NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

Unmanned Aerial Vehicles
Master in Aerospace Engineering

Crazyflie Drone Modelling and Identification

David Madeira Nº 67143
João Costa Nº 67315

Prof. Bruno Guerreiro

April
2024

1 Introduction

Unmanned Aerial Vehicles, or UAVs for short, have been playing a major role in modern days. From amateur and professional photography to country border control, UAVs are rising in popularity, but they wouldn't be available if not for control techniques, models and algorithms.

This project aims at obtaining a realistic model of the Crazyflie 2.1 drone for simulation and control design purposes.

Link for the MatLab files.

2 Crazyflie modeling

Non-linear Model

1.1 Obtain the rigid-body nonlinear model of the drone, considering the state variables above.

$$\dot{x} = \begin{bmatrix} \dot{p} \\ \dot{v} \\ \dot{\lambda} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} f_p(\lambda, v)) \\ f_v(v, u, \lambda, \omega) \\ f_\lambda(\lambda, \omega) \\ f_\omega(\omega, u) \end{bmatrix} = \begin{bmatrix} \mathbf{R}(\lambda)\mathbf{v} \\ -\mathbf{S}(\boldsymbol{\omega})\mathbf{v} - \mathbf{g} \cdot \mathbf{R^T} \cdot \mathbf{z_I} + \frac{\mathbf{1}}{\mathbf{m}}(\mathbf{f_d} + \mathbf{f_p}) \\ \mathbf{Q}(\lambda)\boldsymbol{\omega} \\ -\mathbf{J^{-1}}\mathbf{S}(\boldsymbol{\omega})\mathbf{J}\boldsymbol{\omega} + \mathbf{J^{-1}}(\mathbf{n_d} + \mathbf{n_p}) \end{bmatrix} \tag{1}$$

Being f_p the function that represents *Thrust* of all motors and f_d the function responsible for adding *Drag*.

1.2 Obtain the Earth gravity force vector, $\mathbf{f}_g$, and air drag force vector, $\mathbf{f}_a$, described at the body frame, considering the respective moments to be negligible.

Table 1 shows the constants used and the value used throughout this report.

Quantity	Symbol	Value	Unit
Drag Coefficient	C_p	1	–
Mass	m_R	0.027	kg
Air density	ρ	1.125	kg/m^3
Gravity	g	9.81	m/s^2
Drone's length	l_d	0.0567	m
Drone's width	w_d	0.0567	m
Drone's height	h_d	0.0290	m
Drone's arm length	a_d	0.0250	m

Table 1: Constants

For simplicity purposes the Drag Coefficient was assumed as 1.

Given that $\beta = -\frac{1}{2} \cdot C_p \cdot A \cdot \rho$, and each side of the vehicle has a different area, we obtain:

$$\beta = \begin{bmatrix} \beta_x & 0 & 0 \\ 0 & \beta_y & 0 \\ 0 & 0 & \beta_z \end{bmatrix} = \begin{bmatrix} -\frac{1}{2} \cdot C_p \cdot A_x \cdot \rho & 0 & 0 \\ 0 & -\frac{1}{2} \cdot C_p \cdot A_y \cdot \rho & 0 \\ 0 & 0 & -\frac{1}{2} \cdot C_p \cdot A_z \cdot \rho \end{bmatrix} \tag{2}$$

Considering,

$$A_x = A_y = l_d \times h_d \qquad \text{and} \qquad A_z = l_d \times w_d \tag{3}$$

The Earth gravity described at the body frame can be written as follows:

$$\begin{aligned} {}_B\mathbf{g} &= gR^T = gR(\psi)R(\theta)R(\phi)z_I \\ &= g \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \end{aligned} \tag{4}$$

1.3 Obtain the propulsion force vector, $\mathbf{f}_p$, and moment vector, $\mathbf{n}_p$ expressions at the vehicle center of mass (origin of the body frame) as a function of the individual thrust forces generated by each of the 4 rotors, $\mathbf{u} = [T_1\ T_2\ T_3\ T_4]$.

Assuming $\mathbf{u} = [T_1\ T_2\ T_3\ T_4]$ and $C_\mathbf{M} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$, we get:

$$\text{Mu} = \begin{bmatrix} f_1 & f_2 & f_3 & f_4 \end{bmatrix} \tag{5}$$

$$\text{fp} = \text{Mu} \times \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix} \tag{6}$$

$$\text{np} = \begin{bmatrix} n_1 + \mathbf{S}(T_1) \cdot f_1 & n_2 + \mathbf{S}(T_2) \cdot f_3 & n_3 + \mathbf{S}(T_3) \cdot f_3 & n_4 + \mathbf{S}(T_4) \cdot f_4 \end{bmatrix} \times \begin{bmatrix} T1 \\ T2 \\ T3 \\ T4 \end{bmatrix} \tag{7}$$

1.4 Obtain the 4 × 4 matrix that maps the four individual forces of the rotors uT into the control input vector u = $[T\ n_p]^T$, where $T = T1 + T2 + T3 + T4$ is the total thrust, and $n_p = [n_x\ n_y\ n_z]^T$ are the moments generated by the propulsion of the 4 rotors.

As we want next to make the drone go up with a smalll thrust, our total Thrust has to be the force of gravity in addition with that small thrust. So the control input vector comes:

$$\mathbf{u} = \begin{bmatrix} m \cdot g + T_0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \text{with } T_0 = 0.01\text{N} \tag{8}$$

This is a reasonable value for T_0 as it represents around 3% of the equilibrium thrust (=m·g)

1.5 Obtain the final nonlinear model of the drone and simulate the model in Matlab/Octave, considering initial resting state vector and small constant thrust, equal to all rotors.

Finally, we can simulate the final linear model, and our results are presented in the figure 1.

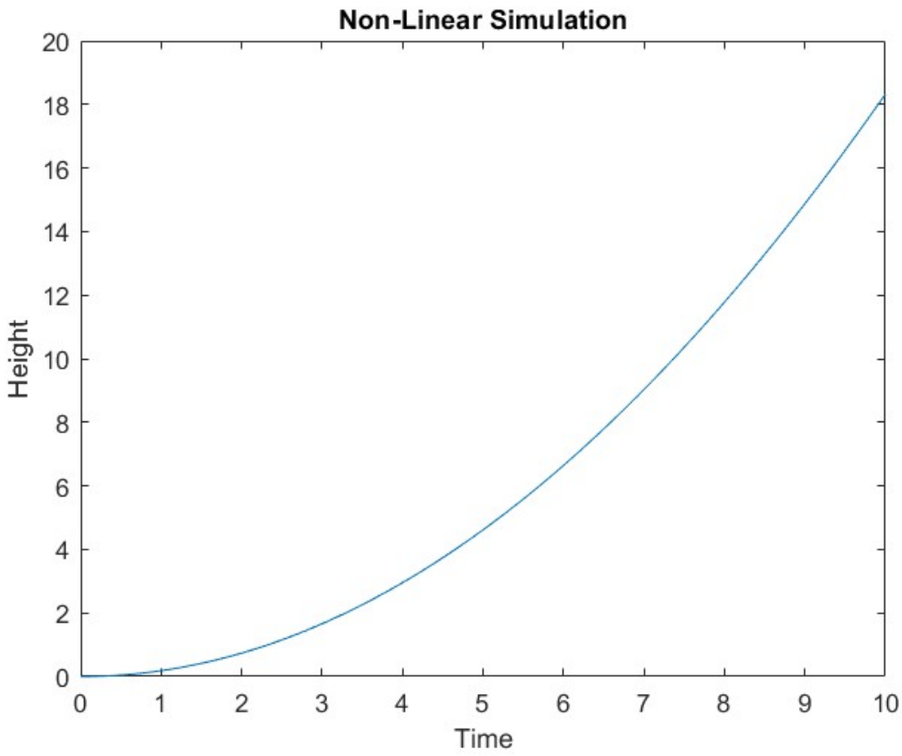

Figure 1: Simulation of the Non Linear Model

Modes and linearized models

After having a complete nonlinear model of the vehicle we can analyze the main modes of the vehicle individually. For simplicity, assume a diagonal inertia matrix, $\mathbf{J} = diag(J_x, J_y, J_z)$ and the following operation modes and equilibrium conditions:

OP.1 Hover: zero linear and angular velocity;

OP.2 Horizontal flight: zero vertical velocity and angular velocity, and constant horizontal velocity.

The perturbations to the equilibrium conditions can be defined for the state as $\widetilde{\mathbf{x}} = \mathbf{x} - \mathbf{x}_e$, and for the input as $\widetilde{\mathbf{u}} = \mathbf{u} - \mathbf{u}_e$.

1.6 Obtain the equilibrium conditions, $\mathbf{x_e}$ and $\mathbf{u_e}$ for each operation mode.

For hover conditions(OP1) we have no horizontal velocity so we just have to make sure the quad-rotor maintains its position at an arbitrary height (for this part we considered 1.5m), hence:

$$F_R = m\ddot{p} = T - m \cdot g = T_l + \delta T - m \cdot g \Leftrightarrow \delta T = 0 \tag{9}$$

The variable T_l denotes the "linearization" thrust, defined as the product of the mass m and the gravitational acceleration g ($T_l = m \cdot g$). This definition is integral to the linearization process, as the resultant linearized thrust f_p incorporates this term.

$$\mathbf{x_e} = \begin{bmatrix} p_x \\ p_y \\ p_z \\ v_x \\ v_y \\ v_z \\ \phi \\ \theta \\ \psi \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \overset{\text{OP1}}{=} \begin{bmatrix} 0 \\ 0 \\ 1.5 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad \mathbf{u_e} = \begin{bmatrix} \delta T \\ np_x \\ np_y \\ np_z \end{bmatrix} \overset{\text{OP1}}{=} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{10}$$

For the quad-rotor in horizontal conditions (OP2) w'll have 2 equations, one describing horizontal flight and other the vertical flight:

$$g \cdot sin(\theta) - \frac{A_l \cdot \rho \cdot v^2}{m} = 0 \tag{11}$$

$$-g \cdot \cos(\theta) + \frac{1}{m} \cdot T = 0 \tag{12}$$

It was considered that 5^{o} would be a reasonable θ for pitch, resulting in the following:

$$T = 0.2639 \text{ N and } v_x = 3,387 \text{ m/s} \tag{13}$$

$$\mathbf{x_e} = \begin{bmatrix} p_x \\ p_y \\ p_z \\ v_x \\ v_y \\ v_z \\ \phi \\ \theta \\ \psi \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \overset{\text{OP2}}{=} \begin{bmatrix} 0 \\ 0 \\ 1.5 \\ 3,387 \\ 0 \\ 0 \\ 0 \\ 5 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad \mathbf{u_e} = \begin{bmatrix} \delta T \\ np_x \\ np_y \\ np_z \end{bmatrix} \overset{\text{OP2}}{=} \begin{bmatrix} T_l - 0.2639 * \cos(5) \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{14}$$

1.7 Obtain the linearized models ($\dot{\tilde{x}} = A_{OPi}\tilde{x} + B_{OPi}\tilde{u}$) around the above equilibrium points/trajectories, considering Z-Y-X Euler angles kinematics.

The results of each linearized model - obtained previously - are explicit in the figures 2 and 3.

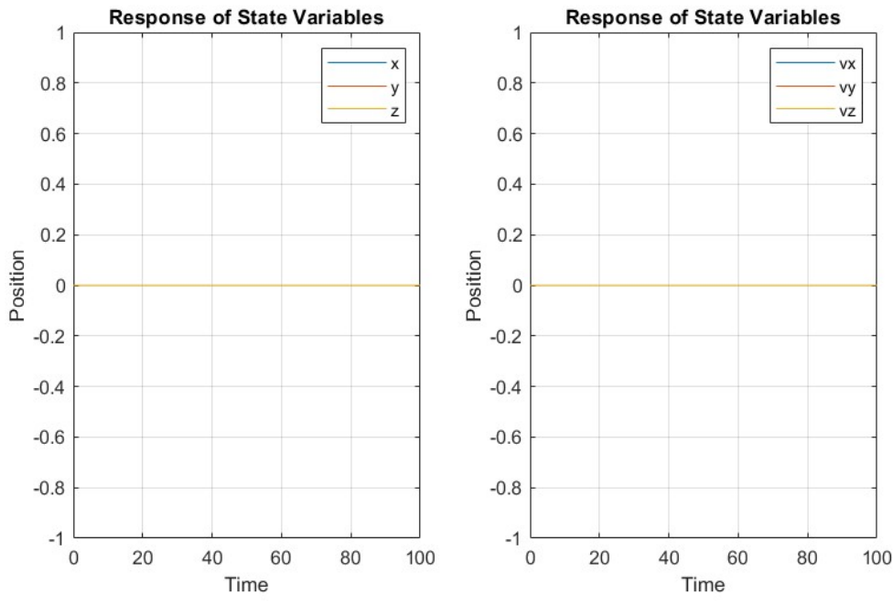

Figure 2: Simulation of Hover Conditions (OP1)

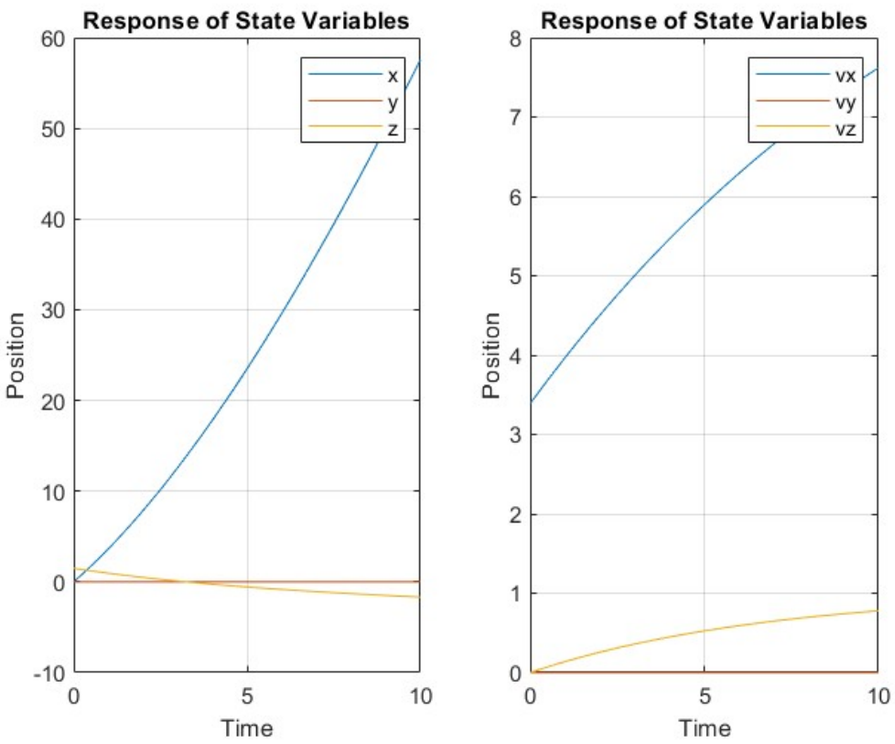

Figure 3: Simulation of Horizontal Flight (OP2)

As we can see by the figure 2 the hover conditions were obtained as predicted.

The horizontal flight, which can be viewed in the figure 3, presents satisfactory performance. It can be observed as well a gradual decline in altitude over time. Notably, the forward velocity v_x showed a tendency to increase, resulting in exponential longitudinal displacement x growth. This behaviour is facilitated by the constant thrust output over time, enabling exponential x expansion and proportional altitude reduction z.

Furthermore, the descent rate during horizontal flight demonstrated that the aircraft descends approximately 2 centimeters for every 1 meter travelled horizontally, underscoring reasonable flight dynamics achieved.

1.8 Decompose each linearized model into modes and discuss the results.

Points 1.8 and 1.9 are presented simultaneously above.

1.9 Analyze each linearized model regarding their controllability, observability and stability.

The controllability, observability and stability of the linearized models can be verified through the linearized matrices A, B, and C.

$$\mathbf{C_{trb}} = \begin{bmatrix} B & AB & A^{n-1}B \end{bmatrix} \tag{15}$$

$$\mathbf{O_{bsv}} = \begin{bmatrix} C \\ CA \\ CA^{n-1} \end{bmatrix} \tag{16}$$

If both matrices above have a rank higher or equal to the number of states of the system (in this case, 12), then the system is controllable and observable. As for stability, if all eigenvalues of A are higher or equal to zero. For the case at hand, as expected, both linearized systems ($OP1$ and $OP2$) are controllable and observable, despite not being stable. Stability could be achieved by implementing a controller.

1.10 Deduce the transfer functions and state space models related with the inner and outer dynamic loops defined by:

$$G_{n_x,\phi}(s) = \frac{\phi(s)}{n_x(s)} \qquad G_{\phi}, p_y(s) = \frac{p_y(s)}{\phi(s)}$$

$$G_{n_y,\theta}(s) = \frac{\theta(s)}{n_y(s)} \qquad G_{\theta}, p_x(s) = \frac{p_x(s)}{\theta(s)}$$

$$G_{n_y,\psi}(s) = \frac{\psi(s)}{n_z(s)} \qquad G_{T}, p_z(s) = \frac{p_z(s)}{T(s)}$$

The transfer functions defined above were obtained by converting the whole space state system into its corresponding transfer function. Since the output matrix for each linearized model is a 12-by-12 identity matrix (while also having 12 states), it was possible to directly extract most of the transfer functions intended, as they were defined in the transfer function variable of the linearized systems. As for the transfer functions that have dependencies on ϕ-p_y and θ-p_x, since they do not affect each other directly, it was necessary to find the corresponding transfer function for the numerator and denominator, and then divide them.

1.11 Discuss the closed loop stability of the modes defined by $G_T, p_z(s)$ and $G_\phi, p_y(s)$, for instance, considering root-locus arguments for a given control structure.

The closed loop stability of the modes defined by $G_T, p_z(s)$ and $G_\phi, p_y(s)$ was inferred through a root-locus analysis, which can be found below.

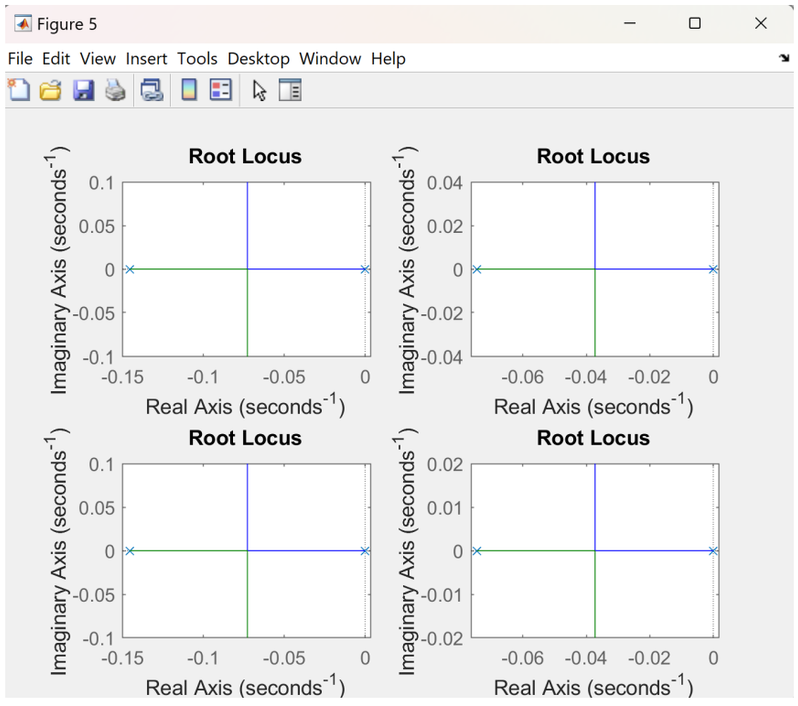

Figure 4: Root-Locus for $G_T, p_z(s)$ (left column) and $G_\phi, p_y(s)$ (right column) for both OP1 (top row) and OP2 (bottom row)

As expected from the stability analysis, all cases are not stable due to the existence of poles in 0.

3 Identification

This section focuses on the identification of the Crazyflie model and its comparison to the simulated model. In order to achieve this goal, a small flight dataset was acquired, analysed, and divided into 6 different datasets which consider movement in 3 axes (x, y, and z). These datasets were in regards to *pitch*, *roll*, *thrust*, and *positions* in the x, y, and z axes. Given the amount of noise present in the datasets, each was trimmed to the point where the signal to be considered contained mostly the movement of the drone.

The next step was to calculate the relation between certain inputs and outputs via identification of 3 transfer functions:

$$G_\theta, p_x(s) = \frac{p_x(s)}{\theta(s)} \tag{17}$$

$$G_\phi, p_y(s) = \frac{p_y(s)}{\phi(s)} \tag{18}$$

$$G_T, p_z(s) = \frac{p_z(s)}{T(s)} \tag{19}$$

Such identification was made possible by using Matlab's *systemIdentification* tool. Each transfer function was estimated with four different groups of poles and zeros, which allowed the identification of those that presented the best fits when compared to the original dataset.

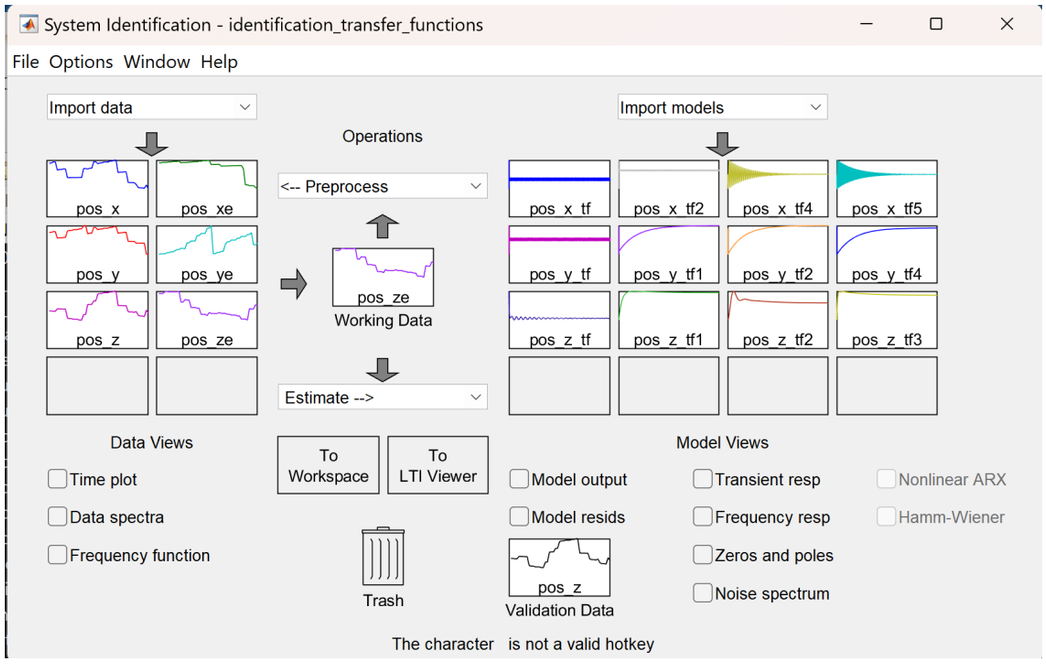

Figure 5: *systemIdentification* tool with the loaded datasets

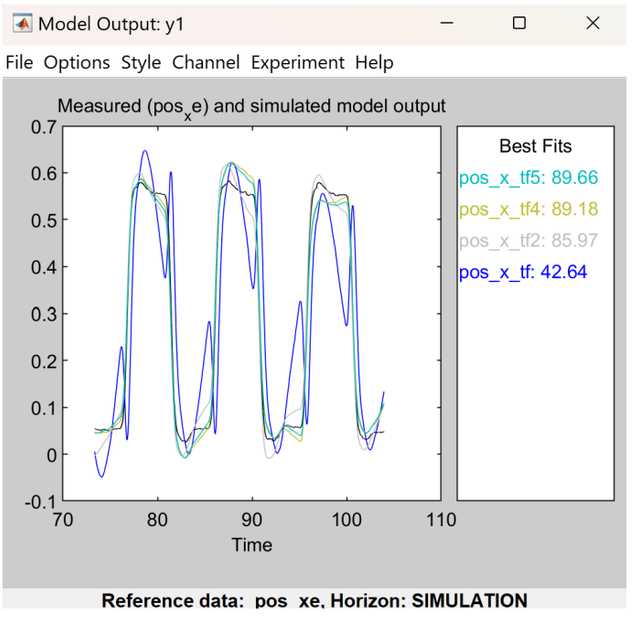

Figure 6: Best Fits for $G_\theta, p_x(s)$

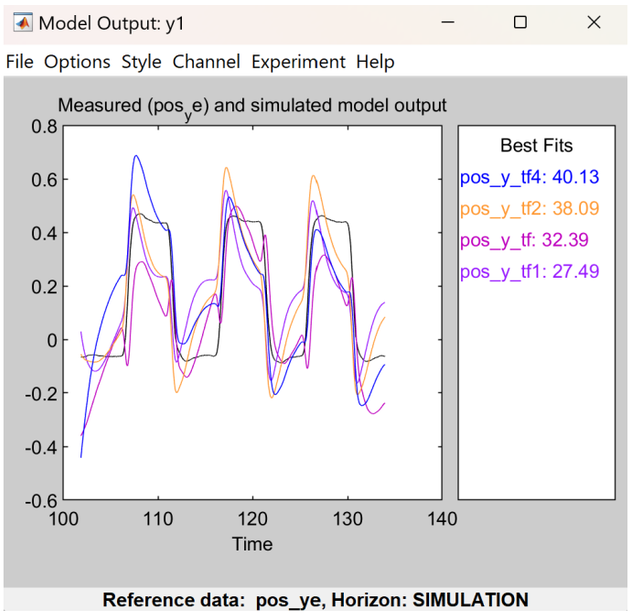

Figure 7: Best Fits for $G_\phi, p_y(s)$

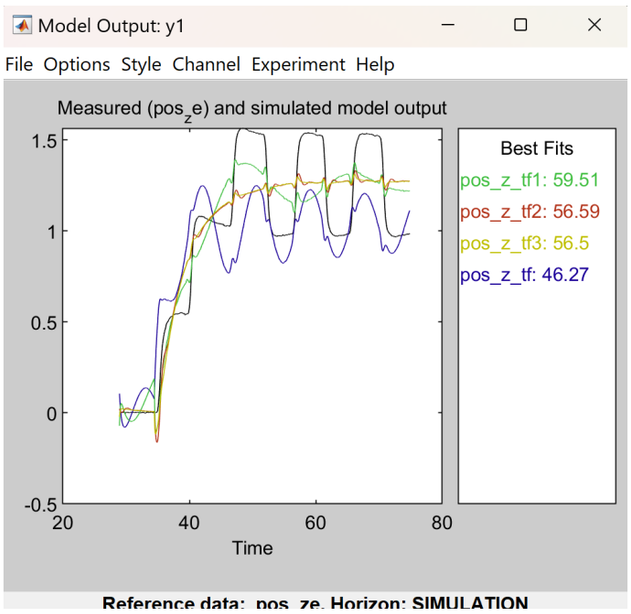

Figure 8: Best Fits for $G_T, p_z(s)$

The best fits identified were:

1. Position in the x-axis: 2 poles and 1 zero
2. Position in the y-axis: 2 poles and 1 zero
3. Position in the z-axis: 4 poles and 3 zeros