k) - z_o)^2} - R_o$定義寫就好了，Ro 就是障礙物的半徑，上面在定義障礙物就會用到

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 之间的避障

障碍物的安全半径也可以视为无人机(follower)的安全半径(防碰撞距离).



d_M：無人機安全距離

以P_1表示 follower1，中心點座標為(P_1^x, P_1^y, P_1^z)

以P_2表示 follower2，中心點座標為(P_2^x, P_2^y, P_2^z)

如果 follower1 和 follower2 之间的距离小于d_D，则两者间需要避障

$$d_{12}(k) = \sqrt{(P_1^x - P_2^x)^2 + \left(P_1^y - P_2^y\right)^2 + (P_1^z - P_2^z)^2}$$

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

(1) follower1 和 follower2 的距离大于安全半径，不需避障

(2) follower1 和 follower2 的距离小于安全半径，需要避障

功能 follower1 和 follower2 之间的距离最少要保持两个 d_M 的距离，以免碰撞

MPC4 的介紹

这里是为了避免 follower 在避障过程中会有产生碰撞的机会，所以要有 MPC4 (MPC4 在 main 中的权重要比 MPC2 大)(因为我的 MPC2 是以向量跟随 leader，有可能 leader 避障成功但 follower 相撞或有相关问题)，可将 mpc2 中的约束条件更改为软性约束(先完成避障后若有发生运行不了的问题再更动此部分)

MPC4 的功能是防止碰撞

目标函数一样是输入(2)，b 一样为自行定义

其余 dD 和 dM 在 main 中阶已经定义过，引用即可

在 mian 中要定义 d12，后于 MPC4 定义时呼叫即可

$$d_{12}(k) = \sqrt{(P_1^x - P_2^x)^2 + \left(P_1^y - P_2^y\right)^2 + (P_1^z - P_2^z)^2}$$

Main 中若 f1 和 f2 的相对距离小于 2 个 dM 则需要启动 MPC4

MPC4 中的约束条件则参考 MPC2，约束 Follower

以上希望能帮助到您 谢谢