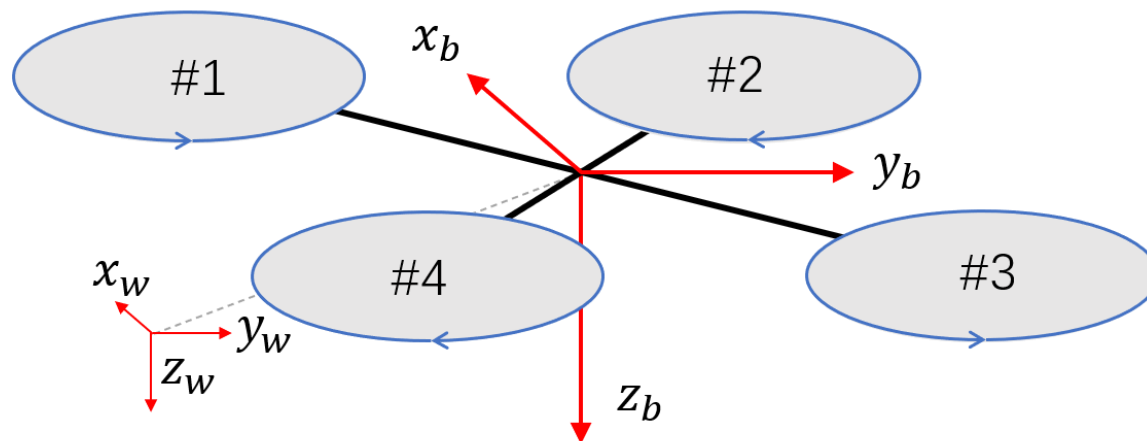


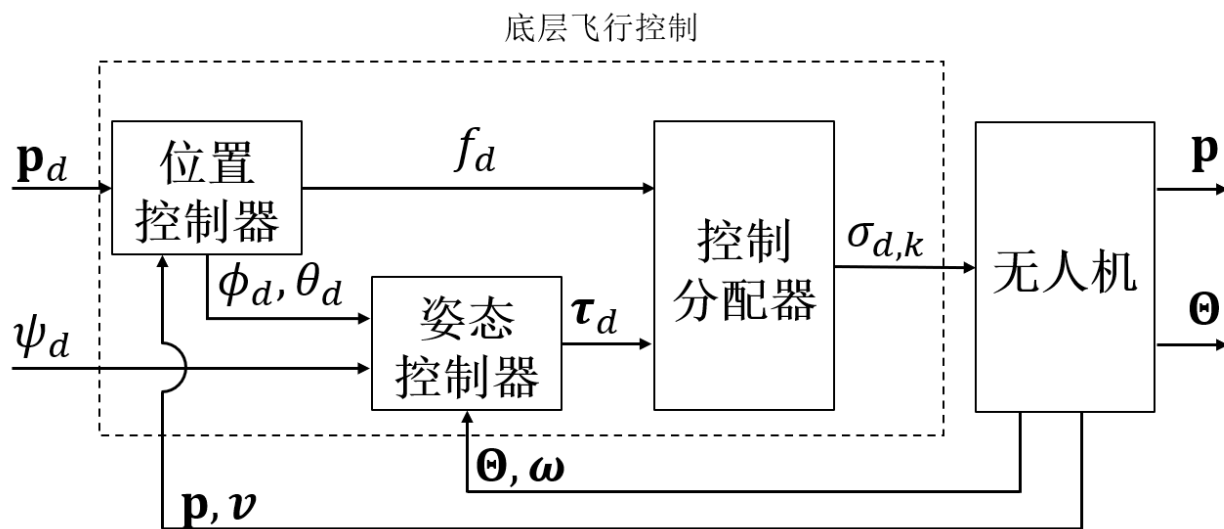
四旋翼无人机的建模与控制

汇报人：孙琦

为实现无人机的位置控制和轨迹跟踪控制，文档对四旋翼无人机的建模与控制进行了较为详细的讲解，在第二节中详细介绍了无人机的动力学模型，在第三节中讲解了四旋翼无人机的拉力模型。针对位置控制和姿态控制，文档在第五节和第六节中分别介绍了基于欧拉角和旋转矩阵的PID位置控制器和PID姿态控制器，并对基于欧拉角的位置和姿态控制器进行了仿真，取得了较为满意的结果。



程序中使用的模型为技术文档中的式（2.10）。



控制流程如上图所示，位置控制器采用式（4.17）和式（4.22），姿态控制器采用式（5.5）。

1. 安装需要的包：numpy 和 matplotlib
2. 设置好模型和控制器的参数
3. 运行main.py
4. 运行结果的曲线图保存在images文件夹中

安装numpy 和 matplotlib 的指令：

```
numpy: pip install numpy -i https://pypi.tuna.tsinghua.edu.cn/simple  
matplotlib: pip install matplotlib -i https://pypi.tuna.tsinghua.edu.cn/simple
```

四旋翼模型参数:



quadrotors_parameters.py

```
m = 2.2 # kg
g = 9.8
dt = 0.01
time = 50 # s
dis = 0.225 # Distance from the center of the motor to the center of the quadrotors. (m)
# Moment of inertia
J_xx = 2.214e-2
J_yy = 2.214e-2
J_zz = 4.203e-2
# Moment of inertia of motor and propellers
J_motor = 1.15e-5
# Fuselage drag coefficient
C_drag = 6.579e-2
# Comprehensive thrust coefficient of single propeller
C_thrust = 1.472e-6
# Comprehensive torque coefficient of single propeller
C_moment = 1.421e-8
```

模型中需要自定义的接口：

扰动：


```
def disturbance(self):  
    """  
    Customize the disturbance.  
    Returns triaxial torque and force.  
    """  
    [torque_x, torque_y, torque_z, f_x, f_y, f_z] = [0, 0, 0, 0, 0, 0]  
    return [torque_x, torque_y, torque_z, f_x, f_y, f_z]
```

噪声：

```
def noise(self):  
    """  
    Customize noise generated during sensor measurement.  
    """  
    return np.array([[0, 0, 0],  
                     [0, 0, 0],  
                     [0, 0, 0],  
                     [0, 0, 0]]) # [[position],[velocity],[attitude],[Attitude velocity]]
```

控制器参数：

位于euler_controller文件的ControllerParameters中。



```
self.tracking = 0
self.position_des = np.array([1, 1, -10]) # [x,y,z]
self.psi_des = 0
```

Tracking = 1, 位置跟踪仿真位置期望由轨迹给出

```
def trajectory(self, t):
    """
    Customize the track to be traced and get the desired position.
    :param t: Current time.
    :return: The desired position.
    """
    x_des_now = np.sin(t * np.pi / 10) + 2
    y_des_now = np.cos(t * np.pi / 10) + 2
    z_des_now = -10
    position_des_now = np.array([x_des_now, y_des_now, z_des_now])
    return position_des_now
```


位置控制器的PID参数:

```
# PID parameters for position controller
self.K_ph_x = 1
self.K_vh_x_p = 0.5
self.K_vh_x_i = 0
self.K_vh_x_d = 0
self.K_ph_y = 1
self.K_vh_y_p = 0.5
self.K_vh_y_i = 0
self.K_vh_y_d = 0
self.K_pz = 1
self.K_vz_p = 1
self.K_vz_i = 0
self.K_vz_d = 1
```

姿态控制器的PID参数:

```
# PID parameters for attitude controller
self.K_phi = 25
self.K_wp_phi = 1.5
self.K_wi_phi = 0
self.K_wd_phi = 0
self.K_theta = 25
self.K_wp_theta = 1.5
self.K_wi_theta = 0
self.K_wd_theta = 0
self.K_psi = 15
self.K_wp_psi = 1.5
self.K_wi_psi = 12
self.K_wd_psi = 0
```



感谢聆听！