Non-linear Control Approaches for Quadrotor UAVs

Kevin Hu

December 21, 2023

1 Introduction

Unmanned aerial vehicle (UAV) or drone research has been increasing in popularity over the last decade due to its low cost and large range of applications in areas such as aerial photography, package delivery, surveillance, and inspection tasks. Amongst all UAV types including fixed-wing UAVs, multirotor UAVs have especially been popular due to their small size, hovering capabilities, and the fact that they can takeoff and land at a single point rather than requiring a runway to take off. There are several different types of multirotor UAVs with different numbers of propellers, but none quite as popular as the classic quadrotor UAV. The reason for their popularity is due to their comparatively simple dynamics relative to other multirotor UAVs. However, that is not to say that their models are not complex. Quadrotor dynamic models are nonlinear and dynamically coupled. The quadrotor is also underactuated, having only 4 controls, the speed of each of the four propellers, and being required to control movement in 6 degrees of freedom, x, y, and z positions and roll, pitch, and yaw angles. While linear PID control is adequate in some instances as the dynamic model can be linearized, complex, accurate, and fast trajectories cannot be performed well without nonlinear control methods.

Here we explore two non-linear control methods for controlling a quadrotor, integral backstepping and sliding mode control. We will explore how these non-linear control methods relate to Lyapunov stability theory [1].

Figure 1: Quadrotor UAV [2]

2 Nonlinear Control Techniques

Nonlinear control techniques are able to deal with the nonlinear characteristics of nonlinear systems such as underactuations and dynamic coupling [1]. Here we explain two nonlinear control approaches, integral backstepping and sliding mode control, and their connection to classic Lyapunov stability theory.

2.1 Integral Backstepping

Integral backstepping is a recursive controller approach which depends on a proposed Lyapunov function in order to derive the system control law. Unlike some other nonlinear control approaches, it avoids performing non linearity cancellation, allowing for high flexibility in the types of non-linear systems it can be applied to. In order to derive the control laws based on the integral backstepping technique, first determine the error function between the desired input and the system actual output, then propose a Lyapunov function. Then use virtual controls to make the derivative of the Lyapunov function negative definite. Virtual controls are not physical inputs to the system, rather mathematical constructs that are used to simplify the analysis and design process. For example, a virtual control when modeling the acceleration of a system could be the velocity error between the desired velocity and the actual velocity of the system, when actual input and output errors are the positions of the system. Finally these steps are repeated recursively until the control law is obtained [1].

Integral backstepping can be performed on nonlinear systems of the form:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= f(x) + g(x)u \\ y &= x_2 \end{aligned}$$

To perform integral backstepping on the system, we can first define the error function of the system as:

$$e_1 = x_{1d} - x_1$$

and its Lyapunov function as:

$$V_1 = \frac{1}{2}e_1^2$$

We take the derivative of the Lyapunov function resulting in

$$\begin{aligned} \dot{V}_1 &= e_1\dot{e}_1 \\ &= e_1(\dot{x}_{1d} - \dot{x}_1) \end{aligned}$$

We can then add and subtract the gain term $k_1 e_1$ to the $\dot{V}_1$ function where $k1 > 0$ and replace $\dot{x}_1$ with the virtual control variable v_1.

$$
\begin{aligned}
\dot{V}_1 &= e_1(\dot{x}_{1d} - v_1 + k_1 e_1 - k_1 e_1) \\
&= -k_1 e_1^2 + e_1(\dot{x}_{1d} - v_1 + k_1 e_1)
\end{aligned}
$$

In order for the derivative of the Lyapunov function $\dot{V}_1$ to be negative definite in order for stability, we need the term $\dot{x}_{1d} - v_1 + k_1 e_1$ to vanish, which is possible if we choose the virtual control v_1 such that:

$$
v_{1d} = \dot{x}_{1d} + k_1 e_1 + c_1 \int e_1 dt
$$

where $c_1 > 0$ and $c_1 \int e_1 dt$ goes to zero.

Then we define a second error term of the system with virtual control:

$$
e_2 = v_{1d} - v_1
$$

and its Lyapunov function as:

$$
V_2 = \frac{1}{2} e_2^2
$$

Taking the derivative of the Lyapunov function, and applying the same strategy as before we get:

$$
\begin{aligned}
\dot{V}_2 &= e_2 \dot{e}_2 \\
&= e_2(\dot{v}_{1d} - \dot{v}_1) \\
&= e_2(\dot{v}_{1d} - \dot{\dot{x}}_1) \\
&= e_2(\dot{v}_{1d} - \ddot{x}_1) \\
&= e_2(\dot{v}_{1d} - \ddot{x}_1 + k_2 e_2 - k_2 e_2) \\
&= -k_2 e_2^2 + e_2(\dot{v}_{1d} - \ddot{x}_1 + k_2 e_2)
\end{aligned}
$$

where $k1 > 0$. Remembering that the derivative of the Lyapunov function has to be negative definite for stability, we need to ensure that the term $e_2(\dot{v}_{1d} - \ddot{x}_1 + k_2 e_2)$ vanishes. Thus we want,

$$
\begin{aligned}
0 &= e_2(\dot{v}_{1d} - \ddot{x}_1 + k_2 e_2) \\
&= \dot{v}_{1d} - \ddot{x}_1 + k_2 e_2
\end{aligned}
$$

Now substituting in the terms $v_{1d} = \dot{x}_{1d} + k_1 e_1 + c_1 \int e_1 dt$ and the system's non-linear model, since $\dot{x}_1 = x_2$ so $\ddot{x}_1 = \dot{x}_2$ and $\dot{x}_2 = f(x) + g(x)u$, we get:

$$0 = \ddot{x}_{1d} + k_1 \dot{e}_1 + c_1 e_1 - f(x) - g(x)u + k_2 e_2$$

Remember that $c_1 \int e_1 dt$ goes to zero, which means $c_1 e_1$ also goes to zero so $c_1 e_1$ can be removed from the equation. Solving the equation for the input u, we get the result that:

$$u = \frac{1}{g(x)}(\ddot{x}_{1d} + k_1 \dot{e}_1 + k_2 e_2 - f(x))$$

where e_1 and e_2 are two error terms and k_1, k_2, and c_1 are three positive tunable parameters.

2.2 Sliding Mode Control

Sliding mode control is one of the most accurate and robust non-linear control methods against disturbances and model uncertainty. In addition to typical non-linear systems, sliding mode control is a control tool that is also able to work with variable structure systems (VSS), which are systems that consist of continuous subsystems with a proper switching logic, since it produces a discontinuous controller. A sliding mode controller constrains a system to a sliding surface s, which can be separated into two phases, the reaching phase and then a sliding phase. During the reaching phase, trajectories not on the surface $s = 0$, converge towards it and reaches it in a finite amount of time. Then the sliding phase begins and motion is constrained to the surface [1].

The sliding surface may be expressed as a differential equation as a function of an error term e to allow the error to converge to zero, such that $s = c_0 e + c_1 \dot{e} + \ldots + c_{n-1} e^{n-1(.)} + c_n e^{n.}$ where n is the relative degree of the system.

The relative degree can be explained in terms of the system

$$\begin{aligned} \dot{x} &= f(x) + g(x)u \\ y &= h(x) \end{aligned}$$

As a reminder, the Lie derivative for functions a and b is $L_a b(x) = \frac{db(x)}{dx} a(x)$ and the notation $L_a^2 b(x) = L_a L_a b(x) = \frac{d(L_a b(x))}{dx} a(x)$. A system has a relative degree $r \in N$ at a point x_0 if $L_g L_f^k h(x) = 0 \; \forall x$ in a neighbourhood of x_0 and all $k \leq r - 2$ and $L_g L_f^{r-1} h(x_0) \neq 0$. In other words, the relative degree is the number of times the output y has to be differentiated before the input u appears explicitly.

There are several different types of reaching laws that control the derivative of the sliding surface. The constant rate reaching law is $\dot{s} = -Ksgn(s)$ and the constant plus proportional rate reaching law is $\dot{s} = -Qs - Ksgn(s)$, where $Q, K > 0$ and $sgn(s)$ represents the sign function. $sgn(s)$ is equal to -1 if $s < 0$, 0 if $s = 0$, and 1 if $s > 0$.

Using Lyapunov stability analysis, if we take the Lyapunov function candidate $V = \frac{1}{2}s^2$, the condition of stability is that $\dot{V}$ is negative definite. We can show that both the constant rate reaching law and the constant plus proportional rate reaching law allows for stability.

For the constant reaching law:

$$\dot{V} = s\dot{s}$$
$$\dot{V} = -Ks(sgn(s))$$

Notice that if $s < 0$, then $sgn(s) < 0$, causing $\dot{V}$ to be negative. If $s > 0$, then $sgn(s) > 0$, which also causes $\dot{V}$ to be negative. If $s = 0$, then $sgn(s) = 0$, causing $\dot{V} = 0$ which gives the condition that $\dot{V} = 0$ if and only if $s = 0$ for negative definiteness. Thus, the constant reaching law gives stability.

A similar argument can be provided for the constant plus proportional rate reaching law:

$$\dot{V} = s\dot{s}\dot{V} = -Qs^2 - Ks(sgn(s))$$

Notice that $-Qs^2$ is already a negative definite function, and $-Ks(sgn(s))$ is the same as the constant reaching law, which we have already shown is negative definite. Thus, the constant plus proportional rate reaching law also gives stability.

2.3 Feedback Linearization

Feedback linearization is a control strategy used to control non-linear systems. It involves transforming a non-linear system into an equivalent linear control system using a change of variables and change of control input. Feedback linearization is applied to a non-linear systems in the form:

$$\dot{x} = f(x) + g(x)u$$
$$y = h(x)$$

and use a change of coordinates $z = \Phi(x)$ and a change of control input $u = a(x) + b(x)v$ so that the dynamics of $x(t)$ can be reflected in the linear control system as a function of $z(t)$ such that:

$$\dot{z} = Az(t) + bv(t)$$

Using the notion of relative degree introduced in the previous section, assuming the relative degree of the system is n, after performing differentiation on the output n times we have:

$$
\begin{aligned}
y &= h(x) \\
\dot{y} &= L_f h(x) \\
\ddot{y} &= L_f^2 h(x) \\
&\vdots \\
y^{(n-1)} &= L_f^{n-1} h(x) \\
y^{(n)} &= L_f^n + L_g L_f^{n-1} h(x) u
\end{aligned}
$$

due to the fact that $L_g L_f^{n-1} h(x) = 0$ for $i = 1, \ldots, n-2$ and that u has no direct effect on any of the first $(n-1)$ derivatives.

The coordinate transform $T(x)$ that transforms x into z uses the first $(n-1)$ derivatives such that:

$$
z = T(x) = \begin{bmatrix} z_1(x) \\ z_2(x) \\ \vdots \\ z_n(x) \end{bmatrix} = \begin{bmatrix} y \\ \dot{y} \\ \vdots \\ y^{(n-1)} \end{bmatrix} = \begin{bmatrix} h(x) \\ L_f h(x) \\ \vdots \\ L_f^{n-1} h(x) \end{bmatrix}
$$

and

$$
\begin{bmatrix} \dot{z}_1(x) \\ \dot{z}_2(x) \\ \vdots \\ \dot{z}_n(x) \end{bmatrix} = \begin{bmatrix} L_f h(x) \\ L_f^2 h(x) \\ \vdots \\ L_f^n h(x) + L_g L_f^{n-1} h(x) u \end{bmatrix} = \begin{bmatrix} z_2(x) \\ z_3(x) \\ \vdots \\ L_f^n h(x) + L_g L_f^{n-1} h(x) u \end{bmatrix}
$$

.

The feedback control law $u = \frac{1}{L_g L_f^{n-1} h(x)}(-L_f^n h(x) + v)$ gives the linearized system:

$$
\begin{bmatrix} \dot{z}_1(x) \\ \dot{z}_2(x) \\ \vdots \\ \dot{z}_n(x) \end{bmatrix} = \begin{bmatrix} z_2(x) \\ z_3(x) \\ \vdots \\ v \end{bmatrix}
$$

.

Using a state feedback control law of $v = -Kz$, where the state vector z is the output y and its first $(n-1)$ derivatives results in the linear system $\dot{z} = Az$ with

$$
A = \begin{bmatrix} 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & 1 \\ -k_1 & -k_2 & -k_3 & \ldots & -k_n \end{bmatrix}
$$

[3][4]

Now we introduce an equivalent matrix formulation that can be applied systems with multidimensional states. The nonlinear system can be written as:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}$$
$$\mathbf{y} = \mathbf{h}(\mathbf{x})$$

where $\mathbf{x} \in \mathbb{R}^{\mathbf{n}}$, $\mathbf{f}(\mathbf{x}) \in \mathbb{R}^{\mathbf{n}}$, $\mathbf{h}(\mathbf{x}) \in \mathbb{R}^{\mathbf{m}}$ and $\mathbf{g}(\mathbf{x}) \in \mathbb{R}^{\mathbf{n}\times\mathbf{m}}$. Assume that the system can be transformed into a linear and decoupled control system with the application of the diffeomorphism $\mathbf{\Phi}(\mathbf{x}) \in \mathbb{R}^{\mathbf{n}}$.

Let $\xi_{\mathbf{j},\mathbf{k}} = \phi_{\mathbf{j},\mathbf{k}}(\mathbf{x}) = \mathbf{L}_{\mathbf{f}}^{\mathbf{k}-\mathbf{1}}\mathbf{h}_{\mathbf{j}}(\mathbf{x})$ with $k \in (1,\ldots,r_j)$ and $j \in (1,\ldots,m)$ where r_j represents the relative order.

Let the decoupling matrix $\mathbf{\Lambda}(\mathbf{x}) \in \mathbb{R}^{(\mathbf{m}\times\mathbf{m})}$ be defined as:

$$\mathbf{\Lambda}(\mathbf{x}) = \begin{bmatrix} L_{\mathbf{g_1}}L_{\mathbf{f}}^{r_1-1}\mathbf{h_1}(\mathbf{x}) & \ldots & L_{\mathbf{g_m}}L_{\mathbf{f}}^{r_1-1}\mathbf{h_1}(\mathbf{x}) \\ \vdots & \ddots & \vdots \\ L_{\mathbf{g_1}}L_{\mathbf{f}}^{r_m-1}\mathbf{h_m}(\mathbf{x}) & \ldots & L_{\mathbf{g_m}}L_{\mathbf{f}}^{r_m-1}\mathbf{h_m}(\mathbf{x}) \end{bmatrix}$$

where $\mathbf{\Lambda}(\mathbf{x})$ is non-singular and let $\mathbf{b}(\mathbf{x}) \in \mathbb{R}^{\mathbf{m}}$ be defined as:

$$\mathbf{b}(\mathbf{x}) = \begin{bmatrix} L_{\mathbf{f}}^{r_1}\mathbf{h_1}(\mathbf{x}) \\ \vdots \\ L_{\mathbf{f}}^{r_m}\mathbf{h_m}(\mathbf{x}) \end{bmatrix}$$

and let $\mathbf{v} \in \mathbb{R}^{\mathbf{m}}$ denote the transformed input variables vector where:

$$\mathbf{v_j} = \mathbf{L}_{\mathbf{f}}^{\mathbf{r_j}}\mathbf{h_j}(\mathbf{x}) + \sum_{\mathbf{i}=\mathbf{1}}^{\mathbf{m}} \mathbf{L}_{\mathbf{g_i}}\mathbf{L}_{\mathbf{f}}^{\mathbf{r_j}-\mathbf{1}}\mathbf{h_j}(\mathbf{x})\mathbf{u_i} = \dot{\xi}_{\mathbf{j},\mathbf{r_j}}$$

.

The non-linear static state feedback control law is $\mathbf{u} = -\mathbf{\Lambda}^{-\mathbf{1}}(\mathbf{x})\mathbf{b}(\mathbf{x}) + \mathbf{\Lambda}^{-\mathbf{1}}(\mathbf{x})\mathbf{v}$ [5].

3 Quadcopter Modeling and Equations of Motion

The nonlinear model of the UAV is based on the Newton and Euler equations of motion, $\mathbf{F} = \mathbf{ma}$ and $\mathbf{M} = \mathbf{I}\dot{\omega} + \omega \times (\mathbf{I}\omega)$. The model considers two different reference frames, the inertial frame I (O_I, X_I, Y_I, Z_I), and the body fixed frame B (O_B, X_B, Y_B, Z_B) [5].

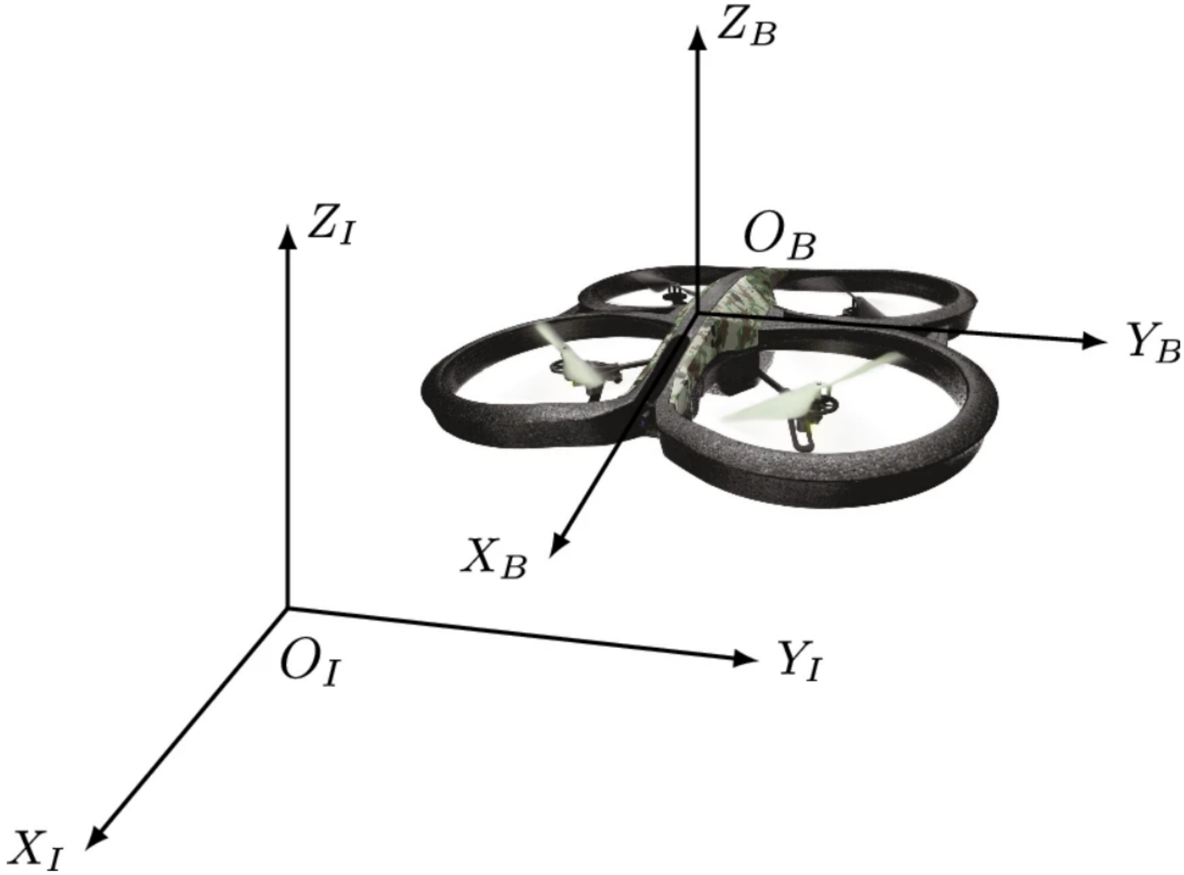

Figure 2: Quadrotor Reference Frames [5]

The position of the quadrotor in the inertial frame can be represented as $\xi = \begin{bmatrix} x & y & z \end{bmatrix}^\top$ and the roll, pitch and yaw Euler angles can be represented as $\gamma = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^\top$.

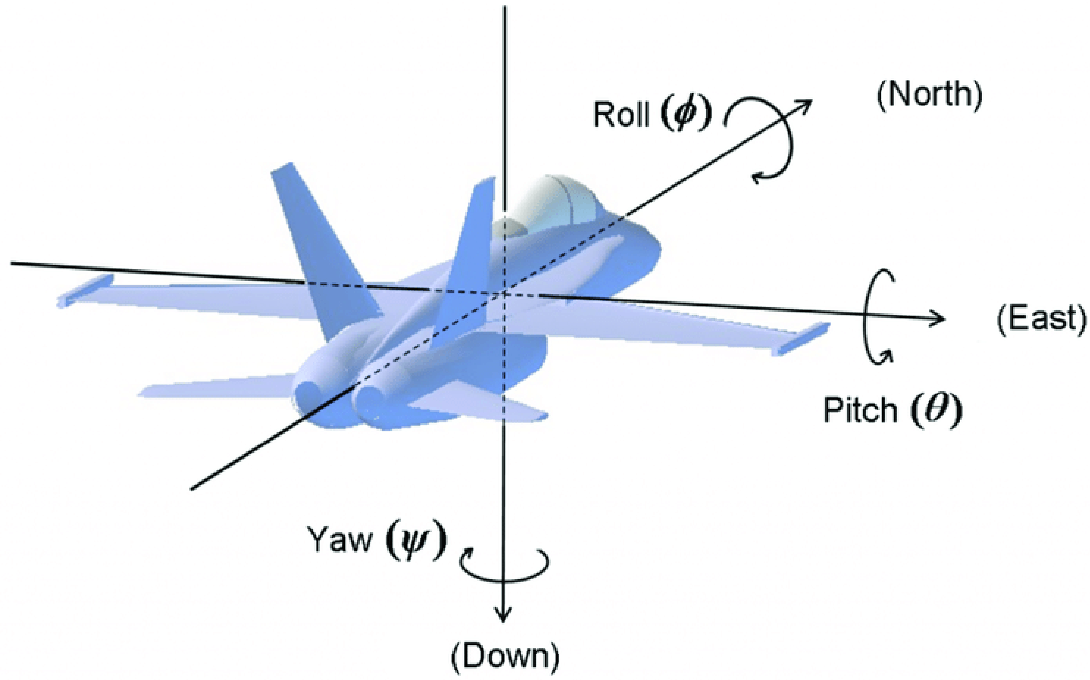

Figure 3: Roll, Pitch, and Yaw Angles and Axes of Rotation [6]

The rotation from the body fixed frame to the inertial frame can be represented by the rotation matrix:

$$\mathbf{R_{IB}} = \begin{bmatrix} c\psi c\theta & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ s\psi c\theta & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi s\theta \end{bmatrix}$$

where c is the shorthand for $cos(\cdot)$, s is the shorthand for $sin(\cdot)$ [1].

The angular velocity of the body $\omega = \begin{bmatrix} p & q & r \end{bmatrix}^\top$ where p, q, r represent the roll, pitch and yaw rates of the body, can be transformed into the rate of change of the Euler angles $\dot{\gamma} = \begin{bmatrix} \dot{\phi} & \dot{\theta} & \dot{\psi} \end{bmatrix}^\top$ using the transformation $\mathbf{T}(\gamma)$ [5] where:

$$\dot{\gamma} = \mathbf{T}(\gamma)\omega = \mathbf{T}(\gamma) \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & sin(\phi)tan(\theta) & cos(\phi)tan(\theta) \\ 0 & cos(\phi) & -sin(\phi) \\ 0 & sin(\phi)sec(\theta) & cos(\phi)sec(\theta) \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

Some sources use $\begin{bmatrix} \dot{\phi} & \dot{\theta} & \dot{\psi} \end{bmatrix}^\top$ in their state vector [1] and some sources use $\begin{bmatrix} p & q & r \end{bmatrix}^\top$ in their state vector [5]. There does not seem to be an obvious benefit of using one representation over the other, but just remember that they are related via a transformation $\mathbf{T}(\gamma)$. We will show the benefits of both representations, by using both representations in different control solutions.

The control inputs to the system are $U = \begin{bmatrix} U_1 & U_2 & U_3 & U_4 \end{bmatrix}^\top = \begin{bmatrix} T & \tau_\phi & \tau_\theta & \tau_\psi \end{bmatrix}^\top$, where T is the total thrust and τ_ϕ, τ_θ and τ_ψ are the moments in the corresponding axes. For each propeller, $f_i = k_t \omega_i^2$ is the thrust force produced by propeller i, and $T_i = k_d \omega_i^2$ is the drag torque produced by propeller i, where k_t and k_d are the thrust and drag coefficients of the propeller and ω_i is the angular speed of motor i. Using that information, we can express U in terms of the forces and torques from the individual propellers using a technique called thrust mixing. So U can be expressed in terms of each of the propeller forces and torques as:

$$U = \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} f_1 + f_2 + f_3 + f_4 \\ l(f_4 - f_2) \\ l(f_3 - f_1) \\ T_1 - T_2 + T_3 - T_4 \end{bmatrix}$$

where l is the length of the moment arm or the distance between the center of the quadrotor and the center of the propeller which should be the same for all four propellers [1].

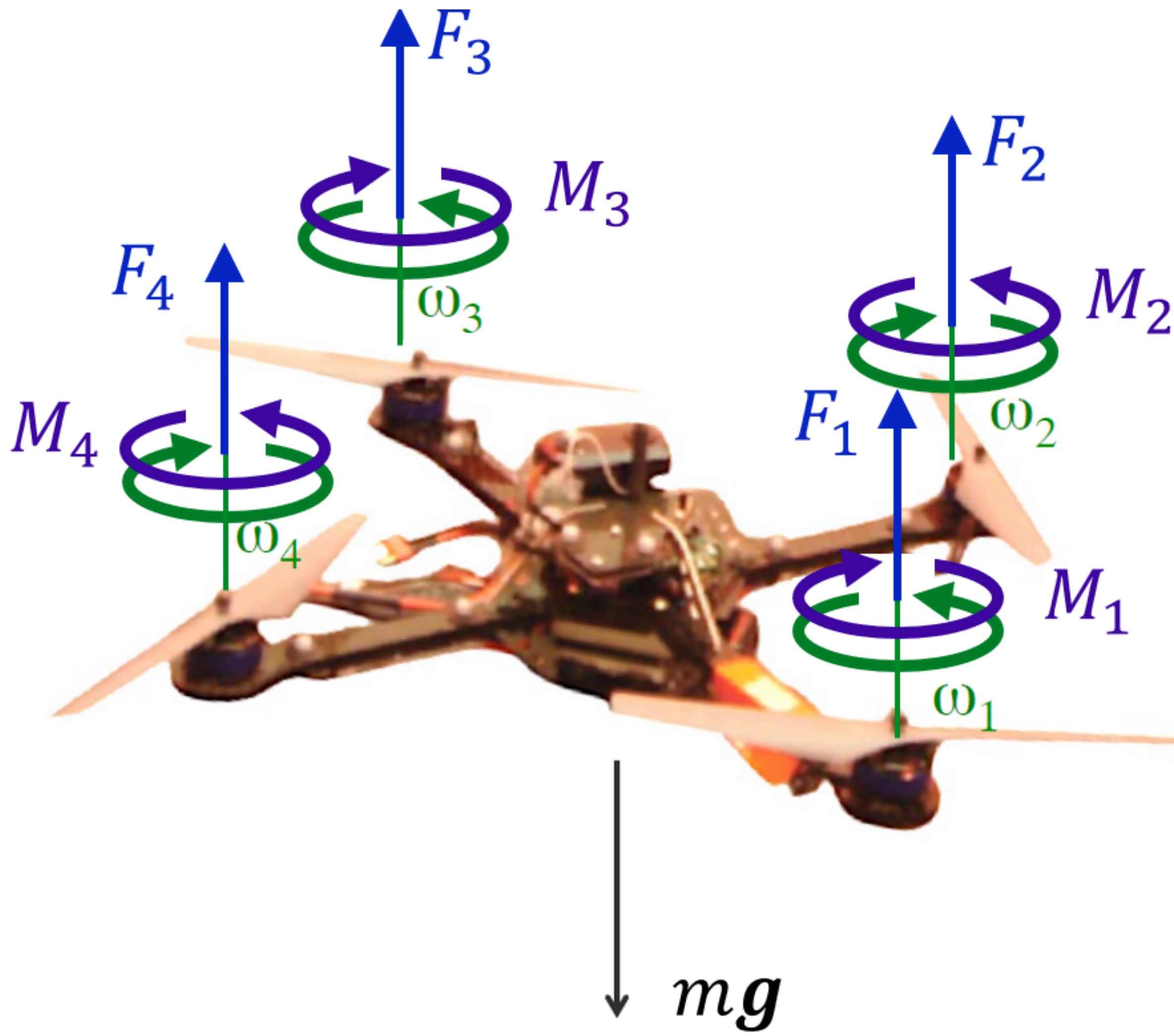

Figure 4: Forces and Torques for Each Propeller on the Quadrotor causing Thrust Mixing [7]

J_r is the moment of inertia of the propeller in the z-axis and $\Omega_r = \omega_1 - \omega_2 + \omega_3 - \omega_4$ which is the sum of the four motors' angular velocity.

U can also be expressed in terms of each of the motors' angular speed as:

$$U = \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} k_t & k_t & k_t & k_t \\ 0 & -k_t l & 0 & k_t l \\ -k_t l & 0 & k_t l & 0 \\ k_d & -k_d & k_d & -k_d \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix}$$

Thus the angular speed of each motor can be determined for a control input U by taking multiplying by the inverse of the matrix and taking the square root [1].

Using all the above information, the equations of motion can be written as follows:

$$
\begin{aligned}
\ddot{x} &= (c\phi s\theta c\psi + s\phi s\psi)\frac{U_1}{m} \\
\ddot{y} &= (c\phi s\theta s\psi - s\phi c\psi)\frac{U_1}{m} \\
\ddot{z} &= (c\phi c\theta)\frac{U_1}{m} - g \\
\ddot{\phi} &= \frac{I_{zz} - I_{yy}}{I_{xx}}\dot{\theta}\dot{\psi} + \frac{J_r}{I_{xx}}\Omega_r\dot{\theta} + \frac{1}{I_{xx}}U_2 \\
\ddot{\theta} &= \frac{I_{zz} - I_{xx}}{I_{yy}}\dot{\phi}\dot{\psi} - \frac{J_r}{I_{yy}}\Omega_r\dot{\phi} + \frac{1}{I_{yy}}U_3 \\
\ddot{\psi} &= \frac{I_{xx} - I_{yy}}{I_{zz}}\dot{\phi}\dot{\theta} + \frac{1}{I_{zz}}U_4
\end{aligned}
$$

Obviously here we use the 12 dimensional state vector:

$$
\begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} & \phi & \theta & \psi & \dot{\phi} \\ \dot{\theta} & \dot{\psi} \end{bmatrix}^\top
$$

If we would rather use $\begin{bmatrix} p & q & r \end{bmatrix}^\top$ in the state vector than $\begin{bmatrix} \dot{\phi} & \dot{\theta} & \dot{\psi} \end{bmatrix}^\top$, we can replace the last three equations [8] with:

$$
\begin{aligned}
\dot{p} &= \frac{I_{zz} - I_{yy}}{I_{xx}}qr + \frac{1}{I_{xx}}U_2 \\
\dot{q} &= \frac{I_{xx} - I_{zz}}{I_{yy}}pr + \frac{1}{I_{yy}}U_3 \\
\dot{r} &= \frac{I_{yy} - I_{xx}}{I_{zz}}pq + \frac{1}{I_{zz}}U_4
\end{aligned}
$$

4 Quadcopter Control

The control scheme of the quadcopter consists of two loops, the inner attitude control loop that gives control commands to the quadcopter, and the outer position control loop that gives control references to the inner loop [1].

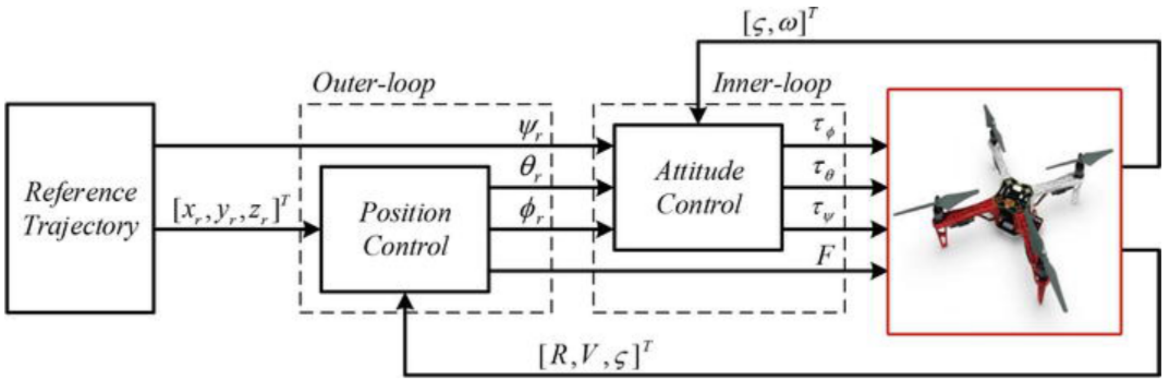

Figure 5: Quadcopter Control Diagram [1]

4.1 Integral Backstepping

Remember that the formula for Integral backstepping is:

$$u = \frac{1}{g(x)}(\ddot{x}_{1d} + k_1\dot{e}_1 + k_2e_2 - f(x))$$

We can find $g(x)$ and $f(x)$ for each equation of motion to solve for each control input [1].

4.1.1 Altitude Control

Here $g(x) = \frac{c\phi c\theta}{m}$ and $f(x) = -g$ so

$$U_1 = \frac{m}{c\phi c\theta}(\ddot{z}_d + k_1\dot{e}_1 + k_2e_2 + g)$$

where $e_1 = z_d - z$, $e_2 = v_{zd} - v_z$, and $v_{zd} = \dot{z}_d + k_1e_1 + c_1\int e_1 dt$.

4.1.2 Attitude Control

Similarly for roll, pitch and yaw control, $g(x) = \frac{1}{I_{xx}}$ and $f(x) = \frac{I_{zz}-I_{yy}}{I_{xx}}\dot{\theta}\dot{\psi} + \frac{J_r}{I_{xx}}\Omega_r\dot{\theta}$ for roll, $g(x) = \frac{1}{I_{yy}}$ and $f(x) = \frac{I_{zz}-I_{xx}}{I_{yy}}\dot{\phi}\dot{\psi} - \frac{J_r}{I_{yy}}\Omega_r\dot{\phi}$ for pitch, and $g(x) = \frac{1}{I_{yy}}$ and $f(x) = \frac{I_{xx}-I_{yy}}{I_{zz}}\dot{\phi}\dot{\theta}$ for yaw.

$$U_2 = I_{xx}(\ddot{\phi}_d + k_3\dot{e}_3 + k_4 e_4 + \frac{I_{yy} - I_{zz}}{I_{xx}}\dot{\theta}\dot{\psi} - \frac{J_r}{I_{xx}}\Omega_r\dot{\theta})$$

where $e_3 = \phi_d - \phi$, $e_4 = \dot{\phi}_d - \dot{\phi}$, and $\dot{\phi}_d = \dot{\phi}_d + k_3 e_3 + c_2 \int e_3 dt$.

$$U_3 = I_{yy}(\ddot{\theta}_d + k_5\dot{e}_5 + k_6 e_6 + \frac{I_{xx} - I_{zz}}{I_{yy}}\dot{\phi}\dot{\psi} + \frac{J_r}{I_{yy}}\Omega_r\dot{\phi})$$

where $e_5 = \theta_d - \theta$, $e_6 = \dot{\theta}_d - \dot{\theta}$, and $\dot{\theta}_d = \dot{\theta}_d + k_5 e_5 + c_3 \int e_5 dt$.

$$U_4 = I_{zz}(\ddot{\psi}_d + k_7\dot{e}_7 + k_8 e_8 + \frac{I_{yy} - I_{xx}}{I_{zz}}\dot{\phi}\dot{\theta})$$

where $e_7 = \psi_d - \psi$, $e_8 = \dot{\psi}_d - \dot{\psi}$, and $\dot{\psi}_d = \dot{\psi}_d + k_7 e_7 + c_4 \int e_7 dt$.

4.1.3 Position Control

The Cartesian motion of a quadcopter depends on the θ and ϕ angles, so here θ and ϕ are used as the outputs of x and y [1]. Isolating for θ and ϕ and adding the appropriate error terms we get:

$$\theta_d = arcsin(\frac{m}{c\phi c\theta U_1}(\dot{v}_{xd} - \frac{s\phi s\psi}{m}U_1 + k_9\dot{e}_9 + k_{10}e_{10}))$$
$$\phi_d = -arcsin(\frac{m}{c\psi U_1}(\dot{v}_{yd} - \frac{c\phi s\theta s\psi}{m}U_1 + k_{11}\dot{e}_{11} + k_{12}e_{12}))$$

,

where $e_9 = x_d - x$, $e_{10} = v_{xd} - v_x$, $v_{xd} = \dot{x_d} + k_9 e_9 + c_5 \int e_9 dt$, $e_{11} = y_d - y$, $e_{12} = v_{yd} - v_y$, $v_{yd} = \dot{y_d} + k_{11}e_{11} + c_6 \int e_{11} dt$. Note that all the constants for all the controllers should be tuned to give the best response. Each state variable controller should have three constants to tune.

4.2 Sliding Mode Control

4.2.1 Altitude Control

We derive the sliding mode control law for altitude as an example. All other sliding mode control laws have a similar structure and is derived in a similar way. We use the sliding surface $s_1 = c_1 e_1 + \dot{e}_1$, where $e_1 = z_d - z$ and $\dot{e}_1 = \dot{z}_d - \dot{z}$. The derivative of the sliding surface is $\dot{s}_1 = c_1\dot{e}_1 + \ddot{e}_1$. $\ddot{e}_1 = \ddot{z}_d - \ddot{z}$ and substituting in the equations of motion, we get $\ddot{e}_1 = \ddot{z}_d - \frac{c\phi c\theta}{m}U_1 + g$. Substituting the expression into the equation for $\dot{s}_1$, we get $\dot{s}_1 = \ddot{z}_d + c_1(\dot{z}_d - \dot{z}) - \frac{c\phi c\theta}{m}U_1 + g$. Substituting in the constant and proportional rate reaching law, we get $-K_1 s_1 - Q_1 sgn(s_1) = \ddot{z}_d + c_1(\dot{z}_d - \dot{z}) - \frac{c\phi c\theta}{m}U_1 + g$. Rearranging for U_1, we get the control law for the altitude which is $U_1 = \frac{m}{c\phi c\theta}(\ddot{z}_d + c_1\dot{e}_1 + K_1 s_1 + Q_1 sgn(s_1) + g)$ [1]. Looking at the structure of equation of U_1 we have a similar structure as for integral backstepping. The structure is $U_1 = \frac{1}{g(x)}(\ddot{x}_d + c_1\dot{e}_1 + K_1 s_1 + Q_1 sgn(s_1) - f(x))$ where this time the error terms are $c_1\dot{e}_1 + K_1 s_1 + Q_1 sgn(s_1)$, which is different from the error terms $k_1\dot{e}_1 + k_2 e_2$ for integral backstepping. Everything else about the structure is the same.

4.2.2 Attitude Control

Using the same $g(x)$ and $h(x)$ as integral backstepping attitude control we have:

$$U_2 = I_{xx}(\ddot{\phi}_d + c_2\dot{e}_2 + K_2 s_2 + Q_2 sgn(s_2) + \frac{I_{yy} - I_{zz}}{I_{xx}}\dot{\theta}\dot{\psi} - \frac{J_r}{I_{xx}}\Omega_r\dot{\theta})$$

where $\dot{e}_2 = \phi_d - \phi$ and $s_2 = c_2 e_2 + \dot{e}_2$.

$$U_3 = I_{yy}(\ddot{\theta}_d + c_3\dot{e}_3 + K_3 s_3 + Q_3 sgn(s_3) + \frac{I_{xx} - I_{zz}}{I_{yy}}\dot{\phi}\dot{\psi} + \frac{J_r}{I_{yy}}\Omega_r\dot{\phi}$$

where $\dot{e}_3 = \theta_d - \theta$ and $s_3 = c_3 e_3 + \dot{e}_3$.

$$U_4 = I_{zz}(\ddot{\psi}_d + c_4\dot{e}_4 + K_4 s_4 + Q_4 sgn(s_4) + \frac{I_{yy} - I_{xx}}{I_{zz}}\dot{\phi}\dot{\theta})$$

where $\dot{e}_4 = \psi_d - \psi$ and $s_4 = c_4 e_4 + \dot{e}_4$.

4.2.3 Position Control

Similar to the integral backstepping position control, we solve for the θ and ϕ reference trajectories. We include the same error terms as for the other sliding mode controls and so we get:

$$\theta_d = arcsin(\frac{m}{c\phi c\theta U_1}(\dot{v}_{xd} - \frac{s\phi s\psi}{m}U_1 + c_5\dot{e}_5 + K_5 s_5 + Q_5 sgn(s_5)))$$
$$\phi_d = -arcsin(\frac{m}{c\psi U_1}(\dot{v}_{yd} - \frac{c\phi s\theta s\psi}{m}U_1 + c_6\dot{e}_6 + K_6 s_6 + Q_6 sgn(s_6)))$$

where $\dot{e}_5 = x_d - x$, $s_5 = c_5 e_5 + \dot{e}_5$, $\dot{e}_6 = y_d - y$, and $s_6 = c_6 e_6 + \dot{e}_6$. Again, all the constants for all the controllers should be tuned to give the best response. Each state variable controller should have three constants to tune.

4.3 Feedback Linearization

4.3.1 Attitude and Altitude Control

We use the variable $\mathbf{x_{in}} = \begin{bmatrix} z & \phi & \theta & \psi & \dot{z} & p & q & r \end{bmatrix}^\top$ to denote the vector of state variables of the inner loop dynamics and $\mathbf{u_{in}} = \begin{bmatrix} U_1 & U_2 & U_3 & U_4 \end{bmatrix}^\top$ to denote the vector of inputs to the system. Using the non linear equations of motion involving p, q and r, having the output $\mathbf{y_{in}} = \begin{bmatrix} z & \phi & \theta & \psi \end{bmatrix}^\top$ and using the relative degree vector $\mathbf{r} = \mathbf{2}, \mathbf{2}, \mathbf{2}, \mathbf{2}$ we have the static state feedback control law $\mathbf{u_{in}} = -\mathbf{\Lambda_{in}^{-1}}(\mathbf{x_{in}})\mathbf{b_{in}}(\mathbf{x_{in}}) + \mathbf{\Lambda_{in}^{-1}}(\mathbf{x_{in}})\mathbf{v_{in}}$.

$$\mathbf{\Lambda_{in}}(\mathbf{x_{in}}) = \begin{bmatrix} c\theta c\phi & 0 & 0 & 0 \\ 0 & \frac{1}{I_{xx}} & \frac{t\theta s\phi}{I_{yy}} & \frac{c\phi t\theta}{I_{zz}} \\ 0 & 0 & \frac{c\theta}{I_{yy}} & -\frac{s\phi}{I_{zz}} \\ 0 & 0 & \frac{s\phi}{I_{yy}c\theta} & \frac{c\phi}{I_{zz}c\theta} \end{bmatrix}$$

with t being the shorthand for $tan(\cdot)$. The vector $\mathbf{b_{in}}(\mathbf{x_{in}})$ holds the terms of the second order time derivative which are independent of the input. The transformed state variables from the diffeomorphism creates the vector of transformed state variables $\xi_{\mathbf{in}} = \mathbf{\Phi}(\mathbf{x_{in}}) = \begin{bmatrix} z & \dot{z} & \phi & \dot{\phi} & \theta & \dot{\theta} & \psi & \dot{\psi} \end{bmatrix}^\top$.

Using the attitude error $\mathbf{e}_\gamma = \gamma - \gamma_\mathbf{d}$ and the position error $\mathbf{e_p} = \mathbf{p} - \mathbf{p_d}$, the transformed input vector $\mathbf{v_{in}}$ can be written as:

$$\mathbf{v_{in}} = \begin{bmatrix} \mathbf{e_3^\top}(-\mathbf{k_{iz}}\dot{\xi}_\mathbf{p} - \mathbf{k_z}\mathbf{e_p} - \mathbf{k_{\dot{z}}}\dot{\mathbf{e}}_\mathbf{p} + \ddot{\mathbf{p}}_\mathbf{d} \\ -\mathbf{K_{i\gamma}}\xi_\gamma - \mathbf{K}_\gamma\mathbf{e}_\gamma - \mathbf{K}_{\dot{\gamma}}\dot{\mathbf{e}}_\gamma + \ddot{\gamma}_\mathbf{d}) \end{bmatrix}$$

where k_{iz}, k_z, $k_{\dot{z}}$, $\mathbf{K_{i\gamma}}$, $\mathbf{K}_\gamma$ and $\mathbf{K}_{\dot{\gamma}}$ are gains resulting from LQR gain computation and ξ_γ and $\xi_\mathbf{p}$ are integral states that solve $\dot{\xi}_\gamma = \mathbf{e}_\gamma$ and $\dot{\xi}_p = \mathbf{e_p}$ [5].

LQR or LQI stands for linear quadratic regulator or linear quadratic integral and they mean similar things. The linear quadratic regulator cost function is given as

$$J = \int_\infty^0 (x^\top Q x + u^\top R u) dt$$

where Q and R represent the weighting matrices for the state vector x and control law vector u. LQR is applied to linear or linearized non-linear control systems in the form of $\dot{x} = Ax + By$, $y = Cx + Du$. The control law u is derived from the minimum of the cost function J and $u = -Kx = -R^{-1}B^\top Px$ where P is a covariance matrix. P is the solution of the algebraic Riccati equation where $A^\top P + PA - PBR^{-1}B^\top P + Q = 0$ [1].

Using the feedback control law and the diffeomorphism, we get the following altitude and attitude tracking dynamics [5]:

$$\begin{bmatrix} \mathbf{e_3^\top}\ddot{\mathbf{e}}_\mathbf{p} \\ \ddot{\mathbf{e}}_\gamma \end{bmatrix} = \begin{bmatrix} \mathbf{e_3^\top}(-\mathbf{k_{iz}}\xi_\mathbf{p} - \mathbf{k_z}\mathbf{e_p} - \mathbf{k_{\dot{z}}}\dot{\mathbf{e}}_\mathbf{p}) \\ -\mathbf{K_{i\gamma}}\xi_\gamma - \mathbf{K}_\gamma\mathbf{e}_\gamma - \mathbf{K}_{\dot{\gamma}}\dot{\mathbf{e}}_\gamma \end{bmatrix}$$

4.3.2 Zero Dynamics Control

Since the inner loop dynamics are stabilized, we solve something called the zero dynamics by setting $\mathbf{y_{in}} = \mathbf{0}$, $\xi_{\mathbf{in}} = \mathbf{0}$ and $\mathbf{v_{in}} = \mathbf{0}$ resulting in the input vector $\mathbf{u_{in}}$ that solves $0 = \mathbf{b_{in}}(\mathbf{x_i n}) + \mathbf{\Lambda_{in}}(\mathbf{x_{in}})\mathbf{u_{in}}$.

We can denote the vector of state variables for the outer dynamics as $\mathbf{x_{out}} = \begin{bmatrix} x & y & \dot{x} & \dot{y} \end{bmatrix}^\top$ and the input of outer loop dynamics as $\mathbf{u_{out}} = \begin{bmatrix} c\phi & s\theta & s\phi \end{bmatrix}$ giving a system in the form:

$$
\begin{aligned}
\dot{\mathbf{x}}_{out} &= \begin{bmatrix} 0 & I_2 \\ 0 & 0 \end{bmatrix} \mathbf{x_{out}} + \mathbf{g} \begin{bmatrix} 0 & 0 \\ c\psi & s\psi \\ s\psi & -c\psi \end{bmatrix} \mathbf{u_{out}} \\
\mathbf{y_{out}} &= \begin{bmatrix} x \\ y \end{bmatrix}
\end{aligned}
$$

Combining the result for the attitde and altitude control with the zero dynamics we can asymptotically stabilize the system. We will not prove the feedback linearization result here, but the proof can be found here [5].

5 References

[1] A. Joukhadar, M. Alchehabi, and A. Jejeh, "Advanced uavs nonlinear control systems and applications," IntechOpen, https://www.intechopen.com/chapters/67705 (accessed Dec. 20, 2023)

[2] A PSO-based tuning algorithm for Quadcopter controllers - researchgate https://www.researchgate.net/publication/329620487 _A_PSO-BASED_TUNING_ALGORITHM_FOR_QUADCOPTER_CONTROLLERS (accessed Dec. 21, 2023).

[3] A. Isidori, Nonlinear Control Systems. London: Springer, 1995.

[4] H. Nijmeijer and S. A. van der, Nonlinear Dynamical Control Systems. New York: Springer Science+Business Media, 2016.

[5] Nonlinear control of a quadrotor micro-UAV using feedback-linearization ..., https://ieeexplore.ieee.org/abstract/document/4957154 (accessed Dec. 21, 2023).

[6] "What is the relation between roll angle and pitch angle?," Aviation Stack Exchange, https://aviation.stackexchange.com/questions/16531/what-is-the-relation-between-roll-angle-and-pitch-angle (accessed Dec. 20, 2023).

[7] Quadrotor Dynamics and Control Slides, Prof. Angela Schoellig.

[8] A. Gibiansky, "Quadcopter Dynamics and Simulation," Quadcopter Dynamics and Simulation - Andrew Gibiansky, https://andrew.gibiansky.com/blog/physics/quadcopter-dynamics/ (accessed Dec. 20, 2023).