Quadcopter Dynamics and Control

Harvard Li

June 23, 2024

Abstract

This essay presents an in-depth analysis of the dynamics of a quadcopter UAV and its ability to achieve swift and stable movement. We begin by establishing a dynamic model that encapsulates the aerodynamic forces and moments acting on the quadcopter during the flight. Subsequently, we explore the design and implementation of the traditional Proportional-Integral-Derivative (PID) controller and its effectiveness in controlling the drone's movements. Furthermore, we delve into a more advanced control, MRAC, elucidating its architecture and its advantage in situations with uncertainties and perturbations.

1 Introduction

An unmanned aerial vehicle (UAV), commonly known as a drone, is an aircraft without any human pilot, crew or passengers on board[7]. A quadcopter UAV distinguishes itself through its four-rotor design, which not only contributes to its manoeuvrability and stability but also greatly simplifies its mechanics compared to the traditional winged design. This design also enables it to perform vertical take-offs and hover in place, which can be beneficial on many occasions.
Quadcopters are renowned for their agility, finding uses in many fields. In cinematography, they offer a wide variety of angles for capturing footage. They can be sent into the disaster to scout or identify individuals in emergencies. In agriculture, UAVs provide a faster way of monitoring and managing crops, optimizing farmers' yield.

Figure 1: The DJI Mavic 3 Classic [3]

1.1 Structure of a Quadcopter

The quintessential part of a quadcopter is its X-shaped frame, which serves as a structural foundation that supports all other components. At the heart of the quadcopter lies the flight controller, which acts as the brain. It is responsible for processing inputs from the pilot (received through the radio receiver) and data from sensors such as GPS. There are also batteries to provide power to the drone. Sometimes, there would be a payload like a camera attached to the centre of the drone.
At the end of each arm is a propeller connected to a motor. An electronic speed controller (ESC) lies on each arm, connecting the motor to the flight controller, which adjusts its speed. This configuration allows the drone to move fast and stably.

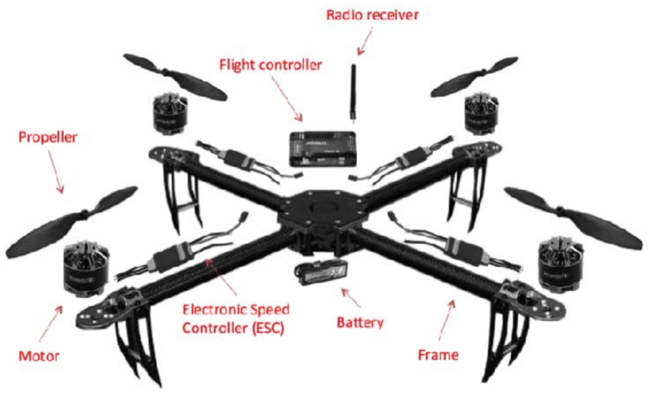


Figure 2: The structure of a quadcopter [4]

2 Quadrotor Dynamics

2.1 Kinematics

2.1.1 Frames of Reference

Before delving into the dynamics of a quadcopter, we must first define the two frames of reference. The first frame is the inertial frame, which is fixed to the earth and gravity is pointing in the negative z direction. The second frame is the body frame, which is attached to the quadcopter at its centre of mass. The lift generated by the propellers will be in the positive z direction and the two pairs of propellers will lie on either the x or the y axis. For simplicity, let propellers 1 and 3 lie on the x axis and propellers 2 and 4 on the y axis. Both the body and the inertial frame are in cartesian coordinates.

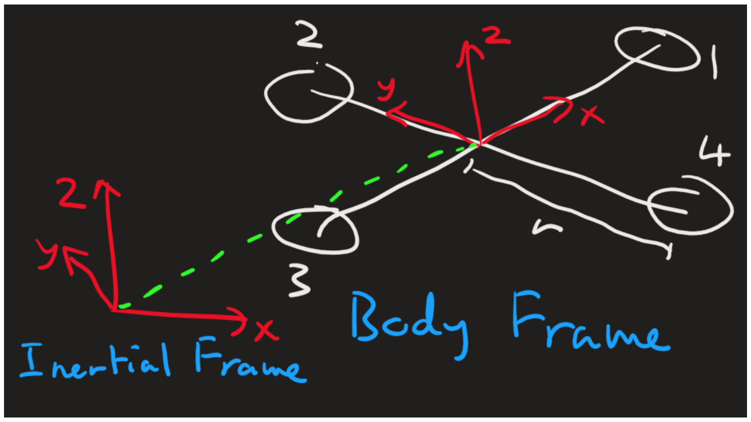


Figure 3: Frames of reference and parameters of quadcopter

2.1.2 Translational and Rotational Axes

Let the position of the quadcopter in the inertial frame be x, y, z. Let roll, pitch and yaw angles be ϕ, θ, ψ.[1] With this in mind, the 3D rotation matrix $\boldsymbol{R_3}$ can be expressed as

$$\boldsymbol{R_3} = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix}$$

[1]roll is rotation about the x axis, pitch is about the y axis and yaw is about the z axis

2.2 Dynamics

For convenience, let us define the angular velocity of the motors to be $\boldsymbol{\omega_1}$, $\boldsymbol{\omega_2}$, $\boldsymbol{\omega_3}$ and $\boldsymbol{\omega_4}$ and their torques to be $\boldsymbol{\tau_1}, \boldsymbol{\tau_2}, \boldsymbol{\tau_3}$ and $\boldsymbol{\tau_4}$. Let the lift induced by the propellers be $\boldsymbol{F_1}$, $\boldsymbol{F_2}$, $\boldsymbol{F_3}$ and $\boldsymbol{F_4}$.

2.2.1 The Translational Subsystem

The net force acting on the system in the inertial frame (excluding outside factors such as wind) is

$$\boldsymbol{F} = \boldsymbol{R_3} \sum \boldsymbol{F_i} - m\boldsymbol{g}$$

By applying Newton's Second Law ($\boldsymbol{F} = m\boldsymbol{a}$), the equation can be rearranged to

$$\begin{bmatrix} m\ddot{x} \\ m\ddot{y} \\ m\ddot{z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + \boldsymbol{R_3} \begin{bmatrix} 0 \\ 0 \\ F_1 + F_2 + F_3 + F_4 \end{bmatrix}$$

This equation can be split into three second order differential equations, which can be used to solve for the position function $\boldsymbol{x}(t)$

$$\ddot{x} = \frac{(F_1 + F_2 + F_3 + F_4)(\cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi)}{m}$$

$$\ddot{y} = \frac{(F_1 + F_2 + F_3 + F_4)(\cos\phi \sin\theta \sin\psi - \sin\phi \cos\psi)}{m}$$

$$\ddot{z} = \frac{(F_1 + F_2 + F_3 + F_4)(\cos\phi \cos\theta)}{m} - g$$

2.2.2 The Rotational Subsystem

The net torque acting on the system is

$$\boldsymbol{\tau} = \boldsymbol{r} \times \sum \boldsymbol{F_i} + \sum \boldsymbol{\tau_i}$$

Due to the design of the drone, $\boldsymbol{r}$ is always perpendicular to $\boldsymbol{F_i}$. In the body frame, $\boldsymbol{F_1}$ and $\boldsymbol{F_3}$ are parallel to the x axis, so its torque should be parallel to the y axis and opposite in direction. Similarly, $\boldsymbol{F_2}$ and $\boldsymbol{F_4}$ are parallel to the y axis, so its torque should be parallel to the x axis, but opposite in direction. Therefore

$$\boldsymbol{\tau} = \boldsymbol{r} \times \sum \boldsymbol{F_i} = \begin{bmatrix} r(F_2 - F_4) \\ r(F_1 - F_3) \\ 0 \end{bmatrix}$$

Propellers 1 and 3 are configured clockwise and propellers 2 and 4 are configured anticlockwise. With this in mind, the net torque can be written as

$$\boldsymbol{\tau} = \begin{bmatrix} r(F_2 - F_4) \\ r(F_1 - F_3) \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \tau_1 - \tau_2 + \tau_3 - \tau_4 \end{bmatrix} = \begin{bmatrix} r(F_2 - F_4) \\ r(F_1 - F_3) \\ \tau_1 - \tau_2 + \tau_3 - \tau_4 \end{bmatrix}$$

Since the geometry of an ideal quadcopter is symmetric about the x, y and z axes, they are the principal axes of the system. The inertia tensor can then be written as a diagonal matrix

$$\boldsymbol{I} = \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix}$$

where I_x, I_y and I_z are the area moment of inertia about the three axes. Euler's equations describes the rotation of a body in the body frame. Its general vector form is

$$\boldsymbol{\tau} = \boldsymbol{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times (\boldsymbol{I}\boldsymbol{\omega})$$

By substituting in the results derived from above, the equality can be rewritten as

$$\begin{bmatrix} r(F_2 - F_4) \\ r(F_1 - F_3) \\ \tau_1 - \tau_2 + \tau_3 - \tau_4 \end{bmatrix} = \begin{bmatrix} I_x\ddot{\phi} + (I_z - I_y)\dot{\theta}\dot{\psi} \\ I_y\ddot{\theta} + (I_x - I_z)\dot{\phi}\dot{\psi} \\ I_z\ddot{\psi} + (I_y - I_x)\dot{\phi}\dot{\theta} \end{bmatrix}$$

Then we can solve for the angular accelerations and represent it as three differential equations

$$\ddot{\phi} = \frac{r(F_2 - F_4) - (I_z - I_y)\dot{\theta}\dot{\psi}}{I_x}$$

$$\ddot{\theta} = \frac{r(F_1 - F_3) - (I_x - I_z)\dot{\phi}\dot{\psi}}{I_y}$$

$$\ddot{\psi} = \frac{\tau_1 - \tau_2 + \tau_3 - \tau_4 - (I_y - I_x)\dot{\phi}\dot{\theta}}{I_z}$$

2.3 User Input

From fluid dynamics, we recall that the lift and torque generated by the propeller is proportional to the square of the angular velocity. If we define k_F and k_τ as the coefficients of ω_i^2, the equations can be written as

$$F_i = k_F \omega_i^2$$

$$\tau_i = k_\tau \omega_i^2$$

Let us define σ as the ratio between the two coefficients, k_F and k_τ, so that it satisfies the equality $\sigma = \frac{k_\tau}{k_F}$. If we then write the input vectors as a linear combination of the forces

$$\boldsymbol{U} = \begin{bmatrix} U_{tx} \\ U_{ty} \\ U_{tz} \\ U_r \end{bmatrix} = \begin{bmatrix} F_2 - F_4 \\ F_1 - F_3 \\ F_1 + F_2 + F_3 + F_4 \\ F_1 - F_2 + F_3 - F_4 \end{bmatrix}$$

This simplifies the six accelerations into the following equations

$$\ddot{x} = \frac{U_{tz}}{m}(\cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi)$$

$$\ddot{y} = \frac{U_{tz}}{m}(\cos\phi \sin\theta \sin\psi - \sin\phi \cos\psi)$$

$$\ddot{z} = \frac{U_{tz}}{m}(\cos\phi \cos\theta) - g$$

$$\ddot{\phi} = \frac{rU_{tx} - (I_z - I_y)\dot{\theta}\dot{\psi}}{I_x}$$

$$\ddot{\theta} = \frac{rU_{ty} - (I_x - I_z)\dot{\phi}\dot{\psi}}{I_y}$$

$$\ddot{\psi} = \frac{\sigma U_r - (I_y - I_x)\dot{\phi}\dot{\theta}}{I_z}$$

In the simulation, the values shown below will be used as the parameters of the quadcopter

$$m = 0.80\text{kg} \qquad r = 0.25\text{m} \qquad g = 9.81\text{N}\cdot\text{kg}^{-1} \qquad \sigma = 1.00$$

$$I_x = 0.015\text{kg}\cdot\text{m}^2 \qquad I_y = 0.015\text{kg}\cdot\text{m}^2 \qquad I_Z = 0.010\text{kg}\cdot\text{m}^2$$

3 The PID Controller

Having built a working dynamic model of the quadcopter UAV, the next step would be to add a control system. Suppose a drone wants to levitate 100m above the ground. At first, this seems quite easy to accomplish, as the drone can switch to hover right before reaching 100m. However, this can be quite inaccurate as inertia will make the drone overshoot the target. In addition, outside factors such as wind, battery power and the condition of the propellers.

To solve this problem, the PID controller will be used. For accurate manoeuvring of the drone, 4 controllers, which corresponds to the 4 inputs, are required. To simplify the calculations, the model will be linearised by applying the small angle approximation. This reduces the model to

$$\ddot{x} = \frac{U_{tz}\theta}{m}$$

$$\ddot{y} = \frac{-U_{tz}\phi}{m}$$

$$\ddot{z} = \frac{U_{tz}}{m} - g$$

$$\ddot{\phi} = \frac{rU_{tx}}{I_x}$$

$$\ddot{\theta} = \frac{rU_{ty}}{I_y}$$

$$\ddot{\psi} = \frac{\sigma U_r}{I_z}$$

With the new linearized system, we can now derive the formula for the PID controller.

The PID controller, as its name suggests, is comprised of 3 controllers: Proportional, Integral and Derivative.

The proportional controller is the main control. As the drone gets closer to the target, it decreases its speed to make sure it does not overshoot. This is achieved by multiplying the error by the proportional gain K_p. The formula for the proportional controller is

$$P = K_p e(t)$$

where $e(t)$ is the error function at time t.
The integral controller provides a correction to the proportional controller by analysing the history of the errors. This is done by integrating over the error function and multiplying the result by the integral constant K_i. The integral controller eliminates steady-state error and controls the drone when it is very close to its target. The formula for the integral controller is

$$I = K_i \int_0^t e(\tau) d\tau$$

The derivative controller provides a prediction to what the error would be like in the future. This is achieved by taking the derivative of the error function at the current time and multiplying the result by the derivative constant K_d. The formula for the derivative controller is

$$D = K_d \frac{de(t)}{dt}$$

where $\frac{de(t)}{dt}$ denotes the time derivative of $e(t)$ at time t.
To find the new input $u(t)$, we just have to add the 3 controllers up

$$u(t) = P + I + D = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt}$$

This might look very complicated and computationally expensive, but we can simplify this algorithm by setting the time interval between each iteration to 1. We should also keeping track of the sum of errors and the previous error, which are crucial for the integral and derivative terms. With these modifications, the algorithm can be simplified to

$$u(t+1) = K_p e(t) + K_i \sum_k e(k) + K_d (e(t) - e(t-1))$$

This algorithm can then be put into a loop which stops when the error becomes sufficiently small.

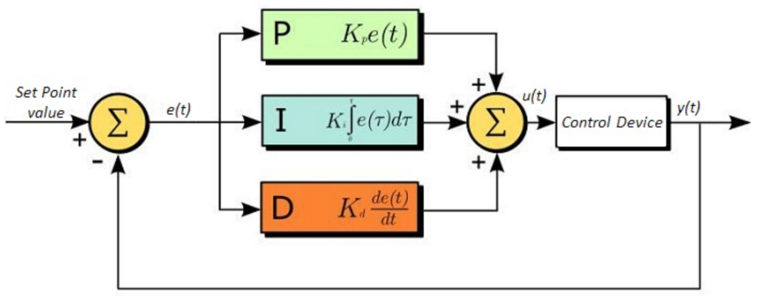


Figure 4: Block diagram of the PID controller [2]

3.1 Translational Control Along the z Axis

The translation along the z axis is relatively simple, as it is independent from the three rotations in the linearized model. This means that only input needed is the U_{tz} input.

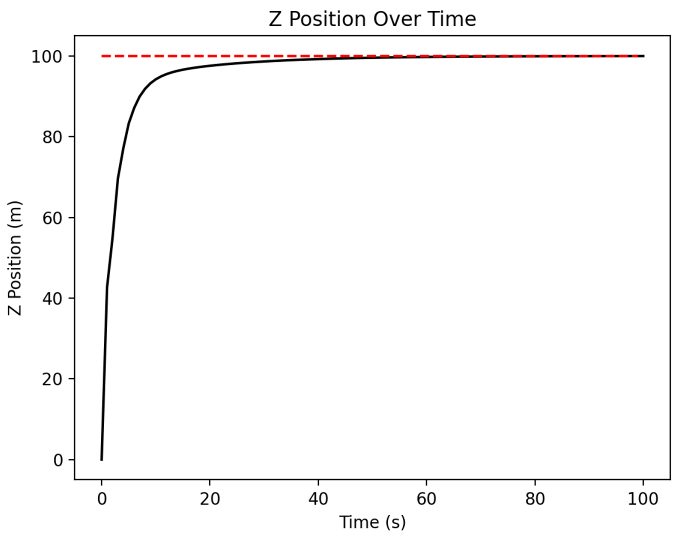


Figure 5: z position of the quadcopter against time

The values used for the constants K_p, K_i and K_d are

$$K_p = 0.4 \qquad K_i = 0.02 \qquad K_d = 1$$

3.2 Translational Control Along the x Axis

The translation along the x axis is a bit more complicated, as it also includes the pitch angle, θ. To simplify the model further, we will be assuming that $U_{tz} = mg$ so that we only have to tune U_{ty} in $\ddot{\theta}$. This simplifies $\ddot{x}$ to

$$\ddot{x} = \theta g$$

Since θ is very sensitive to change, a cascaded (nested) PID controller is needed. The inner controller will be receiving its target (value of θ) from the outer PID. This also means that it will take significantly more time to reach its target destination, as it is a trade-off between accuracy and time.

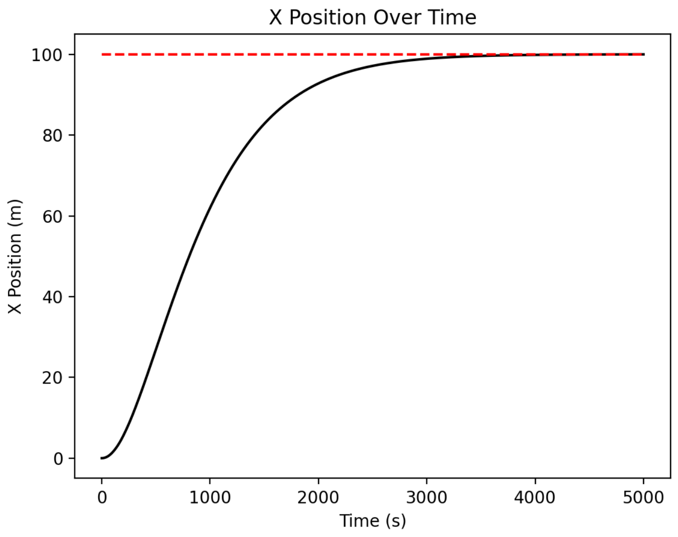


Figure 6: x position of the quadcopter against time

The values used for the inner loop are

$$K_p = 0.01 \qquad K_i = 0.0001 \qquad K_d = 0.01$$

and for the outer loop,

$$K_p = 3 \cdot 10^{-7} \qquad K_i = 10^{-11} \qquad K_d = 3 \cdot 10^{-6}$$

3.3 Translational Control Along the y Axis

The translation along the y axis is the same concept as the x axis. By setting U_{tz} to mg, we can tune U_{tx} in $\ddot{\phi}$ to achieve the target destination. The new equation for $\ddot{y}$ will be

$$\ddot{y} = -\phi g$$

To compensate for the minus sign in the model, the controller gains of the

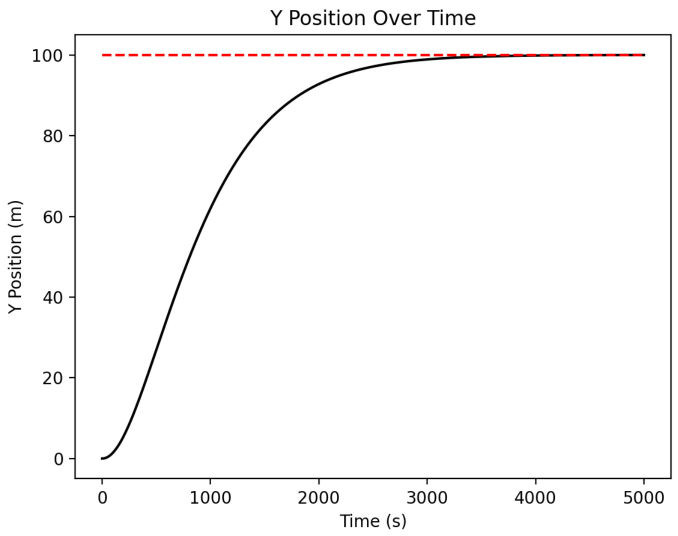


Figure 7: y position of the quadcopter against time

outer loop has to be negative so that the drone will not tilt in the wrong direction. Due to the symmetry of the drone, the values are exactly the same as before, just with a change of signs:

$$K_p = 0.01 \qquad K_i = 0.0001 \qquad K_d = 0.01$$

for the inner loop and

$$K_p = -3 \cdot 10^{-7} \qquad K_i = -10^{-11} \qquad K_d = -3 \cdot 10^{-6}$$

for the outer loop.

3.4 Disturbance

The fact that PID works very well in ideal environments is already clearly demonstrated, but it is another story with the presence of disturbance. We will be using a vector field to describe a gush of wind, which will shift the acceleration of the three positional axes. The wind will be defined as

$$\boldsymbol{W}(x, y, z) = \sin(x + y)\hat{\boldsymbol{x}} + \cos(x - y)\hat{\boldsymbol{y}} + \sin(z^2)\hat{\boldsymbol{z}}$$

The results of the z axis PID controller with the additional wind is

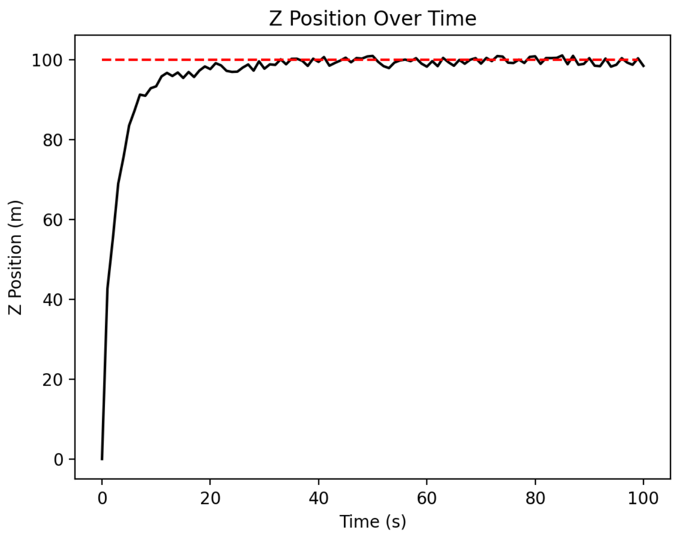


Figure 8: z position of the quadcopter against time in the presence of wind

4 Model Reference Adaptive Control

The PID Controller works very well in stable environments, but once external disturbances are introduced, we need to find a new algorithm to cope with the changes. Since most of the disturbances are unknown, we would have to adapt to the environment based on the movements of the drone - an adaptive controller. We will be using Model Reference Adaptive Control, or MRAC, to control the quadcopter. This controller will try to adapt based on the information of the drone and try to follow a reference model.

4.1 Nominal and Reference Model

To start off, a model of the system dynamics (the nominal model) will be modelled using the following differential equation[2] [3]

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u}$$

where A and B are coefficient matrices, and u is the input vector. We have to track the velocity since our equations of motion are defined by acceleration, which is not present in the model. In addition, tracking velocity can also help reduce oscillations when close to the target.
The input vector $\boldsymbol{u}(t)$ is defined as

$$\boldsymbol{u} = k_r\boldsymbol{r} - k_x\boldsymbol{x}$$

where $\boldsymbol{r}(t)$ is the reference signal (target), and k_x and k_r are the controller gains. By substituting this into the nominal model, we get

$$\dot{\boldsymbol{x}} = (\boldsymbol{A} - k_x\boldsymbol{B})\boldsymbol{x} + k_r\boldsymbol{B}\boldsymbol{r}$$

This can be used to create a closed loop system, where $\boldsymbol{r}(t)$ is the reference signal, k_x and k_r are the controller gains, and $\boldsymbol{x}(t)$ are the measured values. We could use $\boldsymbol{r}(t)$ as the target, but by creating an open loop simulation to describe how an ideal quadcopter would react to the system (the reference model), we can give the drone a more specific target to follow, which will be more accurate. The reference model is defined in a similar way

$$\dot{\boldsymbol{x}}_{\boldsymbol{m}} = \boldsymbol{A}_{\boldsymbol{m}}\boldsymbol{x}_{\boldsymbol{m}} + \boldsymbol{B}_{\boldsymbol{m}}\boldsymbol{r}$$

The ideal situation would be when $\boldsymbol{x} = \boldsymbol{x}_{\boldsymbol{m}}$, as this would imply that the actual drone follows the exact path taken by the ideal drone. The matching equations can be easily derived by setting the 2 velocities equal and the following relations can be found

$$\boldsymbol{A}_{\boldsymbol{m}} = \boldsymbol{A} - k_x\boldsymbol{B}$$

$$\boldsymbol{B}_{\boldsymbol{m}} = k_r\boldsymbol{B}$$

[2] If MRAC is used in an actual experiment, not a simulation, the nominal model will be replaced with the measured data

[3] $\dot{\boldsymbol{x}}$ denotes the time derivative of each term unless stated otherwise

For the system to be stable, we want the error $\boldsymbol{E}(t) = \boldsymbol{x} - \boldsymbol{x_m}$ to tend to $\boldsymbol{0}$. By taking the derivative of $\boldsymbol{E}(t)$

$$\dot{\boldsymbol{E}} = \dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}_{\boldsymbol{m}}$$

and substituting in the nominal and reference model,

$$\dot{\boldsymbol{E}} = (\boldsymbol{A} - k_x\boldsymbol{B})\boldsymbol{x} + k_r\boldsymbol{B}\boldsymbol{r} - \boldsymbol{A_m}\boldsymbol{x_m} - \boldsymbol{B_m}\boldsymbol{r}$$

and taking the matching equations into consideration, we can derive

$$\dot{\boldsymbol{E}} = \boldsymbol{A_m}\boldsymbol{x} - \boldsymbol{A_m}\boldsymbol{x_m} = \boldsymbol{A_m}\boldsymbol{E}$$

For the system to be stable, $\boldsymbol{A_m}$ must be a Hurwitz matrix (meaning all its eigenvalues must have strictly negative real parts).[4]
To update k_x and k_r, a PD controller will be used[5], where the error will be $E = \|\boldsymbol{x} - \boldsymbol{x_m}\|$.[6] In addition, instead of multiplying $\boldsymbol{r}$ and $\boldsymbol{x}$ by a coefficient each, we define a vector $\boldsymbol{k}$ to replace $k_r\boldsymbol{r} - k_x\boldsymbol{x}$, which simplifies the process of tuning.
Looking back at the input vector $\boldsymbol{u}$, it doesn't seem quite right that the velocity and position of the quadcopter can be directly modified by it instead of via the propellers. Therefore, we will be introducing another matrix to convert it into the 4-component input vector $\boldsymbol{U}$. U_{tx}, U_{ty} and U_r are relatively straightforward as they only depend on 1 acceleration each, but U_{tz} needs some extra thought. Since U_{tz} will have to be set constant when dealing with x and y, we will just derive it from $\ddot{z}$ (The coefficient matrix $\boldsymbol{U_m}$ has been expanded out for your sanity):

$$\boldsymbol{U} = \begin{bmatrix} U_{tx} \\ U_{ty} \\ U_{tz} \\ U_r \end{bmatrix} = \boldsymbol{U_m}\dot{\boldsymbol{u}} = \begin{bmatrix} I_x\ddot{\phi}/r \\ I_y\ddot{\theta}/r \\ m\ddot{z} \\ I_z\ddot{\psi}/\sigma \end{bmatrix}$$

[4]The solution to the equation $\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x}$ takes the form of $\boldsymbol{x}(t) = \sum_i c_i e^{\lambda_i t}\boldsymbol{v_i}$, where $\boldsymbol{v_i}$ are the eigenvectors and λ_i are the eigenvalues of $\boldsymbol{A_m}$. For $\boldsymbol{x}(t)$ to converge, λ_i must be strictly negative otherwise $e^{\lambda_i t}$ will grow exponentially.

[5]The Integral term was excluded because it would provide an additional force even when the drone has reached its destination, which could heavily affect the MIT Rule (Section 4.2)

[6]$\|\cdot\|$ denotes the Euclidean norm

4.2 Disturbance Model

We have been assuming that there are no unknown forces acting on the drone. By introducing a disturbance function $f(x)$, our nominal model becomes

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}(\boldsymbol{u} + \boldsymbol{f}(\boldsymbol{x}))$$

Since $\boldsymbol{f}(\boldsymbol{x})$ is unknown, we will define another term $\boldsymbol{u_d} = \boldsymbol{w}^\top \boldsymbol{\varphi}(\boldsymbol{x})$ (the uncertainty model) that will be subtracted from $\boldsymbol{u}(t)$ to cancel out $\boldsymbol{f}(\boldsymbol{x})$.
$\boldsymbol{\varphi}(\boldsymbol{x})$ is a set of basis functions with elements of w being the corresponding weights. The main disturbance that occurs on a quadcopter is wind, which is known for its randomness and unpredictability. Therefore, we will be using Gaussian Radial Basis Functions (Gaussian RBF) as elements of $\varphi(x)$, as they are known as "universal approximators" are very useful in estimating random disturbances. Each Gaussian RBF takes the form

$$\varphi_i(\boldsymbol{x}) = e^{-\sigma_i \|\boldsymbol{x} - \boldsymbol{c_i}\|^2}$$

where σ_i is the width of the curve and $\boldsymbol{c_i}$ is the centre. To update w_i, σ_i and $\boldsymbol{c_i}$, we will be using the MIT Rule, which attempts to minimize the cost function

$$J = \frac{1}{2}\boldsymbol{E}^2$$

by expanding $\boldsymbol{E}(t)$ and substituting in $\boldsymbol{u_d}$, we find

$$J = \frac{1}{2}(\boldsymbol{A_m}^{-1}\dot{\boldsymbol{E}})^2 = \frac{1}{2}(\boldsymbol{A_m}^{-1}(\boldsymbol{A} - k_X\boldsymbol{B})\boldsymbol{x} + (k_r\boldsymbol{A_m}^{-1}\boldsymbol{B} - \boldsymbol{B_m})\boldsymbol{r} - \boldsymbol{x_m} - \boldsymbol{B}\boldsymbol{u})^2$$

The minimization will be done via Gradient Descent, which iteratively adjusts parameters to find the local minima of a cost function. This is done by computing the following

$$\boldsymbol{a}(n+1) = \boldsymbol{a}(n) - \boldsymbol{\eta}\nabla F$$

where $\boldsymbol{F}$ is the cost function, $\boldsymbol{a}(n)$ are the parameters on the nth iteration, and $\boldsymbol{\eta}$ is the learn rate matrix. by substituting in J for F and $\boldsymbol{v_i} = \begin{bmatrix} w_i & \sigma_i & \boldsymbol{c_i} \end{bmatrix}^\top$ for $\boldsymbol{a}$, we can compute the difference equations to update the parameters:

$$w_i(n+1) = w_i(n) + \eta_w E B \varphi_i$$
$$\sigma_i(n+1) = \sigma_i(n) - \eta_\sigma E B w_i \|\boldsymbol{x} - \boldsymbol{c_i}\|^2$$
$$c_i(n+1) = c_i(n) + 2\eta_c E B w_i \sigma_i (\boldsymbol{x} - \boldsymbol{c_i})$$

4.3 Structure of an MRAC

The MRAC starts off by receiving the reference signal $\boldsymbol{r}$ and computing the Reference Model $\boldsymbol{x_m}$. It then enters a closed loop system where a PID controller is used to compute k_r and k_x and MIT Rule is used to compute w, σ and $\boldsymbol{c}$. These three values will then be used in the Disturbance Model to compute $\boldsymbol{u_d}$, which is subtracted from the controller input $\boldsymbol{u}$ to cancel out $\boldsymbol{f}(\boldsymbol{x})$.

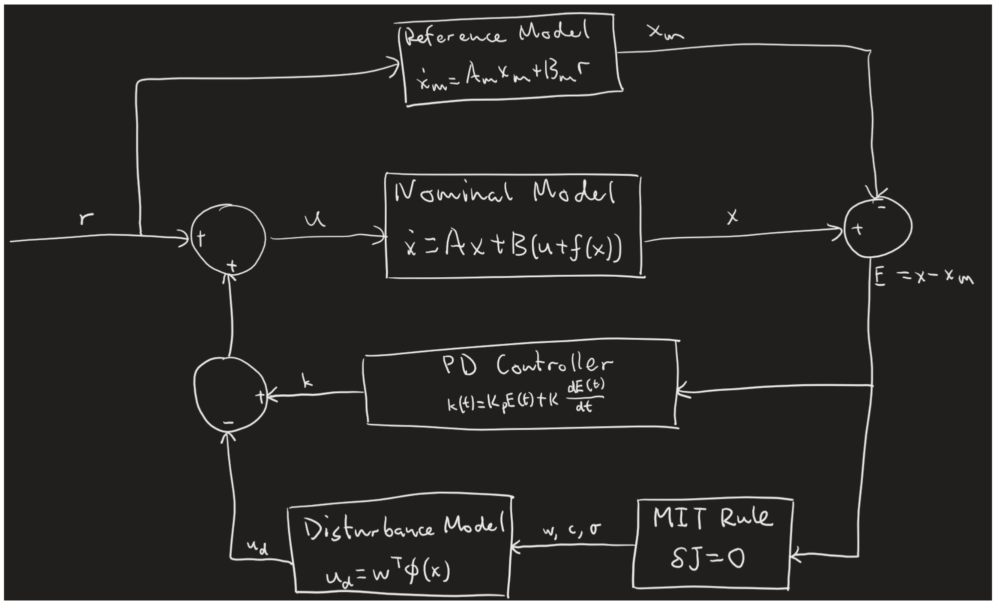


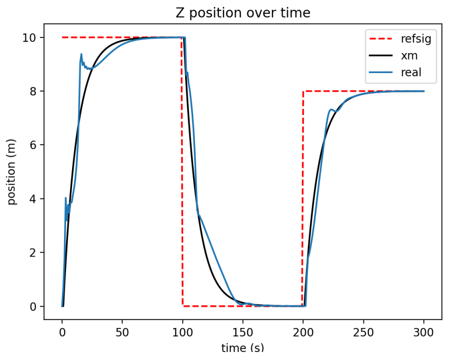
Figure 9: Block diagram of an MRAC

4.4 Translational Control Along the z Axis

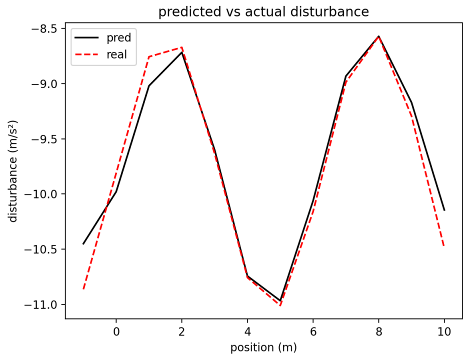
For the control along the z axis, the input vector $\boldsymbol{u}$ becomes the scalar $U_t z$. The nominal model has been replaced with the nonlinear dynamics derived in Section 2. The matrices A_m and B_m take the values

$$A_m = \begin{bmatrix} 0 & 0.3 \\ -0.3 & -0.05 \end{bmatrix} \qquad B_m = \begin{bmatrix} 0 \\ 0.3 \end{bmatrix}$$

The proportional and derivative gains take the values $k_p = 0.4$ and $k_d = 1$ and the learn rates for w, σ and c are (the MIT Rule was updated based on



(a) z position of quadcopter



(b) predicted disturbance

Figure 10: quadcopter in the presence of wind

only the position in the simulation, so there is only one set of learn rates)

$$\eta_w = 0.1 \qquad \eta_\sigma = 10^{-6} \qquad \eta_c = 10^{-7}$$

100 Gaussian RBFs have been used, with initial values of w and σ set to 1. c_i take the integer values from -50 to 49.
The prediction of disturbance by the MIT Rule is quite accurate, with the greatest (relevant) error being around 0.3m. In addition, the quadcopter's path closely resembles the reference model, with just a few jerks and oscillations from time to time.

5 Conclusion

This essay has presented a comprehensive study on the dynamics and control of a quadcopter UAV. We have developed a dynamic model to accurately describe the forces and moments acting on the quadcopter. Through this model, we explored the effectiveness of the Proportional-Integral-Derivative (PID) controller in achieving precise and stable control under ideal conditions.

Furthermore, we introduced Model Reference Adaptive Control (MRAC) as an advanced control strategy to handle uncertainties and external disturbances. Our simulations demonstrated that while the PID controller performs well in stable environments, MRAC provides superior adaptability and

[7]Treat this footnote as an exponent

robustness in the presence of unpredictable disturbances, such as wind.

Our future work could involve implementing these control strategies on actual quadcopter hardware to validate the simulation results. Additionally, exploring other adaptive control techniques (especially nonlinear) and their potential benefits for UAV performance would be a valuable direction to work into.

References

[1] Courage Agho. "Dynamic Model and Control of Quadrotor in the Presence of Uncertainties". In: (2015). URL: `https://www.academia.edu/65303849/Dynamic_Model_and_Control_of_Quadrotor_in_the_Presence_of_Uncertainties` (visited on 02/17/2024).

[2] Alex Cable. *Driver PID Settings*. URL: `https://www.thorlabs.com/newgrouppage9.cfm?objectgroup_id=9013` (visited on 03/13/2024).

[3] *DJI Mavic 3 Classic (DJI RC) – Drone with Camera, 4/3 CMOS Hasselblad Camera, DJI RC Remote Controller, 5.1K HD Video, 46-Min Flight Time, Obstacle Sensing, Drone for Adults, 15km Transmission Range.* Amazon. URL: `%22https://www.amazon.co.uk/dp/B0BGQ8JXDY/ref=as_li_ss_tl?linkCode=gs4&tag=forbes0dc-21` (visited on 02/13/2024).

[4] Valentina Gatteschi et al. "New Frontiers of Delivery Services Using Drones: A Prototype System Exploiting a Quadcopter for Autonomous Drug Shipments". In: *2015 IEEE 39th Annual Computer Software and Applications Conference* 2 (2015), pp. 920–927. URL: `https://api.semanticscholar.org/CorpusID:18731790` (visited on 02/14/2024).

[5] *Model Reference Adaptive Control*. MathWorks. URL: `https://uk.mathworks.com/help/slcontrol/ug/model-reference-adaptive-control.html` (visited on 04/17/2024).

[6] *Proportional-integral-derivative controller*. Wikipedia. URL: `https://en.wikipedia.org/wiki/Proportional%E2%80%93integral%E2%80%93derivative_controller` (visited on 03/12/2024).

[7] *Unmanned aerial vehicle*. Wikipedia. URL: `https://en.wikipedia.org/wiki/Unmanned_aerial_vehicle` (visited on 02/14/2024).