Quadcopter Drone: Adaptive Control Laws

Alfredo M. Garó M.
ECE Grad Student
garomonegro@gatech.edu

Tianyang Cao
ECE Grad Student
tcao77@gatech.edu

Al Chandeck
ECE Grad Student
a.chandeck@gatech.edu

***Abstract*—This project implements adaptive control laws in order to control a Parrot Rolling Spider mini drone. We will go through two steps: Step 1 - Linear Adaptive Control, Step 2 - Nonlinear Controller. We simulate the performance of our model reference adaptive controller (MRAC) and our fixed-gain linear controller, and show that the former outperforms the latter in stabilization and tracking tasks.**

***Index Terms*—Model Reference Adaptive Controller, Linearization, Quadcopter, Parrot Rolling Spider, Nonlinear Controller.**

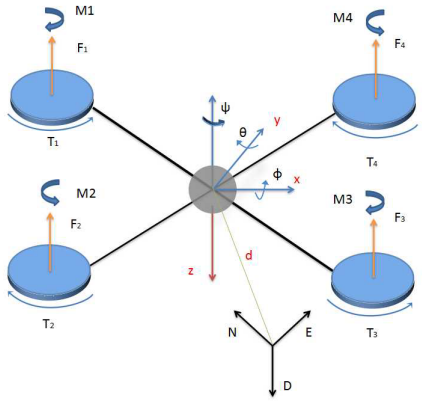
I. Introduction

When one tries to control a physical plant/system, an accurate dynamics model is crucial. Traditional control laws are designed assuming the estimated plant parameters are ideal. This poses a potential problem, since failure to obtain the plant model and parameters correctly would lead to a failure in the control tasks. But as we know, the real world is unpredictable and not only could the system parameters change over time, but also the properties of the environment. Adaptive Control techniques are capable of dealing with such parameter uncertainty problems. By the inverse Lyapunov functions approach, we can design feedback controllers with dynamic gains that can adapt to the uncertainties, as opposed to the fixed gains in the traditional static controllers.

We will go through several steps to design a functional adaptive controller for a quadcopter, specifically the Parrot Rolling Spider, and show that this controller is better able to control the drone for stabilization and tracking tasks than the traditional controllers.

II. Equations of Motion and State-Space

A. Equations of Motion



Before we start diving into the control theory, we need the equations of motions from which we will derive our state-space equations. For any drone we normally need nine equations of motion. These define the dynamics of: the linear acceleration $(\ddot{x}, \ddot{y}, \ddot{z})$ with respect to the world frame, angular velocities $(\dot{p}, \dot{q}, \dot{r})$ with respect to the drone's body frame, and, lastly, the Euler rates $(\dot{\phi}, \dot{\theta}, \dot{\psi})$.

$$\ddot{x} = -\frac{T}{m}[\cos(\phi)\sin(\theta)\cos(\psi) + \sin(\phi)\sin(\psi)]$$

$$\ddot{y} = -\frac{T}{m}[\cos(\phi)\sin(\theta)\sin(\psi) - \sin(\phi)\cos(\psi)]$$

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

$$\dot{p} = \frac{\tau_x + I_y qr - I_z qr}{I_x}$$

$$\dot{q} = \frac{\tau_y - I_x pr + I_z pr}{I_y}$$

$$\dot{r} = \frac{\tau_z + I_x pq - I_y pq}{I_z}$$

Let $\Theta = [\phi, \theta, \psi]^\intercal$ and let $w = [\dot{p}, \dot{q}, \dot{r}]^\intercal$. Using the following transformation we can obtain the Euler Rates from the Body rates.

$$\dot{\Theta} = W^{-1} w$$

$$W^{-1} = \frac{1}{\cos(\theta)} \begin{bmatrix} \cos(\theta) & \sin(\phi)\sin(\theta) & \cos(\phi)\sin(\theta) \\ 0 & \cos(\phi)\cos(\theta) & -\sin(\phi)\cos(\theta) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix}$$

With this transformation we get:

$$\dot{\phi} = p + \frac{r\cos(\phi)\sin(\theta)}{\cos(\theta)} + \frac{q\sin(\theta)\sin(\phi)}{\cos(\theta)}$$

$$\dot{\theta} = q\cos(\phi) - r\sin(\phi)$$

$$\dot{\psi} = \frac{r\cos(\phi)}{\cos(\theta)} + \frac{q\sin(\phi)}{\cos(\theta)}$$

B. State-Space Representation

Now we have to go from the equations of motion to the state-space representation. The latter will define the states of the system, in our case the quadcopter, that are relevant to us. These states are $(x_i,\ i = 1, ..., 12)$, which together make up our state vector X as follows:

$$X = [x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}]^\intercal$$

Components $x_i,\ i = 1, ..., 12$ forming the state vector X correspond to the following (in the same order).

$$X = [x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, p, q, r]^\intercal$$

Then, the dynamics of the state vector X, also known as state dynamic, are given by the vector $\dot{X} = [\dot{x_1}, \dot{x_2}, ..., \dot{x_{12}}]^\intercal$, whose components are defined as follows:

$$[\dot{x_1}, \dot{x_2}, \dot{x_3}]^\intercal = [x_7, x_8, x_9]^\intercal \tag{1}$$

$$\dot{x_4} = x_{10} + \sin(x_4)\tan(x_5)x_{11} + \cos(x_4)\tan x_5 x_{12} \tag{2}$$

$$\dot{x_5} = \cos(x_4)x_{11} - \sin(x_4)x_{12} \tag{3}$$

$$\dot{x_6} = \frac{\sin(x_4)}{\cos(x_5)}x_{11} + \frac{\cos(x_4)}{\cos(x_5)}x_{12} \tag{4}$$

$$\dot{x_7} = -\frac{T}{m}[\cos(x_4)\sin(x_5)\cos(x_6) + \sin(x_4)\sin(x_6)] \tag{5}$$

$$\dot{x_8} = -\frac{T}{m}[\cos(x_4)\sin(x_5)\sin(x_6) - \sin(x_4)\cos(x_6)] \tag{6}$$

$$\dot{x_9} = -\frac{T}{m}\cos(x_4)\cos(x_5) + g \tag{7}$$

$$\dot{x_{10}} = \frac{\tau_x}{I_x} + \frac{I_y - I_z}{I_x}x_{11}x_{12} \tag{8}$$

$$\dot{x_{11}} = \frac{\tau_y}{I_y} + \frac{-I_x + I_z}{I_y}x_{10}x_{12} \tag{9}$$

$$\dot{x_{12}} = \frac{\tau_z}{I_z} + \frac{I_x - I_y}{I_z}x_{10}x_{11} \tag{10}$$

where T, τ_x, τ_y, τ_z are defined as:

$$T = K_a({w_1}^2 + {w_2}^2 + {w_3}^2 + {w_4}^2) \tag{11}$$

$$\tau_x = K_a l({w_4}^2 - {w_2}^2)$$
$$\tau_y = K_a l({w_1}^2 - {w_3}^2)$$
$$\tau_z = K_m({w_1}^2 - {w_2}^2 + {w_3}^2 - {w_4}^2)$$

The variables $w_i^2,\ i = 1, 2, 3, 4$, are the squared angular velocities of the propellers. They represent the four direct inputs to the system and are are grouped in a single input vector U, as equation 12 shows in III-A. These and all other variables as well as all other constants are defined in the Appendix section.

III. Step 1: Linear Adaptive Control

A. Linearization

We will linearize the equations of motion about hover, so that the linearized state and control inputs have an equilibrium at X_{eq} with the linearized control input. We will establish the corresponding performance specifications for the system and design a linear feedback controller that will stabilize the system and meet established specifications.

We had previously defined the state dynamics as $\dot{X}$, where $\dot{X} = f(X, U)$ and $f \in \mathbb{R}^{12}$. At hover f must meet $f(X_{eq}, U_{eq}) = 0$. The equilibrium point at the hover, and the corresponding input to achieve it correspond to:

$$X_{eq} = [x_{eq}, y_{eq}, z_{eq}, 0, 0, \psi_{eq}, 0, 0, 0, 0, 0, 0]^\intercal$$
$$U_{eq} = [{w_{eq}}^2, {w_{eq}}^2, {w_{eq}}^2, {w_{eq}}^2]^\intercal$$

Where U_{eq} corresponds to the input U (Equation 12) needed to achieve hover.

$$U = [u_1, u_2, u_3, u_4]^\intercal = [w_1^2, w_2^2, w_3^2, w_4^2]^\intercal \tag{12}$$

Notice that to achieve hover, the rotational speeds of all the propellers must be equal. Also, the thrust T must counter the force due to gravity on the drone $(T = mg)$. From the definition of thrust (Equation 11) we get:

$$w_{eq} = \sqrt{\frac{mg}{4k_a}}$$

As we know, by linearizing a nonlinear time-invariant system we can only get local properties. In order to obtain the linear representation of the system $\Delta\dot{X} = A\Delta X + B\Delta U$ we need to get the Jacobian matrices A and B. Here, ΔX and ΔU are the perturbation about the equilibrium point. The actual state and input are re-defined as $X = X_{eq} + \Delta X$ and $U = U_{eq} + \Delta U$.

$$A = \frac{\partial f}{\partial X}(X, U)|_{(X_{eq}, U_{eq})}$$
$$B = \frac{\partial f}{\partial U}(X, U)|_{(X_{eq}, U_{eq})}$$

Utilizing the previous two equations and our system definition, $\dot{X} = f(X, U)$, we found A and B to be:

$$A = \begin{bmatrix} O_{6x6} & I_{6x6} \\ \Lambda & O_{6x6} \end{bmatrix} \tag{13}$$

$$B = \begin{bmatrix} O_{8x4} \\ \Delta \end{bmatrix} \tag{14}$$

Where O_{ixj} and I_{ixj} are the zero matrix and the identity matrix respectably, of size ixj. Λ and Δ are defined as follows:

$$\Lambda = \begin{bmatrix} 0 & 0 & 0 & g\sin(\psi_{eq}) & g\cos(\psi_{eq}) & 0 \\ 0 & 0 & 0 & -g\cos(\psi_{eq}) & g\sin(\psi_{eq}) & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\Delta = \begin{bmatrix} \frac{K_a}{m} & \frac{K_a}{m} & \frac{K_a}{m} & \frac{K_a}{m} \\ 0 & -\frac{K_a l}{I_x} & 0 & \frac{K_a l}{I_x} \\ \frac{K_a l}{I_y} & 0 & -\frac{K_a l}{I_y} & 0 \\ \frac{K_m}{I_z} & -\frac{K_m}{I_z} & \frac{K_m}{I_z} & -\frac{K_m}{I_z} \end{bmatrix}$$

B. Linear Controller design

With the linearization done, designing the linear controller is simple: we just need to find a gain matrix K such that the close-loop system has all the eigenvalues in the left half plane.

$$\Delta\dot{X} = A\Delta X + B\Delta U \tag{15}$$

$$\Delta U = -K\Delta X \tag{16}$$

Closing the loop:

$$\Delta\dot{X} = (A - BK)\Delta X$$

With ψ_{eq} as zero, we aimed to place all the Eigen Values of the close loop system completely in the $\mathbb{R}$ axis. After the pole placement, we obtained the gain matrix $K = [1\text{x}10^7 K_1, 1\text{x}10^5 K_2]$, here $K \in \mathbb{R}^{M\text{x}N}$ and K_1, K_2 are defined as follows:

$$K_1 = \begin{bmatrix} 1.32 & 1.18 & 2.61 & 2.46 & -1.41 & 1.83 \\ 1.62 & 1.25 & 1.93 & 2.04 & -1.81 & -1.02 \\ 1.54 & 1.24 & 2.61 & 2.54 & -2.04 & 1.83 \\ 1.68 & 1.32 & 1.94 & 2.41 & -1.89 & -1.03 \end{bmatrix}$$

$$K_2 = \begin{bmatrix} 86.80 & 108.79 & 64.44 & 15.47 & -4.90 & 48.56 \\ 106.61 & 104.75 & 55.63 & 10.27 & -9.12 & -37.69 \\ 107.86 & 112.99 & 64.49 & 15.96 & -12.56 & 48.52 \\ 110.72 & 114.92 & 55.76 & 15.48 & -9.61 & -37.69 \end{bmatrix}$$

C. Linear MIMO MRAC design

For this section, we will treat the perturbation ΔX and ΔU from Equation 15 as the state and the input itself for convenience sake.

We will purposely use incorrect parameter values of the Quadcopter in order to create an "Estimated Plant" with parameter mismatch. Then, we will show that the Adaptive Control laws let us track the Reference Model signal more precisely as compared to the Linear Control laws.

Estimated plant: $\dot{x}(t) = A_{pm}x(t) + B_{pm}u(t)$

Here pm stands for Parameter Mismatch, and $A_{pm} = A$ since the linearized matrix A doesn't depend on any parameter of the drone. We obtained B_{pm} assuming a miscalculation of m and l by a factor of 1.5, which gives us an overestimation of 50%.

Reference Model: $\dot{x}_m(t) = A_m x_m(t) + B_m r(t)$

For the reference model we used the real linearized model of our system, since this is how ideally the plant should behave. This means that $A_m = A$ and $B_m = B$. In our case $r(t)$ is the reference model control input that makes $\dot{x}_m$ achieve hover at the desired location (x,y,z) = (2,2,3).

Controller:

$$u(t) = K_x^\intercal(t)x(t) + K_r^\intercal(t)r(t) \tag{17}$$

Here $K_x \in \mathbb{R}^{NxM}$, $K_r \in \mathbb{R}^{MxM}$, where $N = 12$ is the dimension of our state-space, and $M = 4$ is the dimension of the input space.

Adaptation laws:

$$\dot{K}_x(t) = -\Gamma_x x(t)e^\intercal(t)PB \tag{18}$$

$$\dot{K}_r(t) = -\Gamma_r r(t)e^\intercal(t)PB \tag{19}$$

Where $e(t)$ is the tracking error defined as $e(t) = x(t) - x_m(t)$, P is the positive definite matrix found solving the Riccati Equation, and Γ is an arbitrary adaptation gain matrix. We are using the static matrix K that is being used in the Linear Controller as the initial condition of our dynamic matrix K_x and $I_{4\text{x}4}$ as the initial condition of K_r, hence $K_x^\intercal(0) = K$ and $K_r^\intercal(0) = I_{4\text{x}4}$.

D. Results

Below are the graphs of the response of the 12 linearized states for the Linear Controller in Equation 16. Here we can see the system converging to the desired hover position, where the first three states are $[x, y, z]^\intercal = [2, 2, 3]^\intercal$ and the rest are zero.

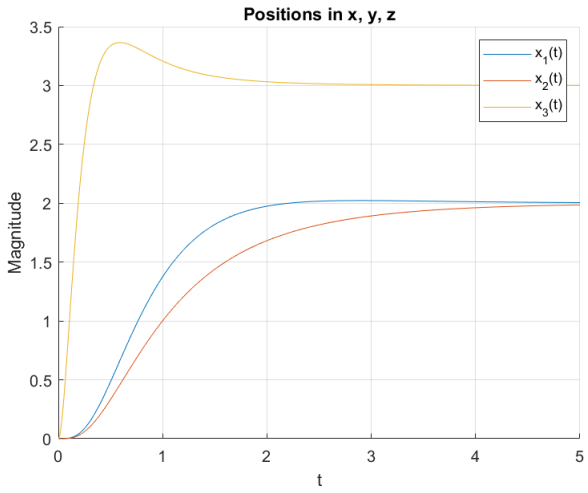


Fig. 1: States $[x_1, x_2, x_3]^\intercal = [x, y, z]^\intercal$

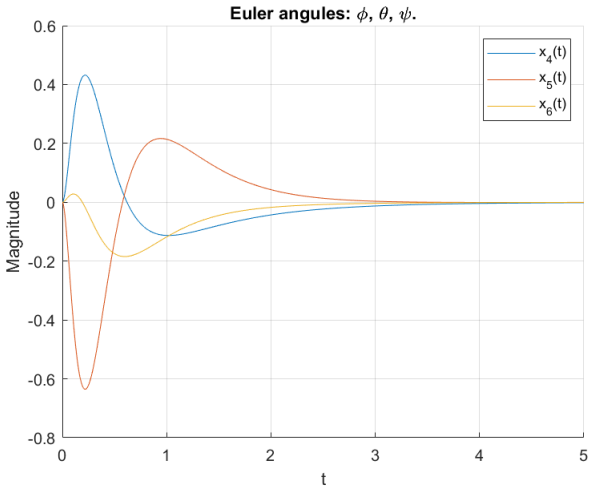


Fig. 2: States $[x_4, x_5, x_6]^\intercal = [\phi, \theta, \psi]^\intercal$

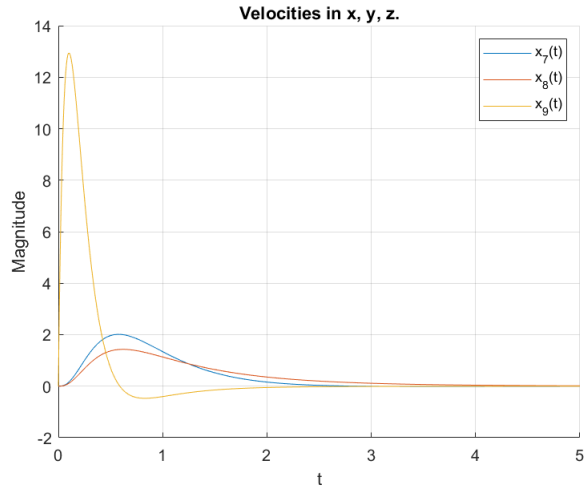


Fig. 3: States $[x_7, x_8, x_9]^\intercal = [\dot{x}, \dot{y}, \dot{z}]^\intercal$

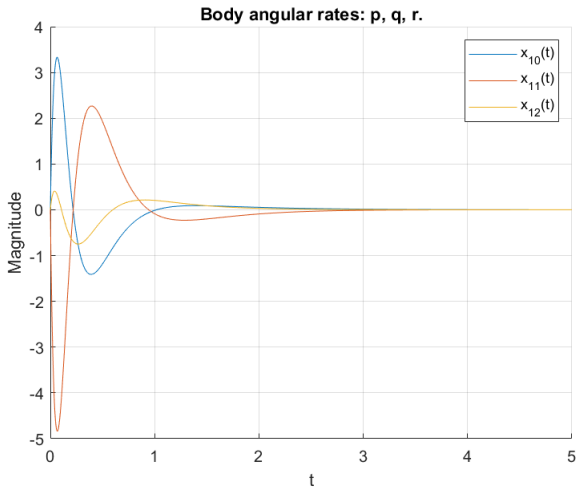


Fig. 4: States $[x_{10}, x_{11}, x_{12}]^\intercal = [p, q, r]^\intercal$

Now, the graphs corresponding to the response of the system with the linear MIMO MRAC, where we can see that the adaptive controller in Equation 17 - 19 is better able to track the reference model than the regular linear controller in Equations 16.

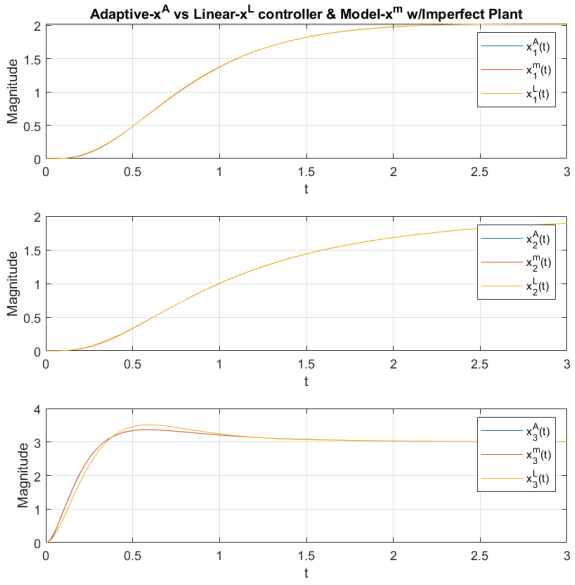


Fig. 5: States $[x_1, x_2, x_3]^\intercal = [x, y, z]^\intercal$

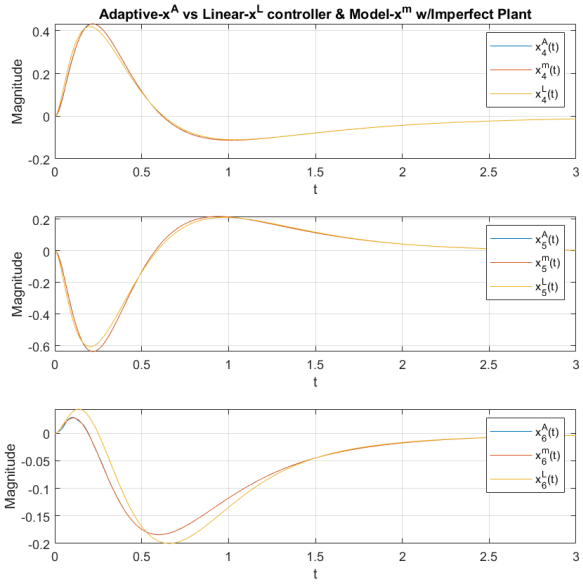


Fig. 6: States $[x_4, x_5, x_6]^\intercal = [\phi, \theta, \psi]^\intercal$

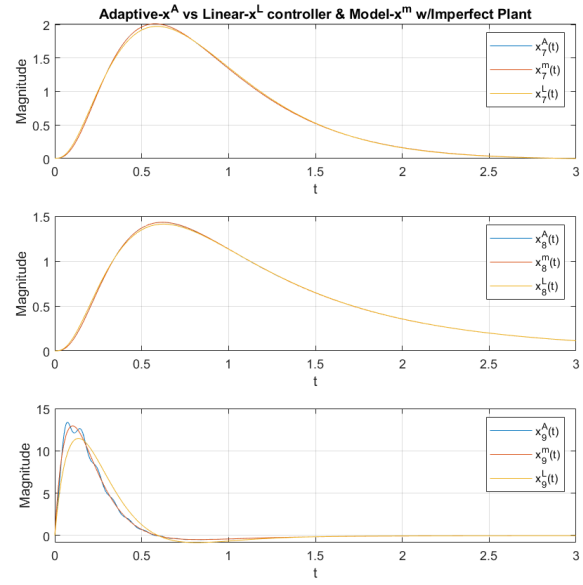


Fig. 7: States $[x_7, x_8, x_9]^\intercal = [\dot{x}, \dot{y}, \dot{z}]^\intercal$

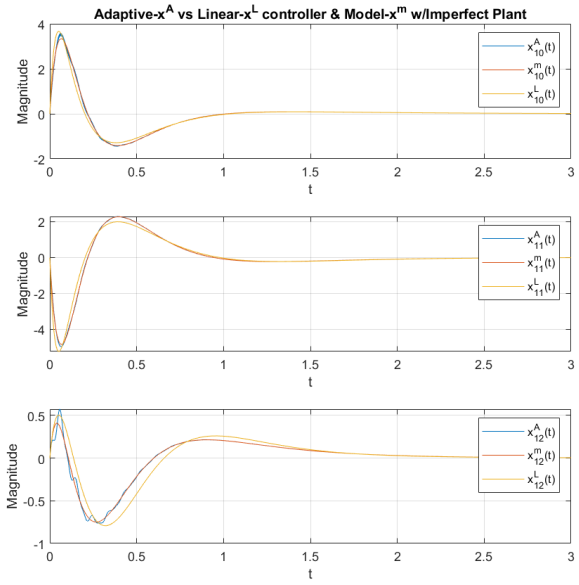


Fig. 8: States $[x_{10}, x_{11}, x_{12}]^\intercal = [p, q, r]^\intercal$

IV. Step 2: Nonlinear Controller

In this section, we are going to augment our linear adaptive controller from the last section so that it can control the actual nonlinear plant. Again, the MRAC approach will be used here, but with the addition of parametrized nonlinearities. Before diving into controller designs, we are going to do a coordinate transformation that will allow us to do our design easily. With this transformation, it is possible to put all the nonlinear terms in the span of the control input. The goal is to get the system into the following form:

$$\dot{X} = AX + B[U + \alpha^\intercal \Phi(X)]$$

With this form, it will be straightforward to design the MIMO MRAC controller.

A. *Differential flatness: Transformation of the State*

This system and many other engineered or man-made mobile vehicles have the property of differential flatness. There exists a transformation of state that will provide full control of the position variables if control over the orientation is relaxed. Let q be a vector representing the position of the center of mass of the drone in the world inertial frame, i.e.:

$$q = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Define e_3 as the third column of the 3D rotation matrix, R.

$$e_3 = \begin{bmatrix} -\cos x_4 \sin x_5 \cos x_6 - \sin x_4 \sin x_6 \\ -\cos x_4 \sin x_5 \sin x_6 + \sin x_4 \cos x_6 \\ -\cos x_4 \cos x_5 \end{bmatrix}$$

The system can then be rewritten as:

$$m\ddot{q} = e_3 \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} K_a u_1 \\ K_a u_2 \\ K_a u_3 \\ K_a u_4 \end{bmatrix} + m \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}$$

Let's define the following coordinate transformation:

$$q' = q + \lambda e_3 \tag{20}$$

Where λ is a positive real constant.
The new transformed states then become:

$$\begin{aligned} x_1' = x' &= x_1 + \lambda(-\cos x_4 \sin x_5 \cos x_6 - \sin x_4 \sin x_6) \\ x_2' = y' &= x_2 + \lambda(-\cos x_4 \sin x_5 \sin x_6 + \sin x_4 \cos x_6) \\ x_3' = z' &= x_3 + \lambda(-\cos x_4 \cos x_4) \\ x_7' &= \dot{x_1'} \\ x_8' &= \dot{x_2'} \\ x_9' &= \dot{x_3'} \end{aligned}$$

1) New State Dynamics: We need to obtain the dynamics equations for the transformed states. Below are some key results from the mathematical derivation:

$$\begin{aligned} \dot{x_1}' = \dot{x_1} + \lambda[&\sin x_4 \sin x_5 \cos(x_6) x_{10} - \cos x_4 \sin(x_6) x_{10} \\ &- \cos x_5 \cos(x_6) x_{11}] \end{aligned}$$

$$\begin{aligned} \dot{x_7'} &= \ddot{x_1'} \\ &= \left[-\tfrac{1}{m}T + \lambda(x_{10}^2 + x_{11}^2)\right]\left[c(x_4)s(x_5)c(x_6) - s(x_4)s(x_6)\right] \\ &\quad + \lambda\left[\tfrac{-I_x+I_y-I_z}{I_x} x_{11}x_{12} + \tfrac{\tau_x}{I_x}\right]\left[s(x_4)s(x_5)c(x_6) - c(x_4)s(x_6)\right] \\ &\quad - \lambda\left[\tfrac{-I_x+I_y+I_z}{I_y} x_{10}x_{12} + \tfrac{\tau_y}{I_y}\right]\left[\cos x_5 \cos x_6\right] \end{aligned}$$

$$\begin{aligned} \dot{x_2}' = \dot{x_2} + \lambda[&\sin x_4 \sin x_5 \sin(x_6) x_{10} + \cos x_4 \cos(x_6) x_{10} \\ &- \cos x_5 \sin(x_6) x_{11}] \end{aligned}$$

$$\begin{aligned} \dot{x_8'} &= \ddot{x_2'} \\ &= \left[-\tfrac{1}{m}T + \lambda(x_{10}^2 + x_{11}^2)\right]\left[\cos x_4 \sin x_5 \sin x_6 - \sin x_4 \cos x_6\right] \\ &\quad + \lambda\left[\tfrac{-I_x+I_y-I_z}{I_x} x_{11}x_{12} + \tfrac{\tau_x}{I_x}\right]\left[\sin x_4 \sin x_5 \sin x_6 - \cos x_4 \cos x_6\right] \\ &\quad - \lambda\left[\tfrac{-I_x+I_y+I_z}{I_y} x_{10}x_{12} + \tfrac{\tau_y}{I_y}\right]\left[\cos x_5 \sin x_6\right] \end{aligned}$$

$$\dot{x_3}' = \dot{x_3} + \lambda[\sin x_4 \cos x_5 x_{10} + \sin x_5 x_{11}]$$

$$\begin{aligned} \dot{x_9'} &= \ddot{x_3'} \\ &= \left[-\tfrac{1}{m}T + \lambda(x_{10}^2 + x_{11}^2)\right]\left[\cos x_4 \cos x_5\right] \\ &\quad + \lambda\left[\tfrac{-I_x+I_y-I_z}{I_x} x_{11}x_{12} + \tfrac{\tau_x}{I_x}\right]\left[\sin x_4 \cos x_5\right] \\ &\quad - \lambda\left[\tfrac{-I_x+I_y+I_z}{I_y} x_{10}x_{12} + \tfrac{\tau_y}{I_y}\right]\left[\sin x_5\right] + g \end{aligned} \tag{21}$$

The next step is to transform g so that it can be factored into the span of the B matrix as well. By inspection, one can find that by decomposing g into the following three terms:

$$\begin{aligned} &\cos x_4 \cos x_5 g, \\ &\sin x_4 \cos x_5 g, \\ &\sin x_5 g \end{aligned}$$

Plugging them into the $\dot{x_7}'$, $\dot{x_8}'$, $\dot{x_9}'$ expressions, the g term in the original equation 21 is successfully factored into the span of B. Finally, we can write out the state dynamics for the transformed states (where c(*) and s(*) represent cos(*) and sin(*), respectively):

$$\dot{x_1}' = x_7'$$

$$\begin{aligned} \dot{x_7'} &= \ddot{x_1'} \\ &= \left[-\tfrac{1}{m}T + \lambda(x_{10}^2 + x_{11}^2) + c(x_4)c(x_5)g\right] \\ &\quad \left[c(x_4)s(x_5)c(x_6) - s(x_4)s(x_6)\right] \\ &\quad + \left[\lambda(\tfrac{I_y-I_z-I_x}{I_x} x_{11}x_{12} + \tfrac{\tau_x}{I_x}) + s(x_4)c(x_5)g\right] \\ &\quad \left[s(x_4)s(x_5)c(x_6) - c(x_4)s(x_6)\right] \\ &\quad - \left[\lambda(\tfrac{-I_x+I_y+I_z}{I_y} x_{10}x_{12} + \tfrac{\tau_y}{I_y}) + s(x_5)g\right]\left[c(x_5)c(x_6)\right] \end{aligned}$$

$$\dot{x_2}' = x_8'$$

$$\begin{aligned} \dot{x_8'} &= \ddot{x_2'} \\ &= \left[-\tfrac{1}{m}T + \lambda(x_{10}^2 + x_{11}^2) + c(x_4)c(x_5)g\right] \\ &\quad \left[c(x_4)s(x_5)s(x_6) - s(x_4)c(x_6)\right] \\ &\quad + \left[\lambda(\tfrac{-I_x+I_y-I_z}{I_x} x_{11}x_{12} + \tfrac{\tau_x}{I_x}) + s(x_4)c(x_5)g\right] \\ &\quad \left[s(x_4)s(x_5)s(x_6) - c(x_4)c(x_6)\right] \\ &\quad - \left[\lambda(\tfrac{-I_x+I_y+I_z}{I_y} x_{10}x_{12} + \tfrac{\tau_y}{I_y}) + s(x_5)g\right]\left[c(x_5)s(x_6)\right] \end{aligned}$$

$$\dot{x_3}' = x_9'$$

$$\begin{aligned} \dot{x_9'} &= \ddot{x_3'} \\ &= \left[-\tfrac{1}{m}T + \lambda(x_{10}^2 + x_{11}^2) + c(x_4)c(x_5)g\right]\left[c(x_4)c(x_5)\right] \\ &\quad + \left[\lambda(\tfrac{-I_x+I_y-I_z}{I_x} x_{11}x_{12} + \tfrac{\tau_x}{I_x}) + s(x_4)c(x_5)g\right]\left[s(x_4)c(x_5)\right] \\ &\quad + \left[\lambda(\tfrac{-I_x+I_y+I_z}{I_y} x_{10}x_{12} + \tfrac{\tau_y}{I_y}) + s(x_5)g\right]\left[s(x_5)\right] \end{aligned}$$

2) Rewriting State Dynamics in Matrix Form: With the new state vector defined as

$$X' = \begin{bmatrix} x_1' & x_2' & x_3' & x_7' & x_8' & x_9' \end{bmatrix}^\intercal$$

we can write the entire state dynamics in the following matrix form:

$$\dot{X}' = A''X' + B''[U + \alpha^\intercal \Phi(X')]$$

Where,

$$\begin{aligned} &B''[U + \alpha^\intercal \Phi] \\ &= B'' \begin{bmatrix} -\tfrac{1}{m}T + \lambda(x_{10}^2 + x_{11}^2) + c(x_4)c(x_5)g \\ \lambda(\tfrac{-I_x+I_y-I_z}{I_x} x_{11}x_{12} + \tfrac{\tau_x}{I_x}) + s(x_4)c(x_5)g \\ -\lambda(\tfrac{-I_x+I_y+I_z}{I_y} x_{10}x_{12} + \tfrac{\tau_y}{I_y}) + s(x_5)g \end{bmatrix} \end{aligned} \tag{22}$$

$$A'' = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$B'' = \begin{bmatrix} 0_{3x3} \\ R \end{bmatrix}$$

where,

$$R = \begin{bmatrix} R_1 & R_2 & R_3 \end{bmatrix}$$

$$R_1 = \begin{bmatrix} c(x_4)s(x_5)c(x_6) \\ c(x_4)s(x_5)s(x_6) - s(x_4)c(x_6) \\ c(x_4)c(x_5) \end{bmatrix} = \begin{bmatrix} a \\ d \\ g \end{bmatrix}$$

$$R_2 = \begin{bmatrix} s(x_4)s(x_5)c(x_6) - c(x_4)s(x_6) \\ s(x_4)s(x_5)s(x_6) + c(x_4)c(x_6) \\ s(x_4)c(x_5) \end{bmatrix} = \begin{bmatrix} b \\ e \\ h \end{bmatrix}$$

$$R_3 = \begin{bmatrix} c(x_5)c(x_6) \\ c(x_5)s(x_6) \\ -s(x_5) \end{bmatrix} = \begin{bmatrix} c \\ f \\ i \end{bmatrix}$$

$$\Phi(X') = \begin{bmatrix} x_{10}^2 + x_{11}^2 \\ c(x_4)c(x_5) \\ x_{11}x_{12} \\ s(x_4)c(x_5) \\ x_{10}x_{11} \\ s(x_5) \end{bmatrix}$$

Finally, we can obtain the desired form of B$[U + \alpha^\intercal \Phi(X')]$ by manipulating equation 22:

$$B''[U + \alpha^\intercal \Phi(X')] = B''[\Lambda U + \Upsilon \Phi(X')]$$

Where,

$$\Lambda = \begin{bmatrix} -\frac{K_a}{m} & \frac{K_a}{m} & \frac{K_a}{m} & \frac{K_a}{m} \\ 0 & -\frac{\lambda K_a l}{I_x} & 0 & \frac{\lambda K_a l}{I_x} \\ -\frac{\lambda K_a l}{I_y} & 0 & \frac{\lambda K_a l}{I_y} & 0 \end{bmatrix}$$

$$\Upsilon = \begin{bmatrix} \lambda & g & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{\lambda(-I_x + I_y - I_z)}{I_x} & g & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{-\lambda(-I_x + I_y - I_z)}{I_x} & g \end{bmatrix}$$

Then, we can rewrite the system as:

$$\dot{X}' = A'X' + B'[U + \alpha^\intercal \Phi(X')]$$

Where,

$$A' = A''$$

$$B' = \begin{bmatrix} 0_{3x3} \\ \kappa \end{bmatrix}$$

$$\kappa = \begin{bmatrix} \kappa_1 & \kappa_2 \end{bmatrix}$$

$$\kappa_1 = \begin{bmatrix} -\frac{K_a}{m}a - \frac{\lambda K_a l}{I_y}c, -\frac{K_a}{m}a - \frac{\lambda K_a l}{I_x}b, \\ -\frac{K_a}{m}d - \frac{\lambda K_a l}{I_y}f, -\frac{K_a}{m}d - \frac{\lambda K_a l}{I_x}e, \\ -\frac{K_a}{m}g - \frac{\lambda K_a l}{I_y}i, -\frac{K_a}{m}g - \frac{\lambda K_a l}{I_x}h, \end{bmatrix}$$

$$\kappa_2 = \begin{bmatrix} -\frac{K_a}{m}a - \frac{\lambda K_a l}{I_y}c, -\frac{K_a}{m}a - \frac{\lambda K_a l}{I_x}b \\ -\frac{K_a}{m}d - \frac{\lambda K_a l}{I_y}f, -\frac{K_a}{m}d - \frac{\lambda K_a l}{I_x}e \\ -\frac{K_a}{m}g - \frac{\lambda K_a l}{I_y}i, -\frac{K_a}{m}g - \frac{\lambda K_a l}{I_x}h \end{bmatrix}$$

$$\alpha^\intercal = \Lambda^+ \Upsilon \,.$$

Here, Λ^+ represents the Moore-Penrose pseudo inverse of Λ. This is the final transformed state dynamics in matrix form.

B. Nonlinear MIMO MRAC Design

Reference Model (in the transformed coordinates):

$$\dot{X}'_m = A_m X'_m + B_m r(t)$$

Adaptive Controller (in the transformed coordinates):

$$e'(t) = X'(t) - X'_m(t)$$
$$u'(t) = K_x^{'\intercal} X'(t) + K_r^{'\intercal} r(t) - \hat{\alpha}^{'\intercal} \Phi(X)$$
$$\dot{K}'_x(t) = -\Gamma_x X'(t) e^{'\intercal} P B'$$
$$\dot{K}'_r(t) = -\Gamma_r r'(t) e^{'\intercal} P B'$$
$$\dot{\hat{\alpha}}'(t) = \Gamma_\alpha \Phi'(X) e^{'\intercal} P B'$$

In order to express the control input in the original coordinates, i.e. convert $u'(t)$ to $u(t)$, we just need to do the following. The only part in $u'(t)$ that is expressed in the transformed coordinates is $X'(t)$. Therefore, if we replace $X'(t)$ by the coordinate transformation from $X'(t)$ to $X(t)$ as follow:

$$x'_1 = x' = x_1 + \lambda[-c(x_4)s(x_5)c(x_6) - s(x_4)s(x_6)]$$

$$x'_2 = y' = x_2 + \lambda[-c(x_4)s(x_5)s(x_6) + s(x_4)c(x_6)]$$

$$x'_3 = z' = x_3 + \lambda[-c(x_4)c(x_4)]$$

$$x'_7 = x_7 + \lambda[s(x_4)s(x_5)c(x_6)x_{10} - c(x_4)s(x_6)x_{10} - c(x_5)c(x_6)x_{11}]$$

$$x'_8 = x_8 + \lambda[s(x_6)(s(x_4)s(x_5)x_{10} - c(x_5)x_{11}) + c(x_4)c(x_6)x_{10}]$$

$$x'_9 = x_9 + \lambda[s(x_4)c(x_5)x_{10} + s(x_5)x_{11}]$$

Then the control input will depend on $X(t)$ instead of $X'(t)$. Therefore, we get the expression for $u(t)$ with everything expressed in the original coordinates.

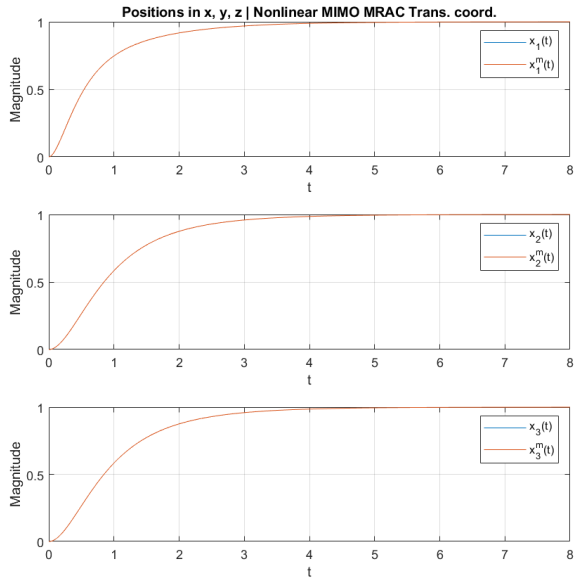


Fig. 9: States trans. coord. $[x'_1, x'_2, x'_3]^\intercal = [x', y', z']^\intercal$

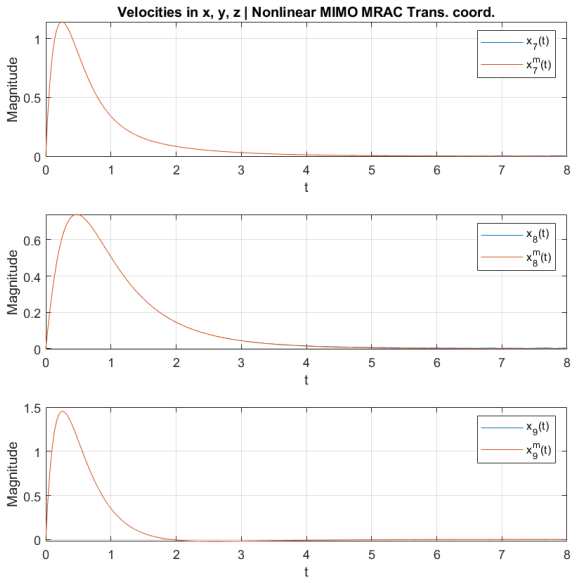


Fig. 10: States trans. coord. $[x'_7, x'_8, x'_9]^\intercal = [\dot{x}', \dot{y}', \dot{z}']^\intercal$

C. Results

The simulation plots for the nonlinear MIMO MRAC controller (Fig. 9 to Fig. 12) and the fixed-gain linear controller (Fig. 13), as well as the response of the nonlinear system to the Linear MIMO MRAC (Fig. 14 and Fig. 15) are provided in this section for comparison. The plant parameter mismatch is the same as before (overestimation by 50%). The plots include graphs of the states of the actual plant $x_{no.}(t)$, and the corresponding states of the model reference $x_{m_{no.}}(t)$, in both the transformed coordinates and the original coordinates. Now, the desired hover state is set at $[x, y, z] = [1, 1, 1]$, and λ is 0.01, which means that the virtual point is just above the drone's center of mass.

In the case of the system in transformed coordinates (Fig. 9 and Fig. 10), the plant states track the reference states perfectly and both of them converge to the ideal values quickly and smoothly.

Now for the original coordinates, the three position states converge to the ideal values relatively fast (Fig. 11). However, they have small non-vanishing oscillations. As to the other three velocity states (Fig. 12), they also converge to zero, but with much larger yet bounded oscillations.

In the case of the response of the nonlinear system to the fixed-gain linear controller many states (both translational and angular) blow up and hence the system is not stabilized. This results are obvious since linearization only works locally in a neighborhood close to the equilibrium.

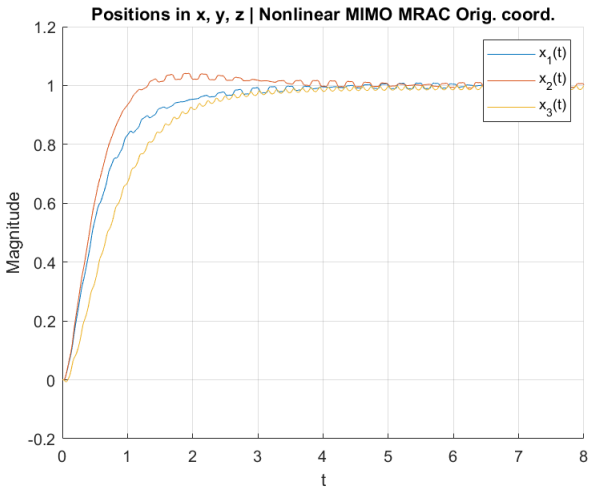


Fig. 11: States orig. coord. $[x_1, x_2, x_3]^\intercal = [x, y, z]^\intercal$

Finally, the state trajectories of the linear MRAC controller applied to the nonlinear plant are displayed in Fig. 14 and Fig. 15. The system blows up as well. This shows the linear MRAC controller is unable to achieve what the nonlinear MRAC does for the nonlinear plant. However, it performs better than the fixed-gain linear controller. This makes sense since it tries to adapt to the unstable situation and thus slows down the blowing-up process.

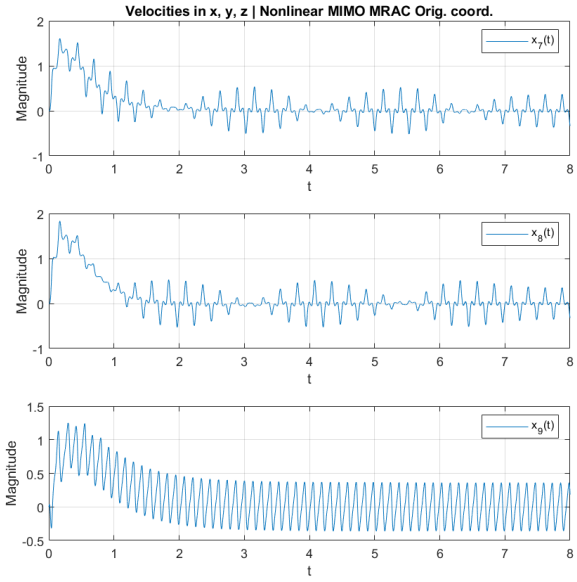


Fig. 12: States orig. coord. $[x_7, x_8, x_9]^\intercal = [\dot{x}, \dot{y}, \dot{z}]^\intercal$

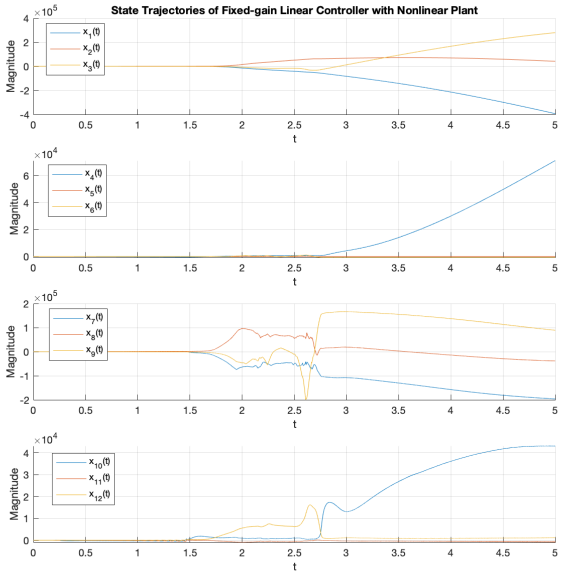


Fig. 13: Fixed-gain linear controller applied to nonlinear dynamics

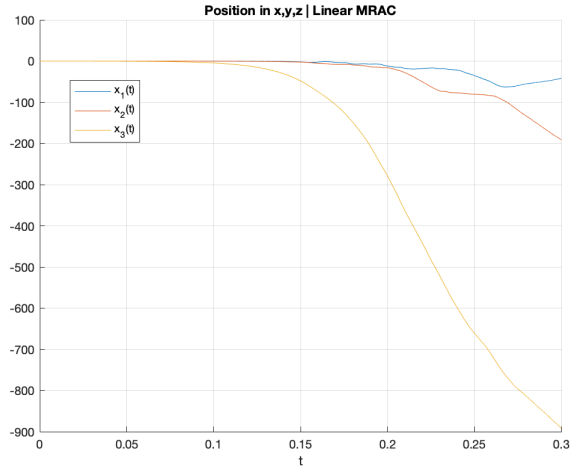


Fig. 14: Linear MRAC controller applied to nonlinear dynamics

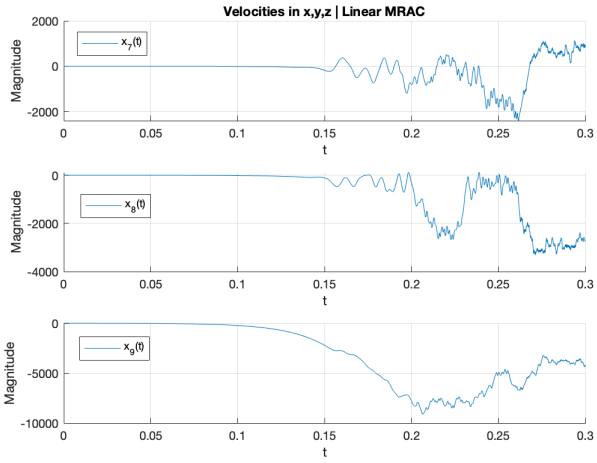


Fig. 15: Linear MRAC controller applied to nonlinear dynamics

V. Appendix

A. Variables

Definition of the variables used throughout the report:
w_i : Angular velocity of propeller i, measured in RPM.
T : Total thrust of the drone.
τ_x : Torque about the x axis.
τ_y : Torque about the y axis.
τ_z : Torque about the z axis.
I_x : Momentum of inertia in the x axis.
I_y : Momentum of inertia in the y axis.
I_z : Momentum of inertia in the z axis.
m : Mass of the drone.
l : Length of each of the quadcopter's arms.
g : Acceleration due gravity.
K_a : Aerodynamic value dependent on air density (constant at low altitudes), thrust coefficient (constant), drone and rotor geometry (constants). Used to calculate thrust exerted by each rotor.
K_m : Aerodynamic value dependent on air density (constant at low altitudes), torque coefficient (constant), drone and rotor geometry (constants). Used to calculate torque exerted by each rotor.

B. Constants

Definitions of the parameters used throughout the report which are specific to the drone being used, the Parrot Rolling Spider:

$m = 68$ grams
$l = 6.24$ cm
$g = 9.81\ \frac{m}{s^2}$
$K_a = 4.72\text{x}10^{-8}\ kg \cdot m$
$K_m = 1.1393\text{x}10^{-10}\ kg \cdot m^2$
$I_x = 6.86\text{x}10^{-5}\ kg \cdot m^2$
$I_y = 9.2\text{x}10^{-5}\ kg \cdot m^2$
$I_z = 1.366\text{x}10^{-4}\ kg \cdot m^2$

VI. REFERENCE

Lyu, Haifeng, "Multivariable Control of a Rolling Spider Drone" (2017). Open Access Master's Theses. Paper 1064. http://digitalcommons.uri.edu/theses/1064