5.2. Quadrotor Controllers

5.2.1. Nonlinear Model Predictive Control

Nonlinear Model Predictive Control (NMPC) is a powerful control strategy that optimizes the performance of a nonlinear model subject to constraints over a finite prediction horizon. It combines system modeling, cost function formulation, optimization, and feedback to compute the optimal control inputs that drive the system toward desired behavior while satisfying constraints. Unlike linear control techniques, NMPC can handle systems with nonlinearity, constraints, and uncertainties, making it suitable for a wide range of applications.

In this study, NMPC is utilized to control the quadrotor motor speeds to follow a certain trajectory and to give the system a margin of noise immunity. NMPC is used in two cases; firstly by taking the 3 position parameters with their rates and the 3 angle parameters with their rates (total of 12 states) as output variables, and secondly by taking the 3 positions and the 3 angles without any rates (total of 6 states).

5.2.1.1. NMPC Setup and Tuning

NMPC require a different treatment than the linear MPC, NMPC requires the prediction model state function and it is advised and considered as best practice to give an analytical Jacobian formula for the prediction model as it significantly enhances the quality and accuracy of the responses and the applied control actions of the NMPC [1][2]. The prediction model state function and its Jacobian function can be specified and passed to the NMPC by their handle.

Before completing the rest of the NMPC setup parameters, the hovering speed is evaluated to be passed as the nominal control target to the NMPC. the hovering speed is evaluated by equating the quadrotor's total thrust with its weight and then solving for the motor hover speed.

$$T = k(u_1 + u_2 + u_3 + u_4) = mg$$

$$4u = 24 * 9.81$$

$$u = 58.86\, rpm^2$$

Back to the NMPC setup parameters, a prediction horizon of 10 to 20 steps is stable for NMPC to capture the hover full dynamics for small UAVs [1]. After several trials, a prediction of 18 steps with 2 steps of control action was found to be the most appropriate for the study of this

specific UAV. Since the hover speed was found to be $58.86\ rpm^2$, the control inputs constraints is set to be [0,70] to give the NMPC some flexibility in applying the appropriate action. As for the control action rates, they are set to [-2,2] to prevent aggressive motor responses that might harm the motors. For the cost function, the standard quadratic cost function is used for the NMPC as it is very suitable for reference tracking and noise reduction [1]. As for the output variables and manipulated variables weights, they are all kept to their default values of 1 for the output variables and 0.1 for the manipulated variables and their rates as well. A manipulated variables target is set to the hovering value of 58.86, and then the model was simulated to see NMPC in action using the ode45 solver and a fixed time step of 0.1s.

5.2.1.2. Results

For the first case where the parameter rates are not passed to the NMPC, the states' responses are delayed by almost 1.33s from the reference trajectory as shown in Fig.5.2.1.

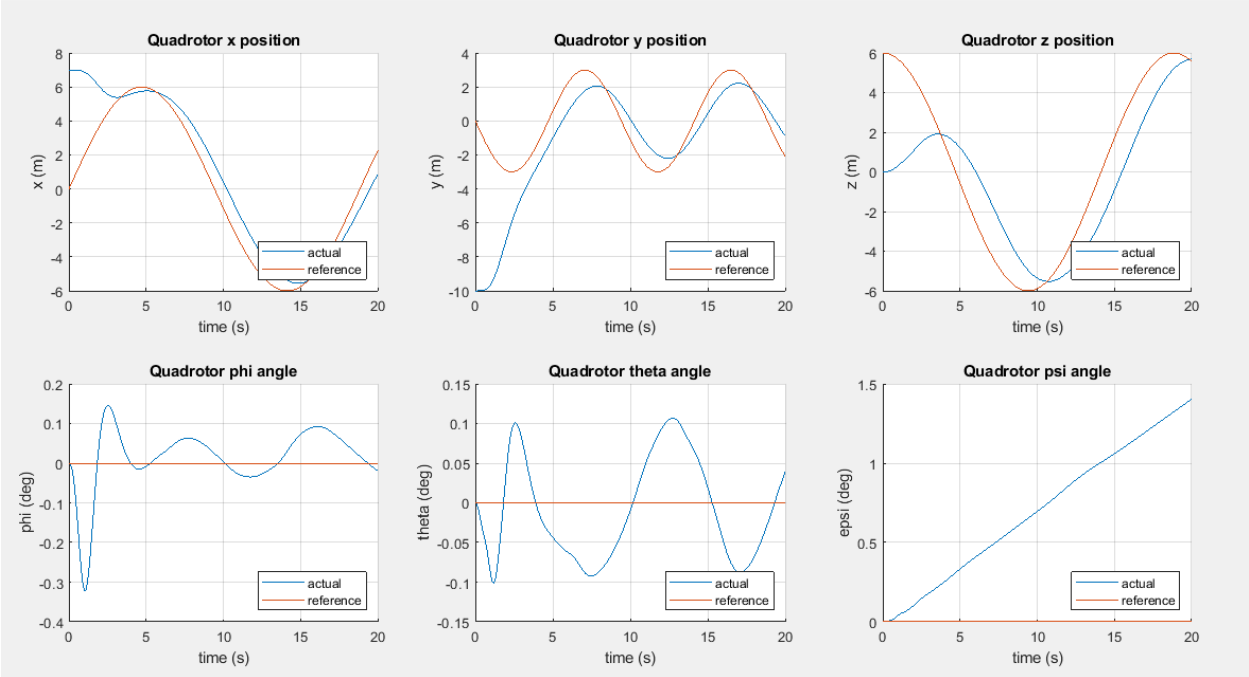

Fig.5.2.1. The states' responses for 6 states NMPC.

As for the control actions, the response is not observed to have any aggressive actions, and is kept to the minimum as shown in Fig.5.2.2.

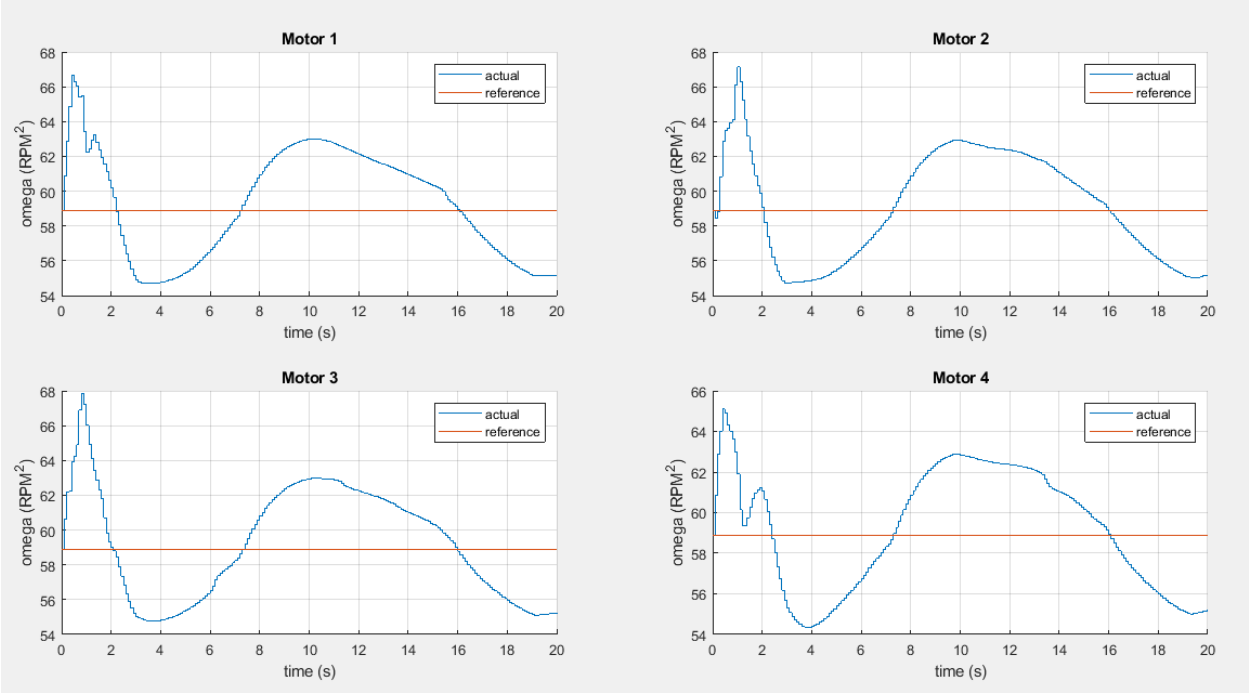

Fig. 5.2.2. The control actions responses for 6 states NMPC.

As for the second case where the parameters are utilized alongside their rates, the states' response is found to be significantly better but with an overshoot in the z position of the quadrotor of 19.32% and a minor overshoot in the y position of 4.67% (see Fig.5.2.3) due to the relatively high moments of inertia and mass of the whole VTOL.

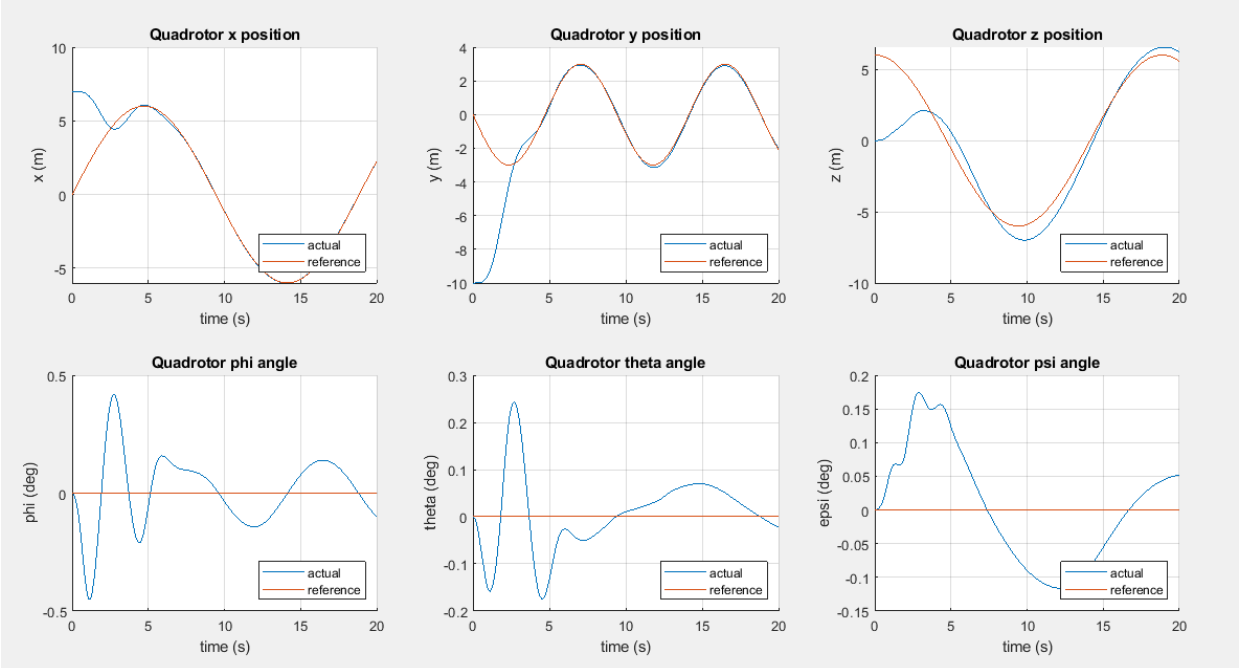

Fig. 5.2.3. The states' responses for 12 states NMPC.

Regarding the control actions, a slightly higher control action is applied than that of the previous case to prevent the 1.33s delay that is observed in the previous case, as indicated in fig.5.2.4.

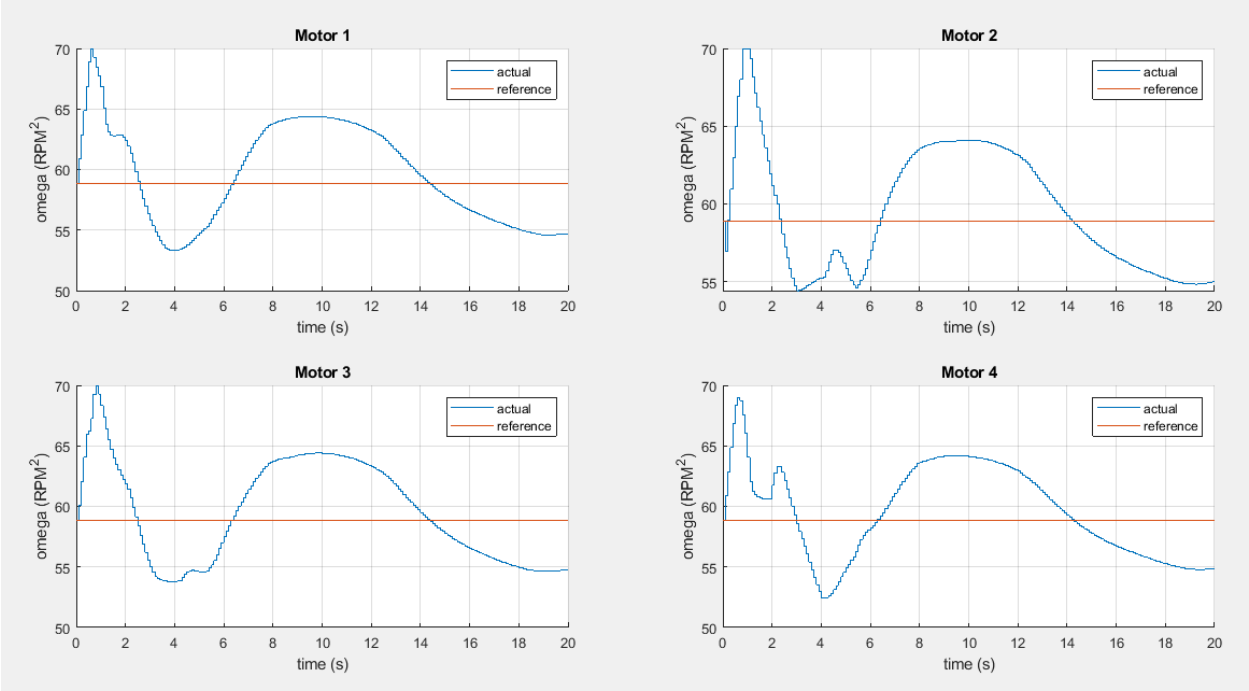

Fig. 5.2.4. The control actions responses for 12 states NMPC.

Overall, the NMPC gives significantly better control in the case of utilizing the model parameters alongside their rates and shows a good command of the quadrotor attitude response.

5.2.2. Proportional-Integral-Derivative Controller

PID control is a simple yet powerful method of controlling system processes. Provided the error signal as input ($e(t)$), which is the difference between the desired value and actual value of the process, the PID controller manipulates the error signal by evaluating three terms; proportional, integral, and derivative. The proportional term is calculated by multiplying the error signal by a proportional gain (k_P). As for the integral term, the error signal is integrated and multiplied by an integral gain (k_i). Finally, the derivative term is calculated by differentiating the error signal and multiplying it by a derivative gain (k_D). Thereafter, the terms are added together creating the control action $u(t)$ of the PID controller, as shown below:

$$\boldsymbol{u(t) = k_P * e(t) + k_i * \int_0^t e(\tau)\, d\tau + k_D * \frac{d}{dt} e(t)}$$

Each of these terms contributes to finding a suitable control action for the process. The proportional term is proportional to the error signal. It applies a corrective action based on the magnitude of the error. It helps the controller to respond quickly to quick variations in the error. But, on the other hand, it could lead to overshoot and steady-state errors. The integral term keeps track of the accumulated past errors, it provides a control action that gradually reduces the accumulated error, which helps in eliminating offset in controlled variables. Finally, the derivative term makes use of the error rate of change to anticipate future trends and dampen rapid changes and oscillations.

The following table summarizes the effect of PID parameters on signal performance characteristics; rise time, overshoot, settling time, and steady-state error.

Parameter	Rise Time	Overshoot	Settling Time	S-S Error
k_P	Decrease	Increase	Small Change	Decrease
k_i	Decrease	Increase	Increase	Decrease
k_D	Small Change	Decrease	Decrease	No change

Table 5.2.1. *PID Parameters Effect on Performance Characteristics (Nise, 2019).*

Performance Characteristics

Rise Time: Represents the time taken to reach around 90-95% of the desired setpoint. A smaller rise time indicates a fast response.

Overshoot: Represents the maximum deviation from the desired setpoint. It's mainly caused due to aggressive control actions.

Settling Time: Represents the time taken by the signal to reach and settle within an acceptable range of the setpoint and remains there indefinitely. It provides an indication of how quickly the system reaches a stable state after a change in the setpoint or disturbance.

Steady-state Error: Represents the deviation that remains after the response reaches its steady state. It provides a measure of the difference between the desired setpoint and the current final value of the controlled variable.

The following figure illustrates the performance characteristics for better visualization.

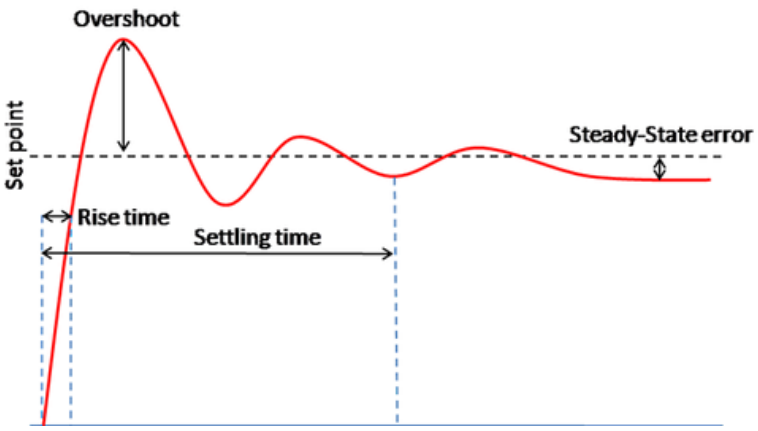

Figure 5.2.5. Performance Characteristics (WPI , 2021).

5.2.2.1 Cascaded PID Controller Architecture

A quadrotor is a highly nonlinear and underactuated system, which means that it has fewer control inputs (actuators) than the degrees of freedom that should be controlled. Namely, a quadrotor has only 4 actuators, meanwhile, it has 6 degrees of freedom, both translational and rotational.

A cascaded PID controller consists of several PID controllers that are arranged hierarchically, where each controller is responsible for controlling one of the system states. This arrangement in essence results in two control loops; outer control loop (position controller) and inner control loop (attitude controller).

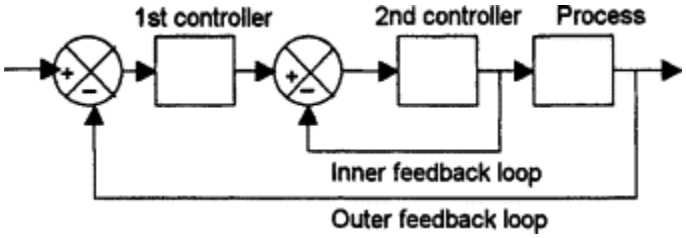

Figure 5.2.6. Cascaded PID Controller (Bolton, 2021).

Due to the decoupled nature of the system dynamics, the system states could be arranged such that the secondary process variables, that influence the primary process variables, are placed in the outer control loop. On the other hand, the primary process variables are placed in the inner control loop. In that sense, all the 6 degrees of freedom of the quadrotor are all controlled with only 4 actuator inputs.

The outer control loop is responsible for controlling the position (x & y) and altitude (z) of the quadrotor. The input to the states is the error signal computed as the difference between the

desired value and the current value (provided by the state estimator). The output of the outer loop serves as a setpoint to the inner controller in addition to the thrust command.

As for the inner control loop, it is responsible for controlling the quadrotor orientation (attitude) in terms of roll, pitch, and yaw. The output of the inner control loop serves as actuator commands.

Simulink Model

The closed control loop is shown below, where the cascaded PID controller is implemented resulting in two control loops; inner (attitude) and outer (position).

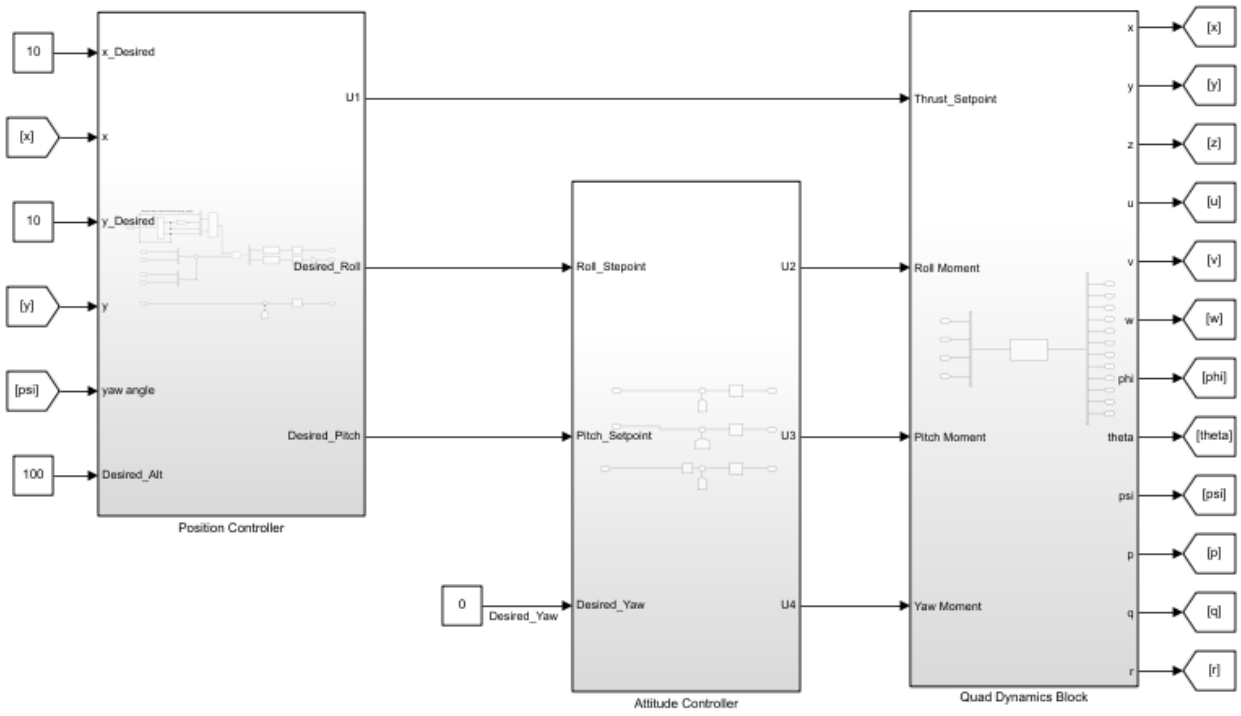

Figure 5.2.7. *Simulink PID Controller Block.*

Simulink Position Controller

The structure of the position controller is shown below, in which 3 PID controllers are implemented for each position state. In addition, the x & y error states are multiplied by a rotation matrix represented by the yaw angle. The rotation matrix works on obtaining the correct

representation of the position vector (x & y) in terms of the heading (yaw) angle of the quadrotor. Such that it's continually updated. In addition, a separate PID controller controls the altitude (z) of the quadrotor, which outputs the thrust setpoint (U1).

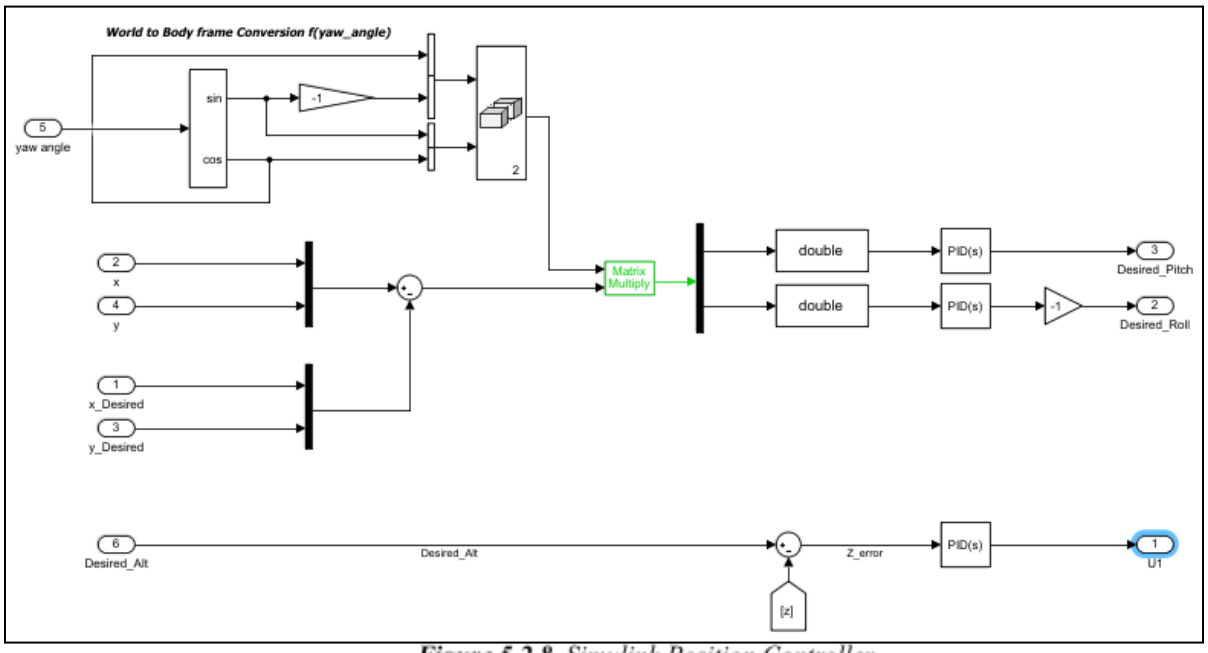

Figure 5.2.8. *Simulink Position Controller.*

Simulink Attitude Controller

The block structure is shown below, where each orientation state (roll, pitch, and yaw) is controlled by a PID controller. The input to both roll and pitch is the output setpoint of the position controller. The output of the inner loop controller is the actuator inputs (Moments) U2, U3, and U4.

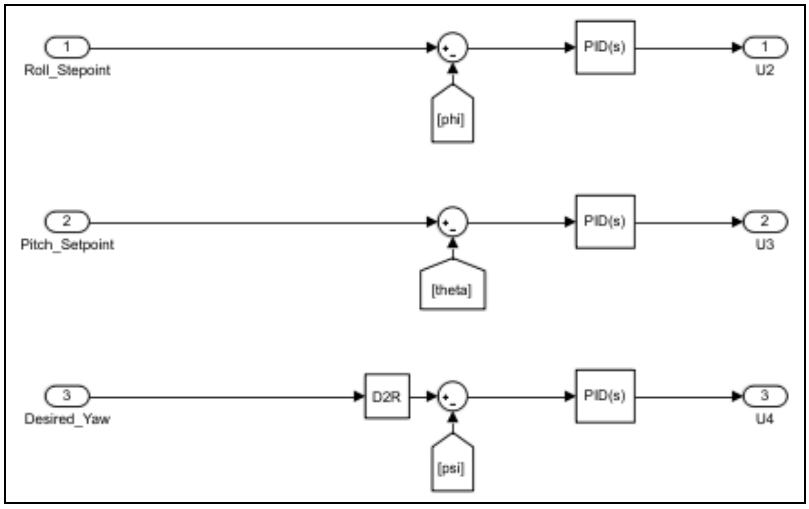

Figure 5.2.9. *Simulink Attitude Controller.*

5.2.2.2 Controller Tuning

There exist several methods that could be used to tune PID controllers, some of these methods are presented below:

Manual Tuning: this approach implies manually tweaking the PID controller gains until an acceptable response is obtained. However, in the case of a quadcopter, there are 6 PID controllers that should be adjusted. Accordingly, such an approach is deemed inefficient and impractical.

Heuristics & Optimization Algorithms: methods like Ziegler-Nichols and Cohen-Coon make use of the system response data and provide gain estimates using formulas. Such methods could be useful in achieving an acceptable response but not yet optimum. However, they could provide a good starting point or an initial guess for PID control parameters.

On the other hand, optimization algorithms like genetic algorithms and Particle Swarm Optimization can find the optimum or near-optimal PID gains. Their main working mechanism is iteratively minimizing a performance index (cost function) specified to the algorithm. While such an approach is more computationally intensive, it can effectively handle a complex system with nonlinear dynamics, as in the current case of a quadrotor.

Proposed Approach

In this work, Ziegler-Nichols Method was used to obtain a good initial guess for PID parameters. Thereafter, the gains were manually tweaked in order to test the response under small perturbations in PID parameters. In such a sense, an estimate for the search area was identified,

which was used together with the initial guess as an input to optimization algorithms in order to fine-tune the controller and reach the best performance possible.

Ziegler-Nichols Method

The method is applied to tune the PID controller in the following manner (Meshram & Kanojiya, 2012):

1. The integral gain is set to zero.
2. The proportional gain is increased incrementally while examining the system response until sustained oscillations are obtained.
3. Set the current proportional gain to K_u.
4. The period of oscillation, time from peak to peak, is measured and set to T_u.
5. The gains are then estimated using the formulas shown in the table below.

Controller	K_p	T_i	T_d	K_i	K_D
PID	$0.6 * K_u$	$T_u/2$	$T_u/8$	$1.2 * K_u/T_u$	$0.075 * K_u T_u$
P	$0.5 * K_u$	-	-	-	-
PI	$0.45 * K_u$	$T_u/1.2$	-	$0.54 * K_u/T_u$	-
PD	$0.8 * K_u$	-	$T_u/8$	-	$0.1 * K_u T_u$

Table 5.2.2. *PID ZN Controller Gain Estimation (Meshram & Kanojiya, 2012).*

In addition to the estimated gains, the model response could be fine-tuned manually as an attempt to improve the performance after the initial guess obtained from the ZN method.

Optimization Algorithms

PID gain optimization algorithms make use of mathematical optimization techniques to find the optimal values for PID parameters. Algorithms achieve this by iteratively searching for the combination of parameters that minimizes the performance index or cost function. This is achieved by relying on feedback from the performance index to update the parameters (Ahmmed et al., 2020). Therefore, they can help in improving the control performance and robustness of the system.

The choice of which optimization algorithm to use is dependent on the system characteristics. In the case of a quadrotor, the system dynamics are highly nonlinear. In addition, the optimization problem involves variables with high-dimensional search space. Moreover, the fitness (cost) function is non-differentiable. Therefore, the gradient is not available for use. Finally, the optimization problem is considered multimodal, meaning that several optimal or near-optimal solutions exist depending on the chosen search space.

All of this narrowed down the options to a few algorithms, among which Genetic Algorithm was suitable for our case.

Genetic Algorithm

The algorithm is inspired by the process of natural selection, in which the algorithm operates on a set of potential solutions, where individuals undergo genetic operations to reproduce better generations (Ahmmed et al., 2020). Reproduction is meant to produce new generations that carry better characteristics contributing to a better fitness value for the objective function. In that sense, the algorithm can help find the optimum PID parameters that minimize the performance index.

Algorithm Setup

Several vital factors should be considered carefully to achieve convergence with suitable computational time while achieving acceptable performance (Mathworks, 2023).

1. **Search Space:** while a large search space can help the algorithm explore further solutions, potentially improving the performance, it can come at the expense of high computational effort. Accordingly, the ZN method was initially used to narrow down to a viable search space.
2. **Population Size:** large population size can also help in better exploration of the search space. However, if it's too high, it may slow down the simulation due to computational overhead. As more individuals in this case will need to be evaluated and selected in each generation (iteration). Therefore, it's recommended to start from 50 to 200 initially and increase incrementally while monitoring performance.
3. **Max Number of Generations:** it determines how long the algorithm will run before termination. Setting a high number of generations can potentially improve the result, as the algorithm will continue searching for solutions. However, it's also computational

overhead. It's also recommended to start with 100 to 1000 and increase incrementally as well.

4. **Performance Index:** the choice of the fitness (cost) function contributes greatly to both the convergence of the algorithm and performance improvement. In this case, two cost functions are proposed; the least squares method and integral time absolute error.

Simulation Results

All the controllers were tuned as PD controllers, meaning that the integral gain was set to zero in all controllers. This was done for the following reasons:

1. **Simplicity:** removing the integral gain will reduce the problem size in terms of the number of variables to be tuned.
2. **Model Inaccuracies:** the mathematical model of the quadrotor may fail to capture the true dynamics exactly of the quadrotor in real-life operation. As the measurements are subject to sensor noises. The integral term is very sensitive to such uncertainties and may lead to instability.

Ziegler-Nichols Method

The method was applied to the model states while maintaining a zero desired yaw angle.

Inner Loop Controller		
State	K_u	T_u
Phi	0.1875	6.93 s
Theta	0.075	7.33 s
psi	0.625	1.6 s

Table 5.2.3. *ZN Results for Inner Loop.*

Outer Loop Controller		
State	K_u	T_u
X	$2.5 * 10^{-3}$	16 s
Y	$3.75 * 10^{-3}$	13.3 s
Z	18.75	3.73 s

Table 5.2.4. *ZN Results for Outer Loop.*

These results were then used to compute the proportional and derivative gains for all the controllers. In addition, some states were manually tuned further in an attempt to improve the response.

5.2.2.2.1 Ziegler-Nichols Results

Testing response of quadrotor given a setpoint in x and y directions (10, 10), starting initially at (0, 0), while altitude was given a setpoint at 110 m.

Outer Loop Response

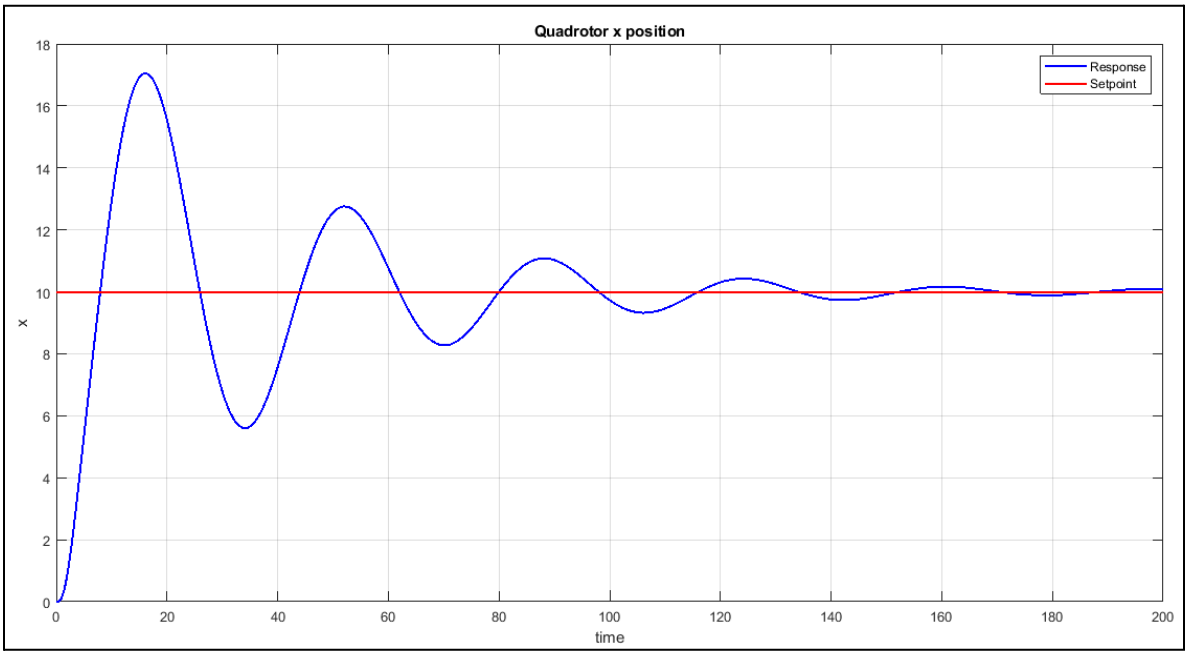

Figure 5.2.10. *X position response (ZN).*

Rise Time	5.1897
Settling Time	146.5576
% Overshoot	70.5037

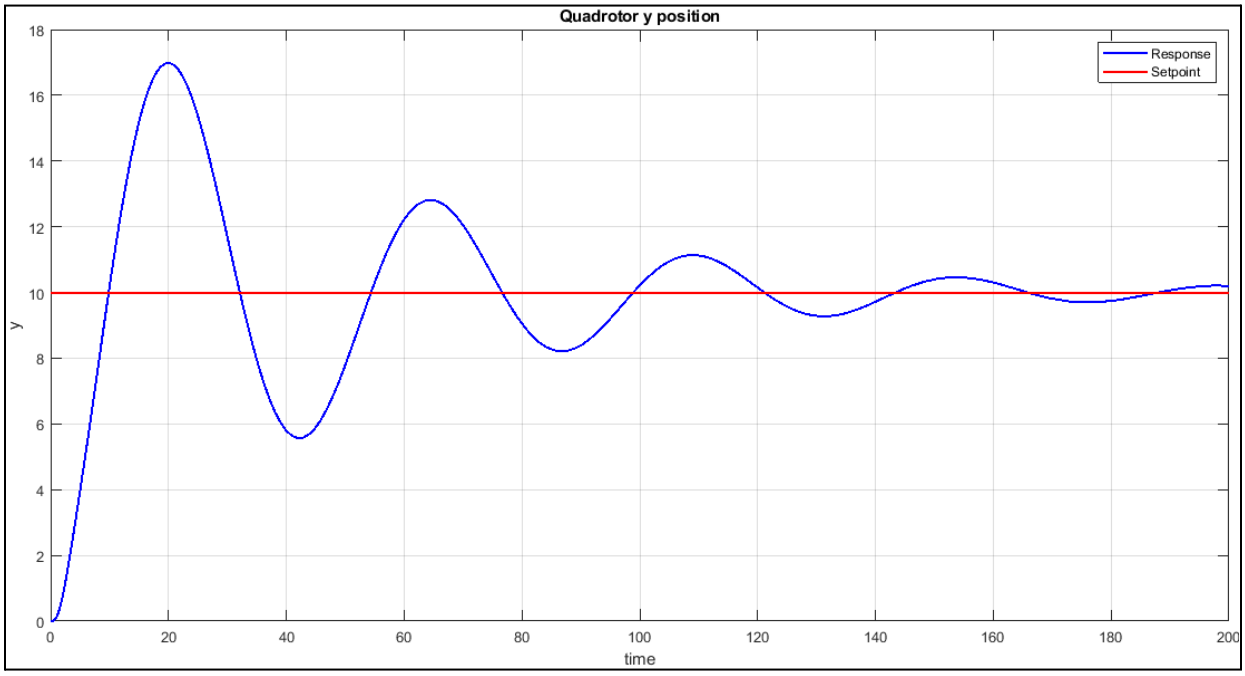

Figure 5.2.11. *Y position response (ZN).*

Rise Time	6.7438
Settling Time	202.47
% Overshoot	69.777

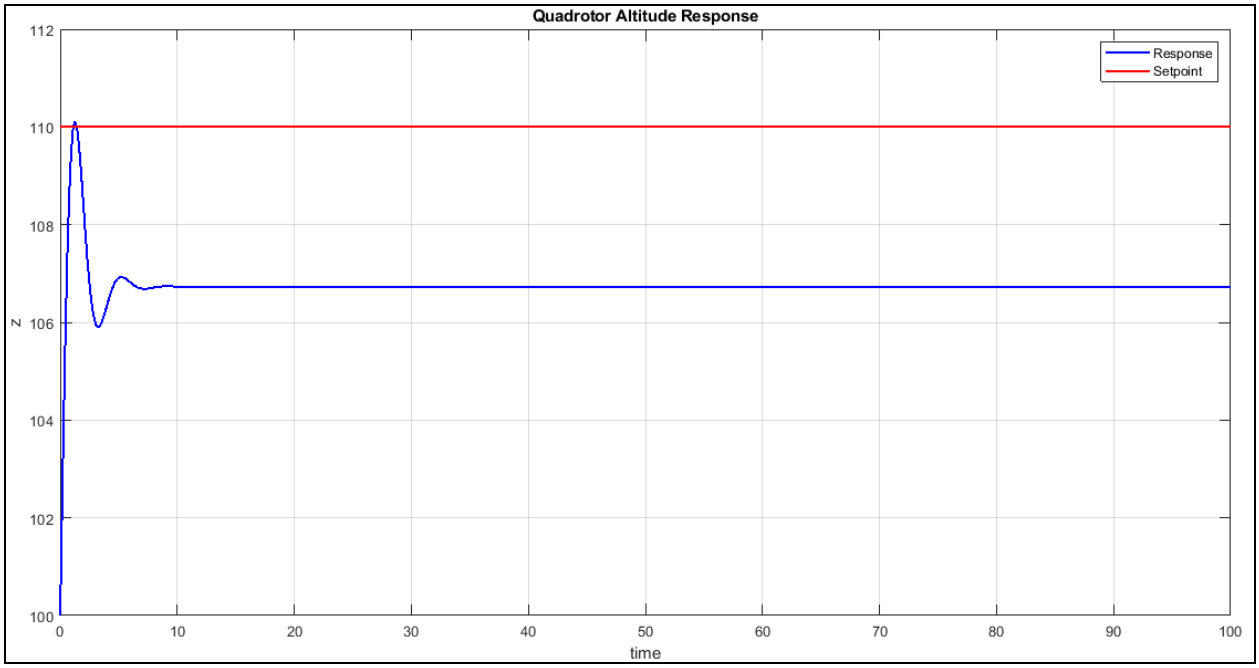

Figure 5.2.12. *Altitude response (ZN).*

Rise Time	0.7711
Settling Time	NAN
% Overshoot	0.0991

Inner Loop Response

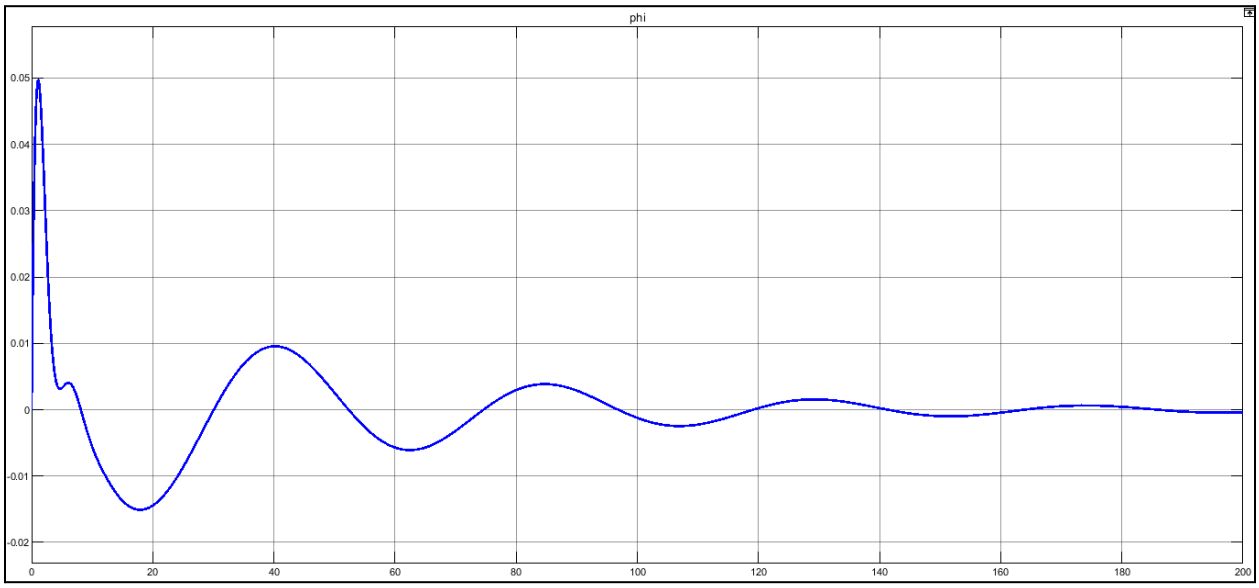

Figure 5.2.13. *Phi Angle Response (ZN).*

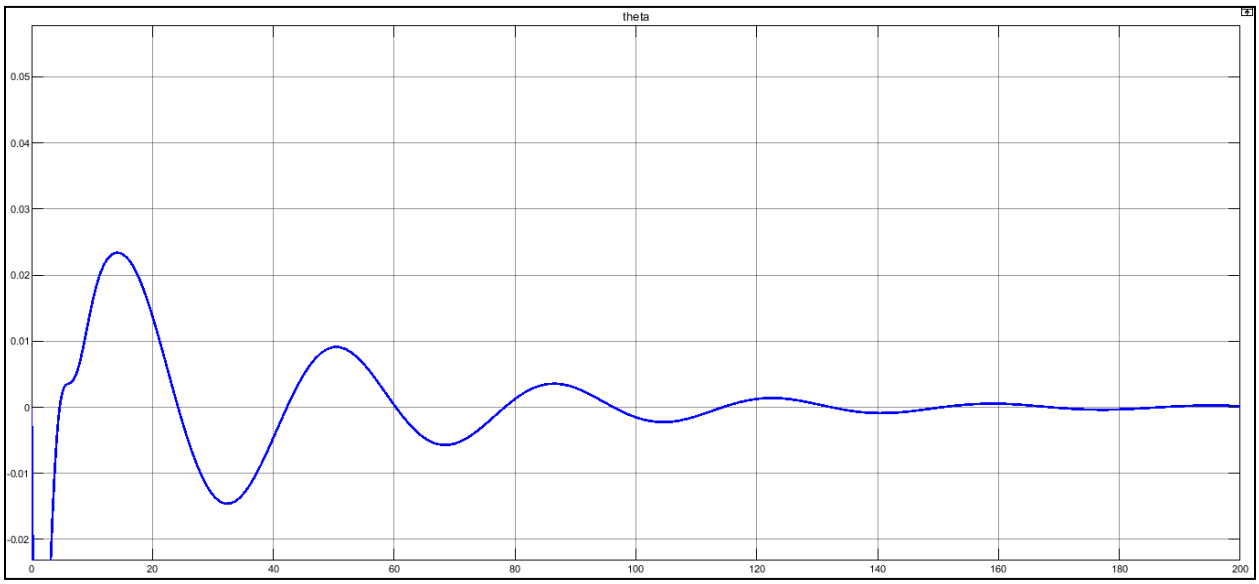

Figure 5.2.14. *Theta Angle Response (ZN).*

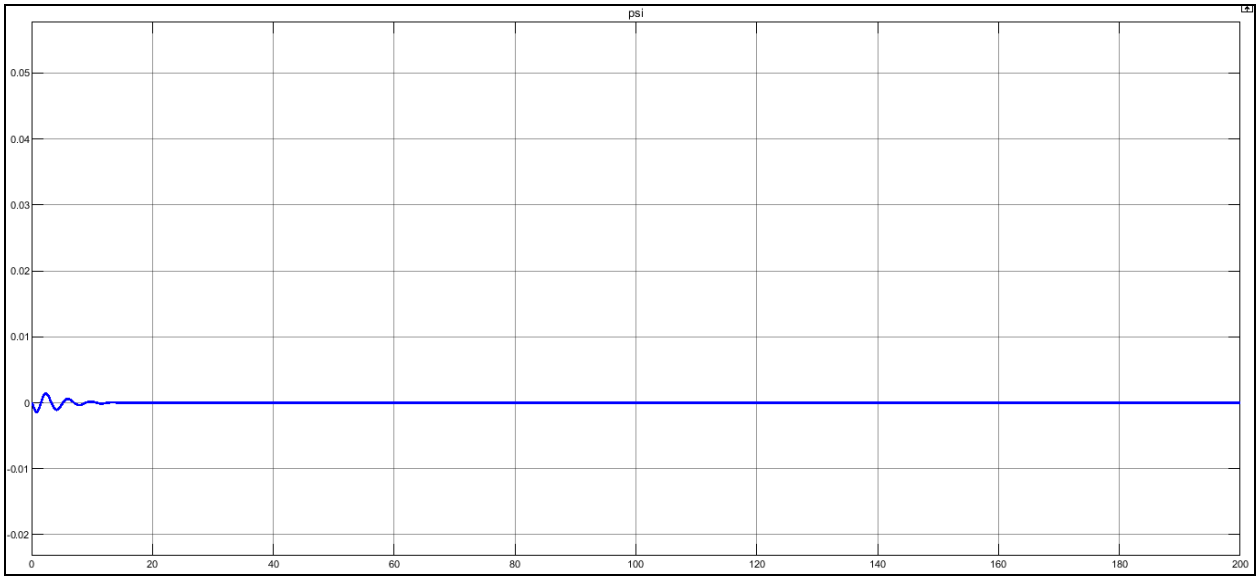

Figure 5.2.15. *Psi Angle Response (ZN).*

Disturbance Rejection Response

The quadrotor controller was tested to damp out 2 degrees of disturbance in both roll & pitch angles.

Outer Loop Response

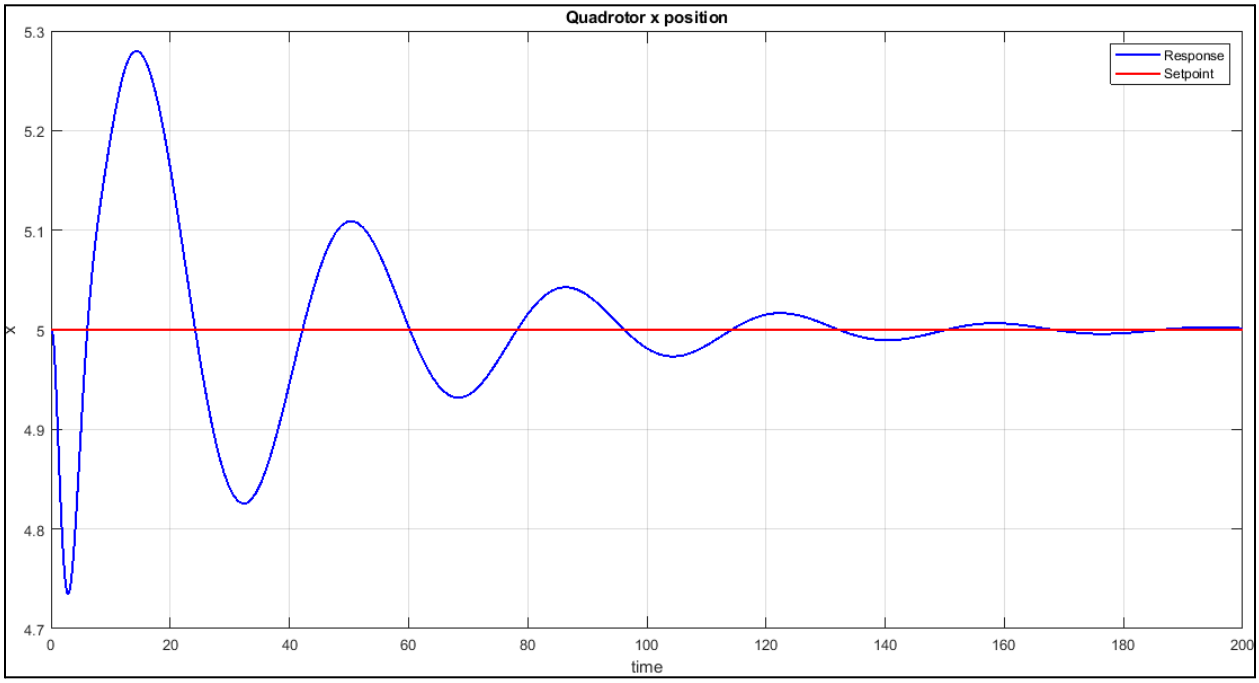

Figure 5.2.16. *X Disturbance Rejection (ZN).*

Settling Time	161.593
% Overshoot	5.5958

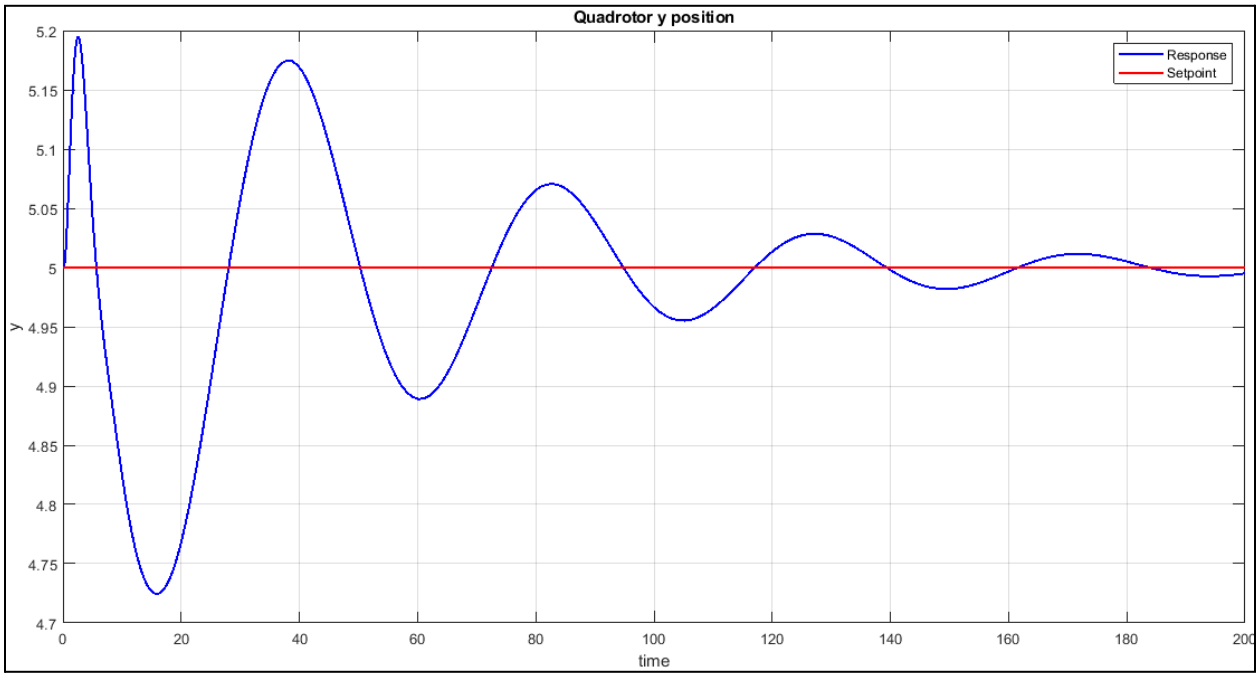

Figure 5.2.17. Y Disturbance Rejection (ZN).

Settling Time	199.1681 s
% Overshoot	3.8935

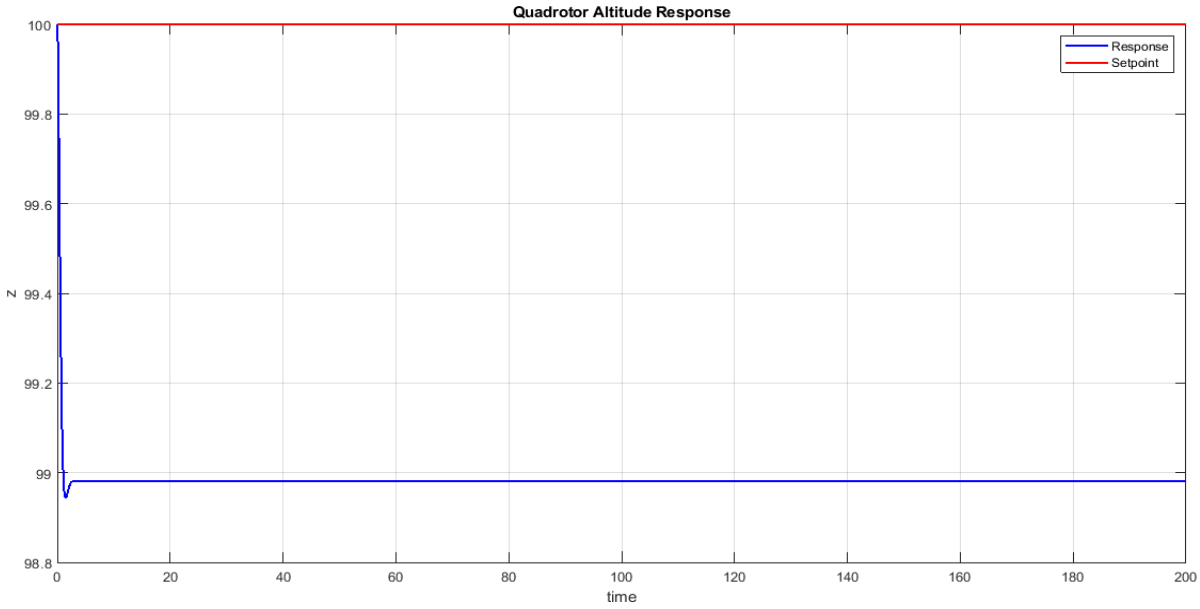

Figure 5.2.18. *Z Disturbance Rejection (ZN).*

Inner Loop Response

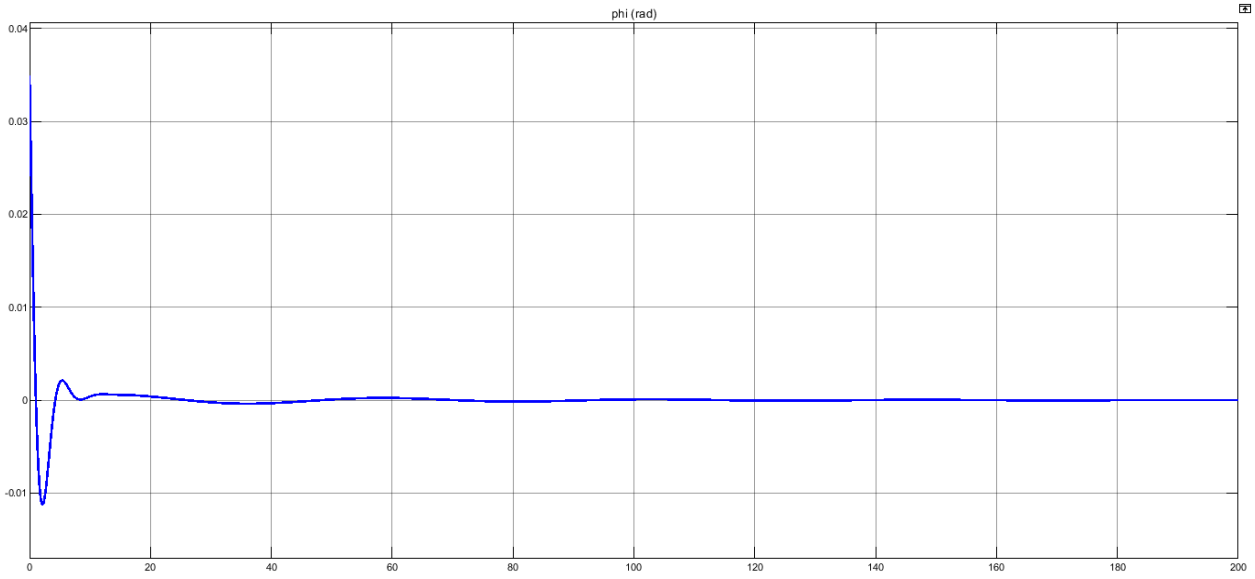

Figure 5.2.19. *Phi Angle Response (ZN).*

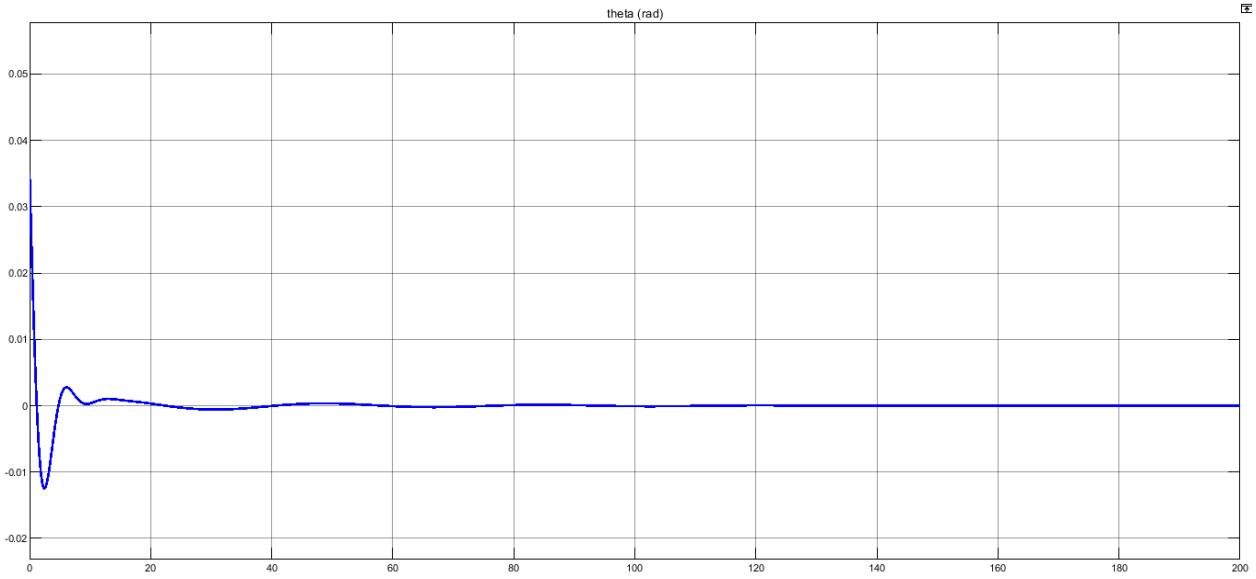

Figure 5.2.20. *Theta Angle Response (ZN).*

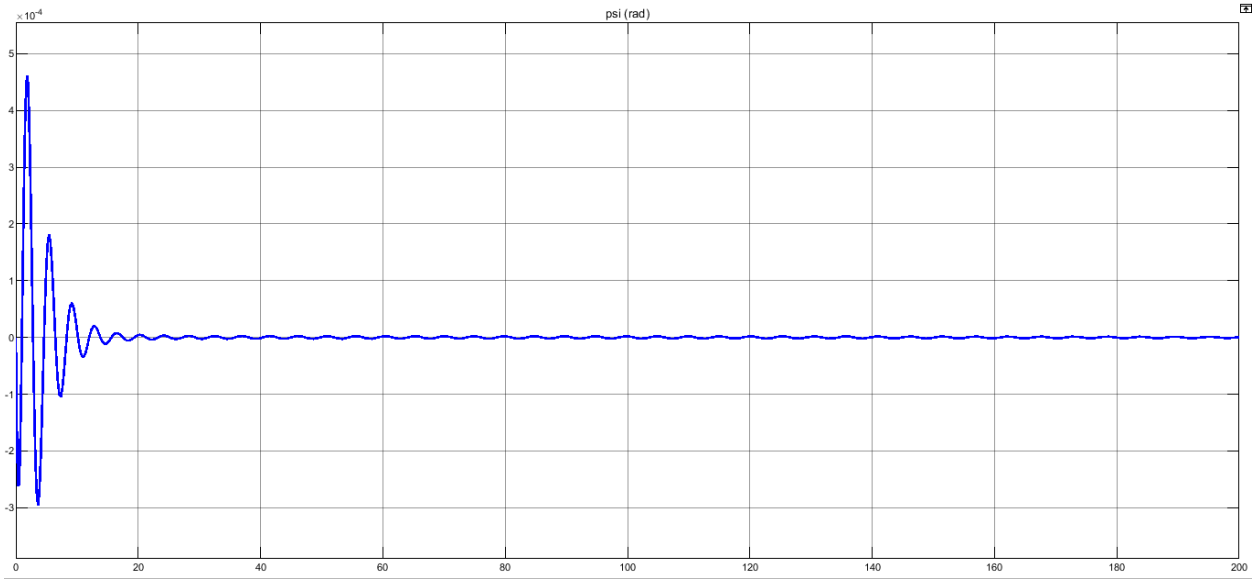

Figure 5.2.21. *Psi Angle Response (ZN).*

Discussion

As for the step response, high levels of overshoot and long settling times were spotted. However, the controller managed to settle eventually around the setpoint. As for disturbance rejection, the overshoot was acceptable, as it oscillates with small values across the setpoint. However, the

settling time problem persists in this case as well.

5.2.2.2.2 Genetic Algorithm Results

The quadrotor was given setpoints to move 10 m in x, y, and z directions.

Outer Loop Response

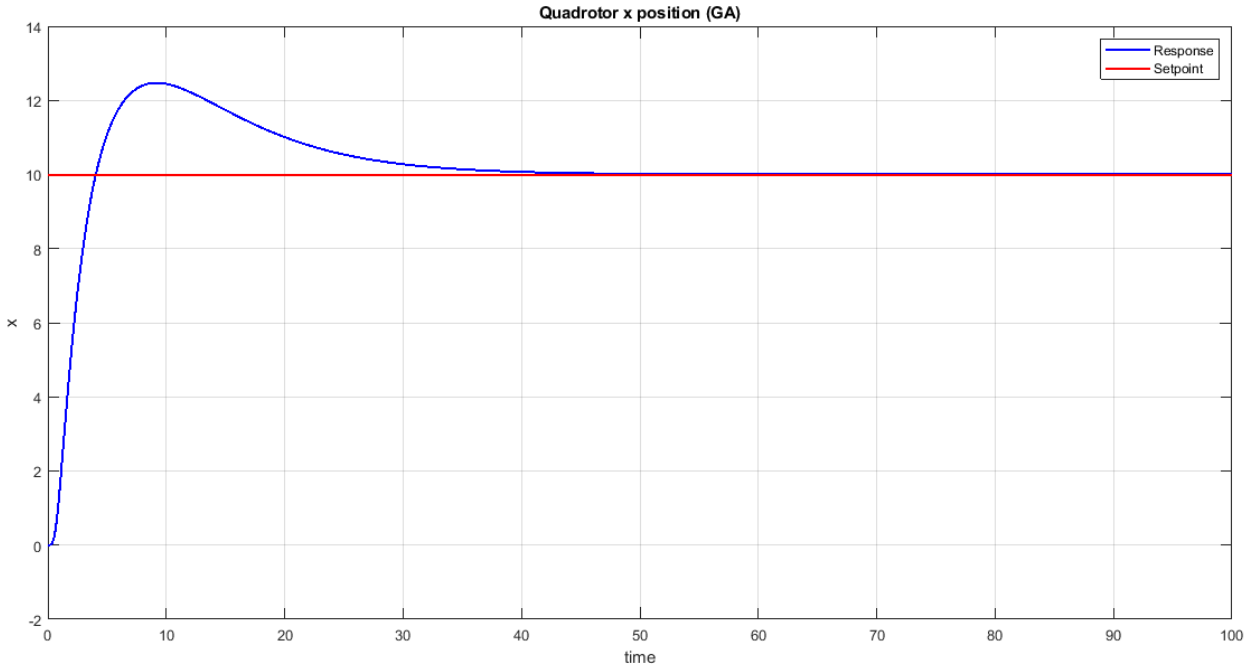

Figure 5.2.22. *X Position Response (GA).*

Performance	Value	Improvement
Rise Time	2.5844	50.2%
Settling Time	32.4265	77.87%
% Overshoot	24.7	64.97%

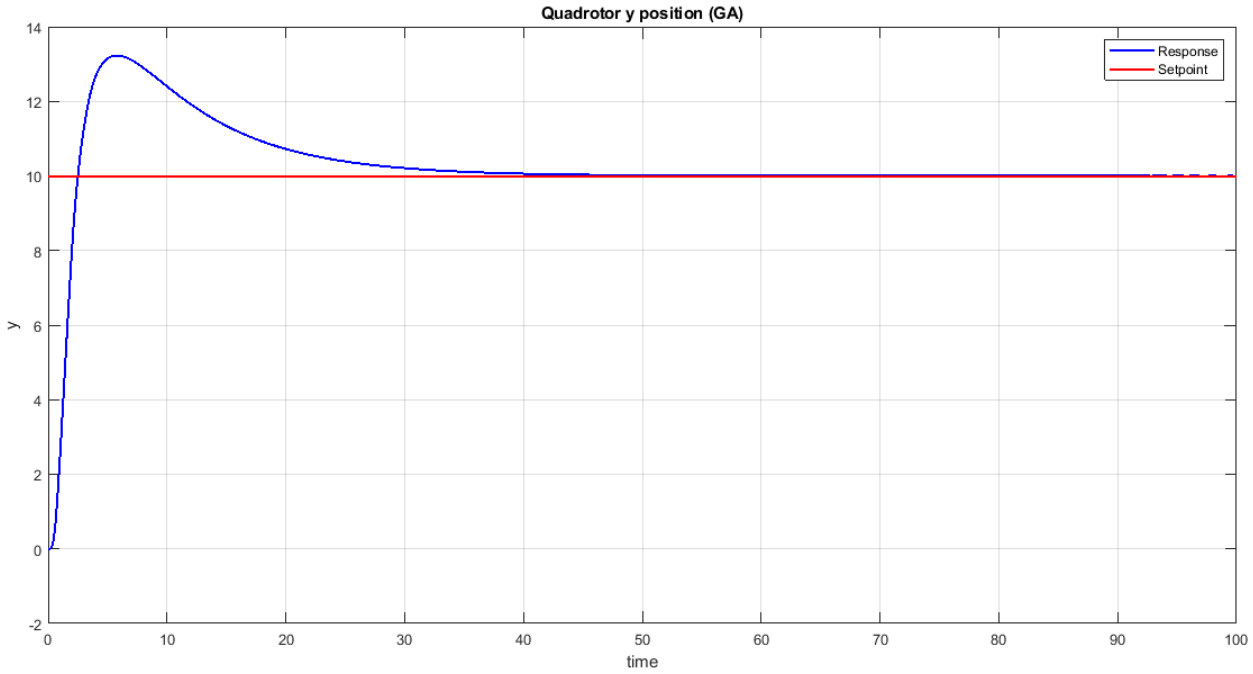

Figure 5.2.23. Y Position Response (GA).

Performance	Value	Improvement
Rise Time	1.5242	77.39%
Settling Time	30.3737	84.99%
% Overshoot	32.2996	53.71%

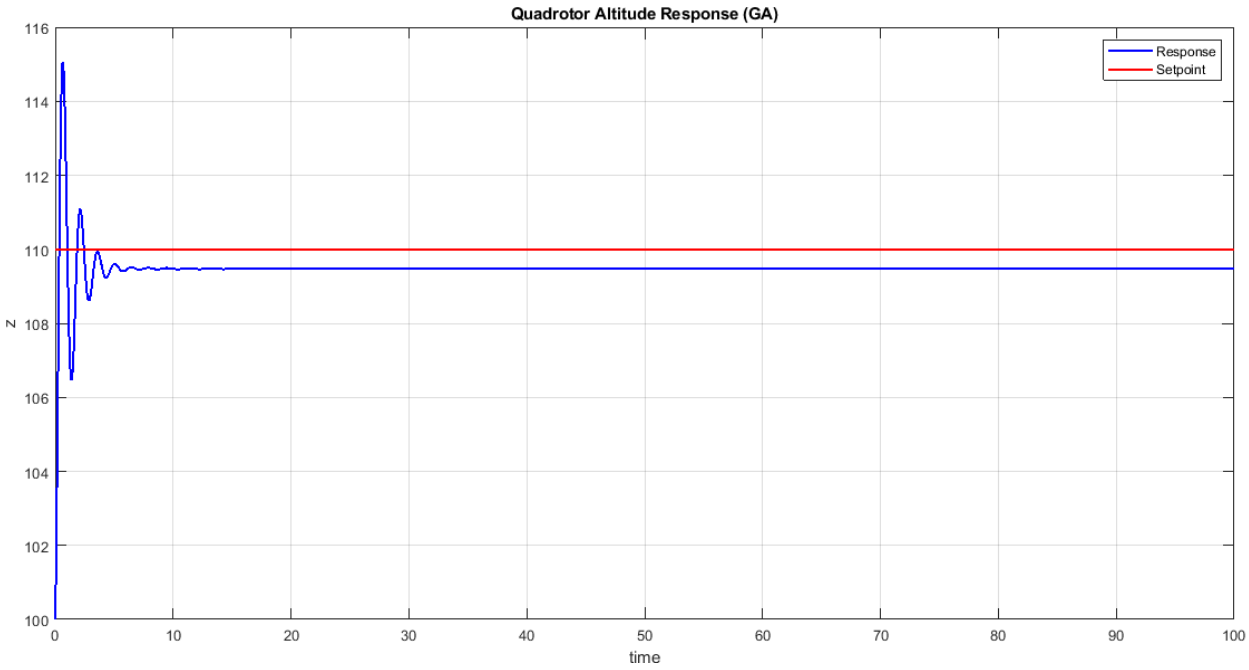

Figure 5.2.24. *Altitude Step Response (GA).*

Performance	Value	Improvement
Rise Time	0.2541	67.04%
Settling Time	NAN	NAN
% Overshoot	4.591	-45%

Inner Loop Response

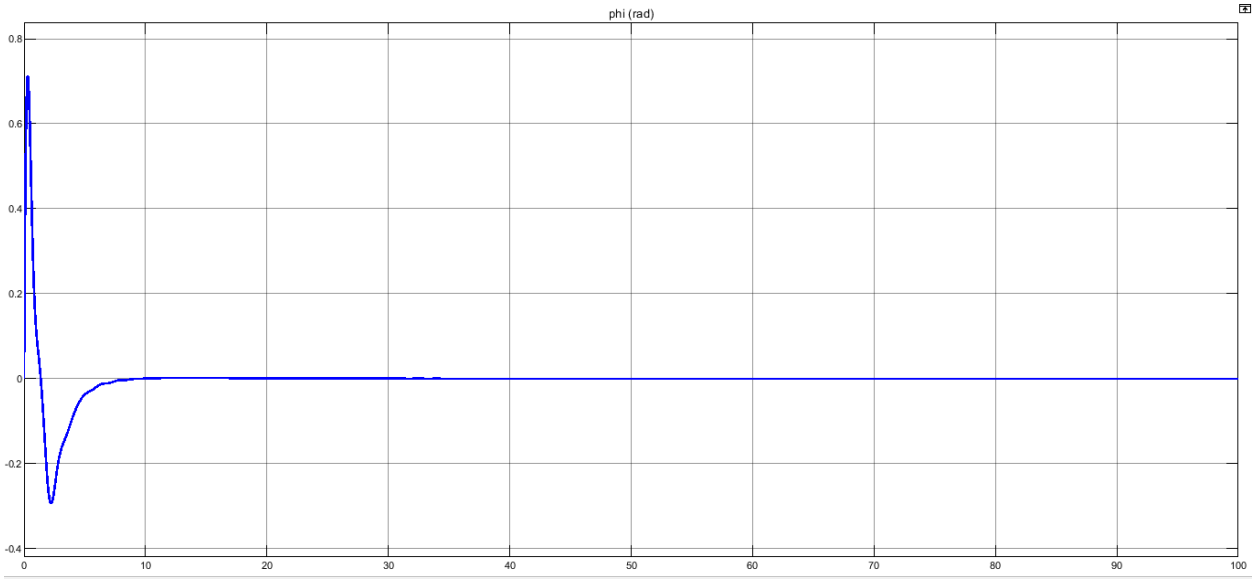

Figure 5.2.25. *Phi Angle Response (GA).*

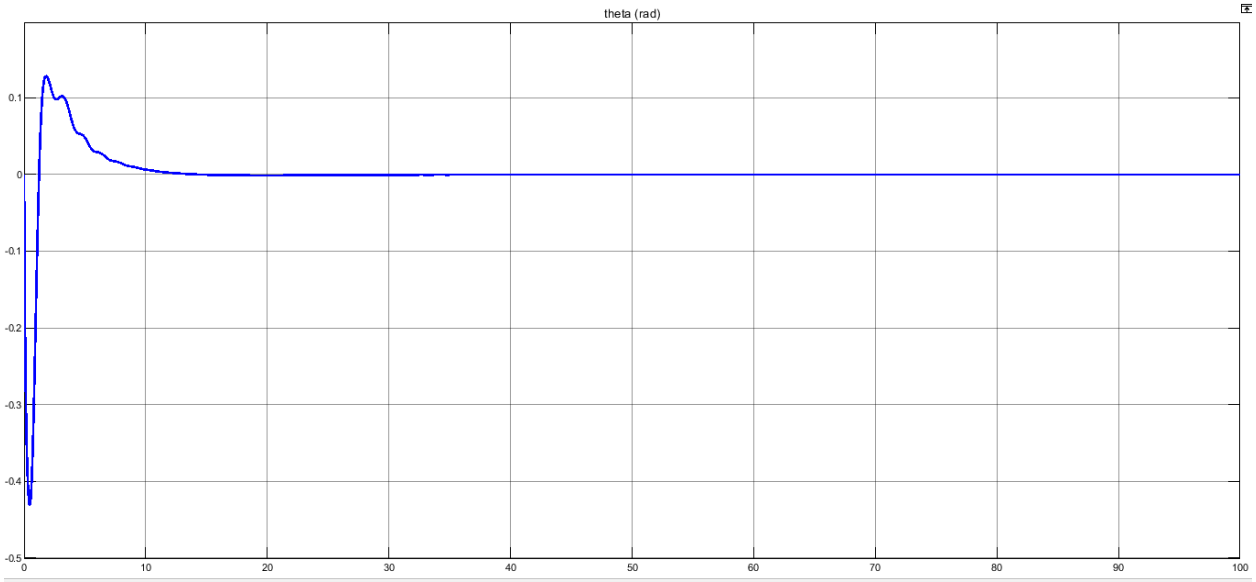

Figure 5.2.26. *Theta Angle Response (GA).*

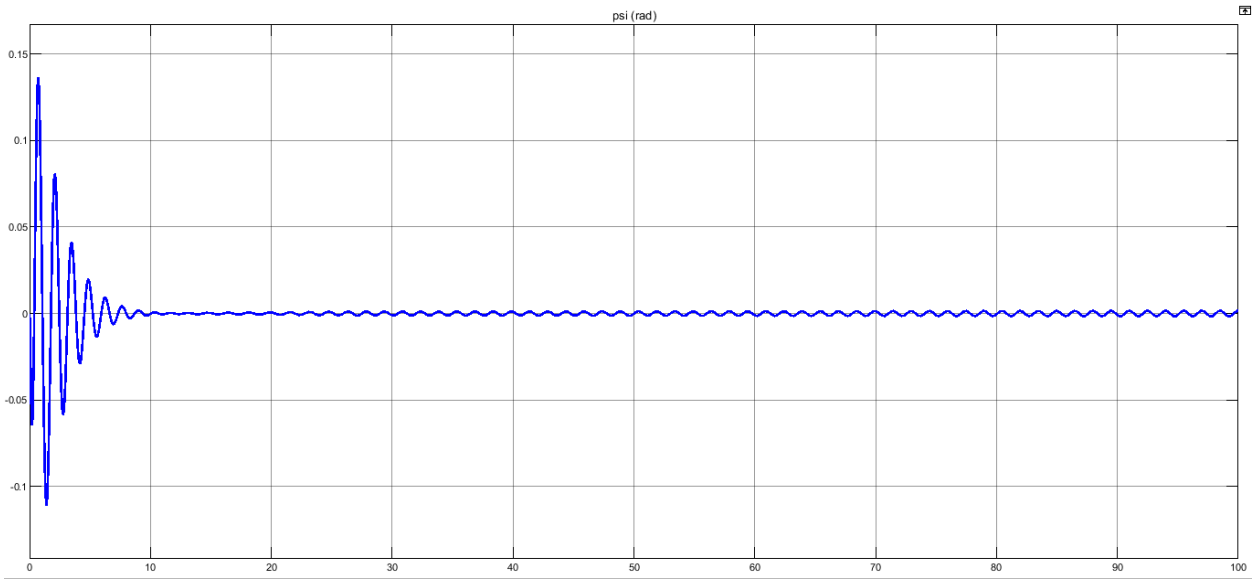

Figure 5.2.27. *Psi Angle Response (GA).*

Disturbance Rejection

The quadrotor, initially set at (5, 5, 100), was given a disturbance of 2 degrees in roll & pitch.

Outer Loop Response

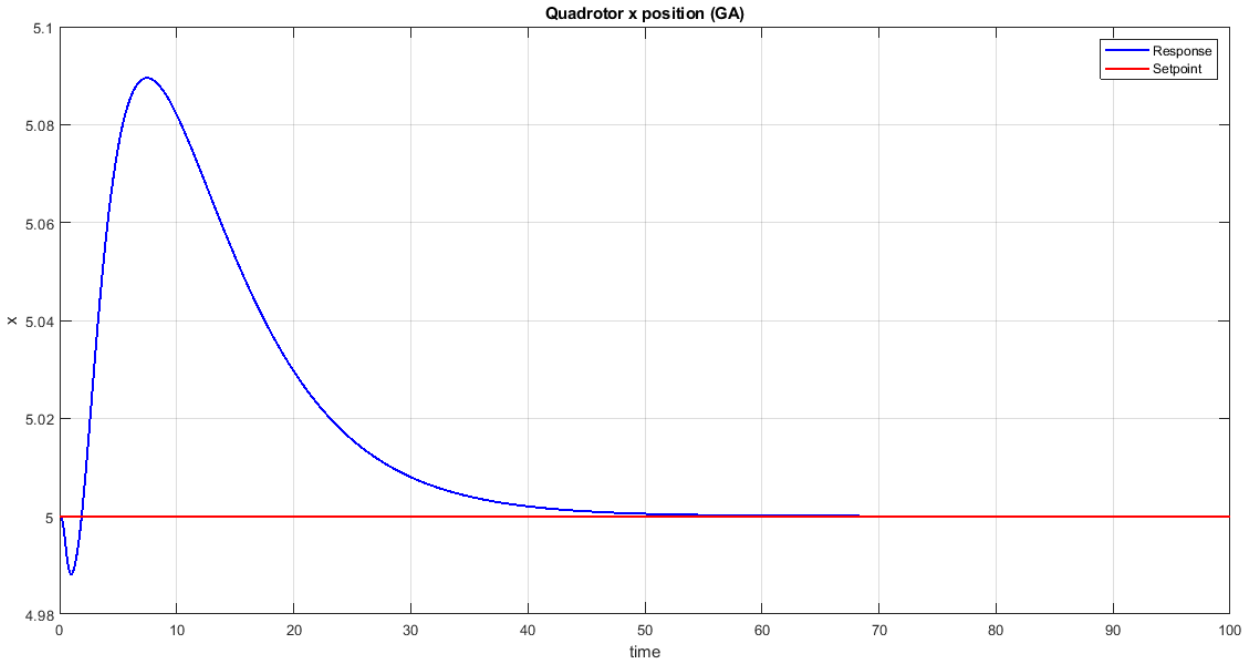

Figure 5.2.28. *X Position Response (GA).*

Performance	Value	Improvement
Settling Time	40.8368	74.73%
% Overshoot	1.79	68%

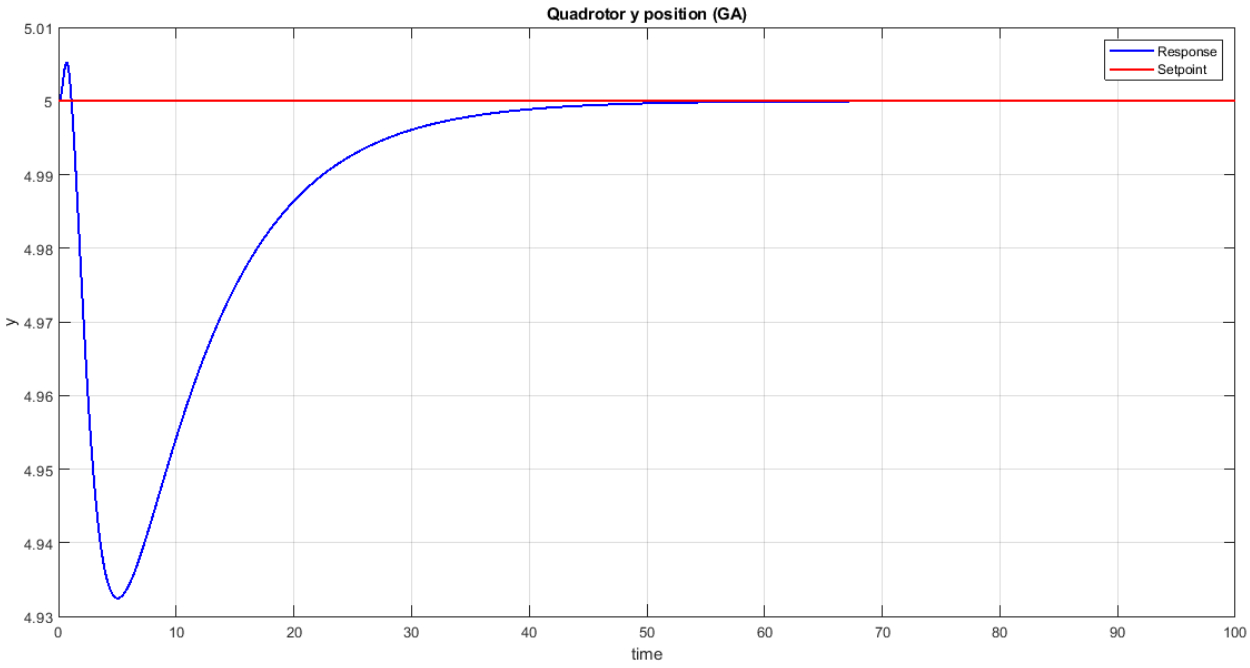

Figure 5.2.29. *Y Position Response (GA).*

Performance	Value	Improvement
Settling Time	38.5587	80.6%
% Overshoot	0.1048	97.3%

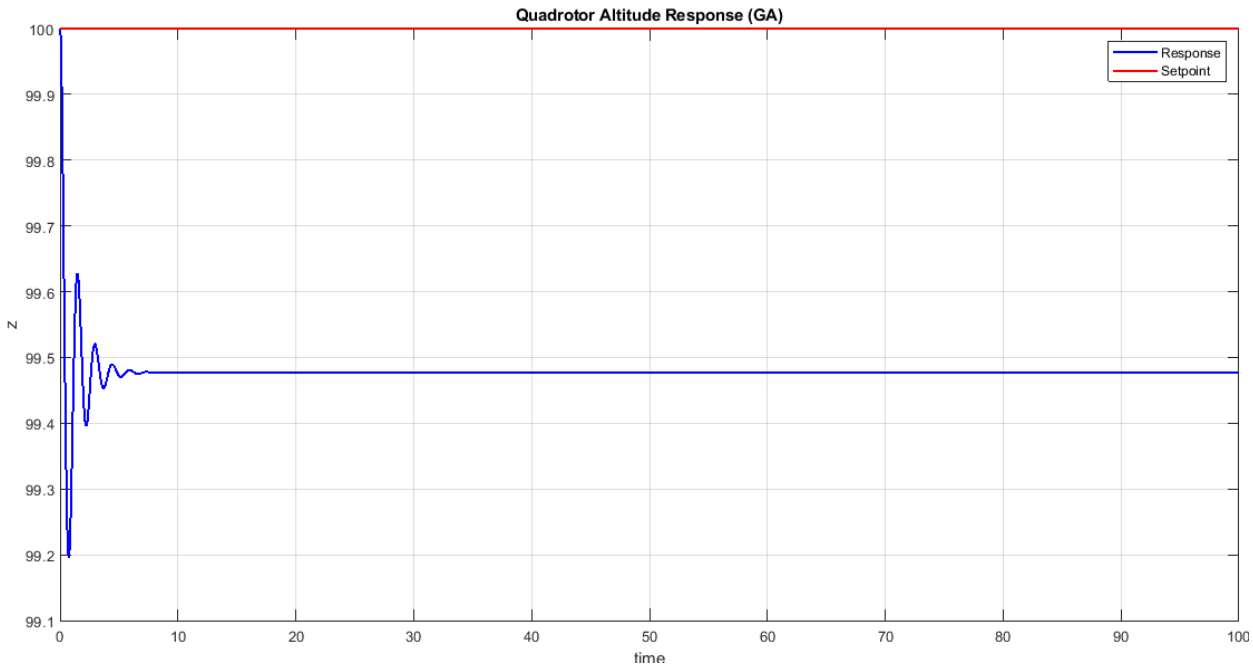

Figure 5.2.30. *Altitude Response (GA).*

Inner Loop Response

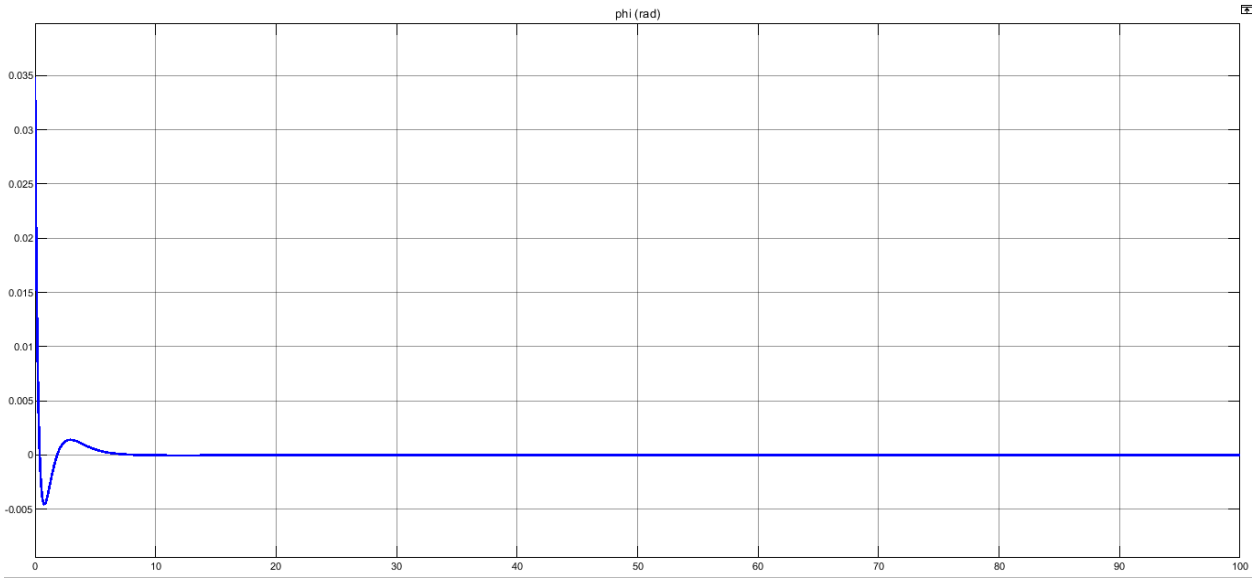

Figure 5.2.31. *Phi Angle Response (GA).*

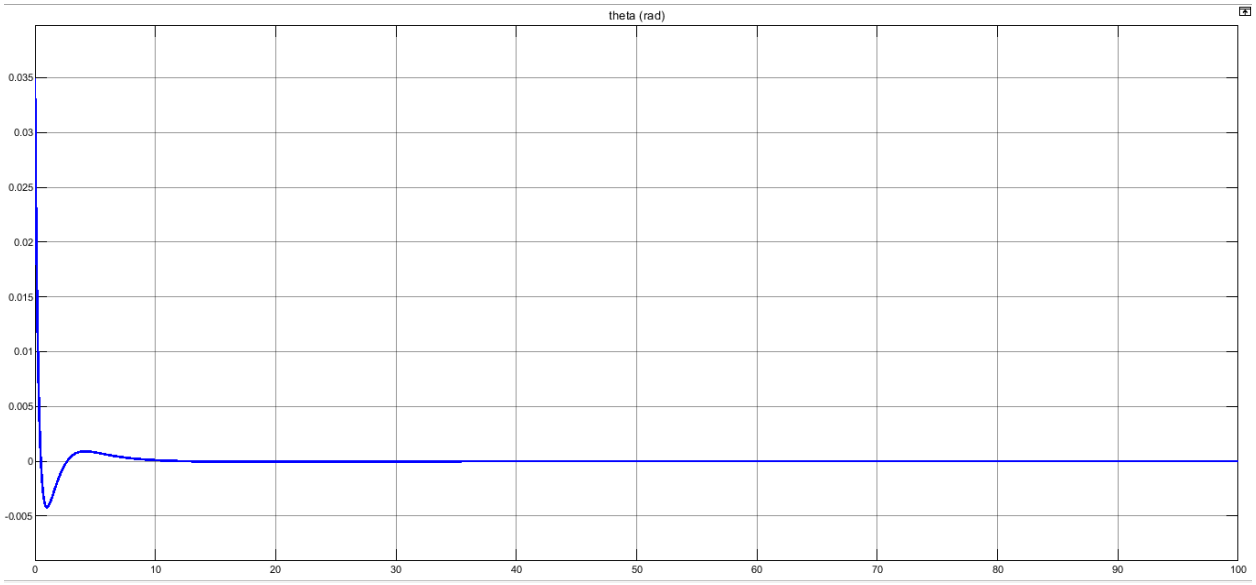

Figure 5.2.32. *Theta Angle Response (GA).*

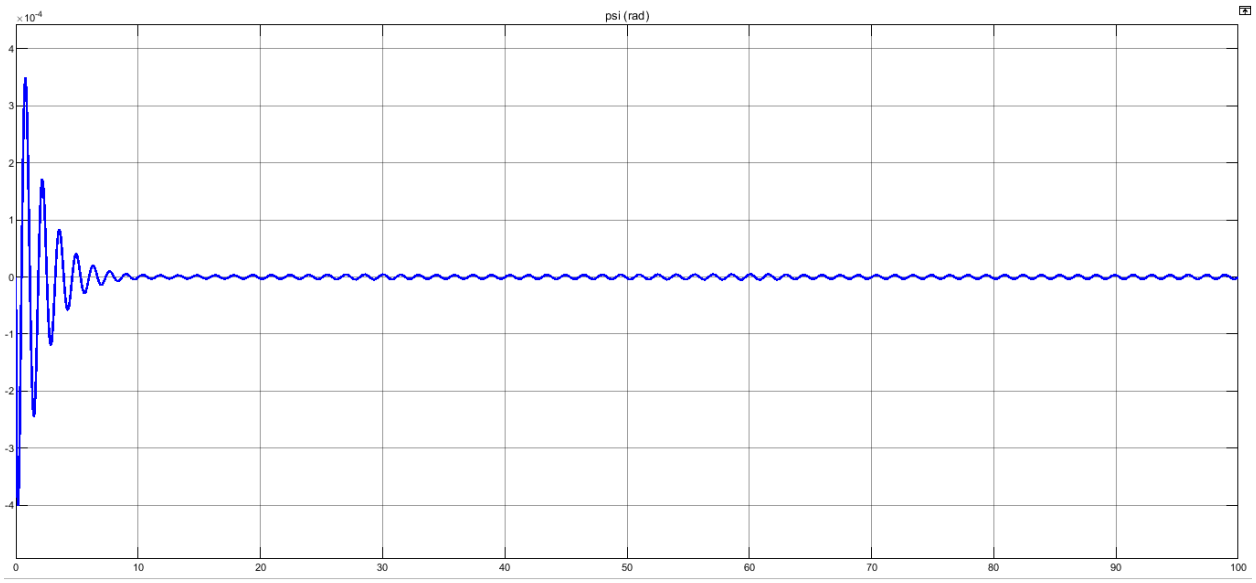

Figure 5.2.33. *Psi Angle Response (GA).*

Discussion

In both test scenarios; step response and disturbance rejection, the outer loop response (x & y) positions improved greatly and that was reflected in the performance criteria as shown above. As for altitude (z), the response did not settle at the setpoint desired exactly, but the error was in the range of 5% of the setpoint. Hence, the controller works best in XY planar motion. On the other hand, difficulties might be present in 3D motion.

Performance Further Improvement

The obtained results may be refined further to achieve better performance. This could be achieved by expanding the search area to include more potentially viable solutions. In addition, optimization settings could be adjusted to have more populations and generations. However, all of this will come as computational overhead.

Another approach is using hybrid algorithms in MATLAB. Instead of just relying on Genetic Algorithm, the algorithm starts with GA and once it's finished, it passes the obtained value to another optimization method. Many methods are available including: fminsearch, patternsearch, and fminunc.

5.2.2.3 Trajectory Following

In this section, the quadrotor was tested to follow aggressive planar trajectories. The quadrotor was tested on two trajectories; circle and infinity sign.

5.2.2.3.1 Circle Trajectory Results

Response to a circle trajectory of radius 5 is shown below. The figures below show a 3D demonstration of the trajectory in x, y, z directions. Then, modeling the response of outer loop elements followed by the inner loop.

Outer Loop Response

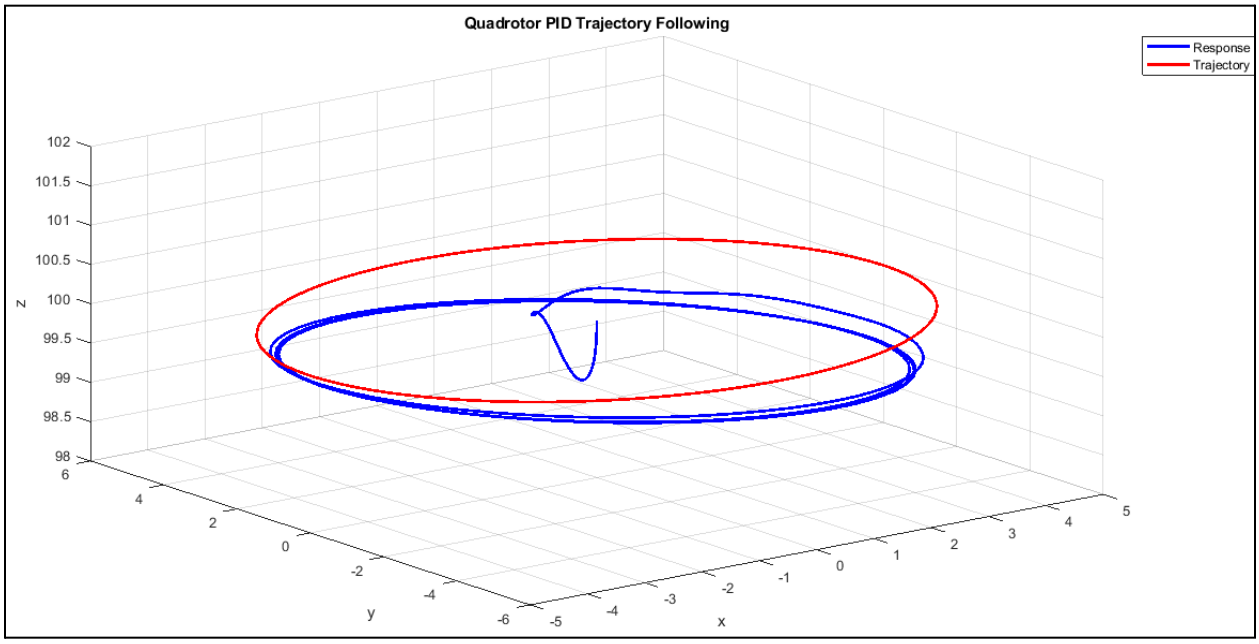

Figure 5.2.34. *3D Response of the Quadrotor.*

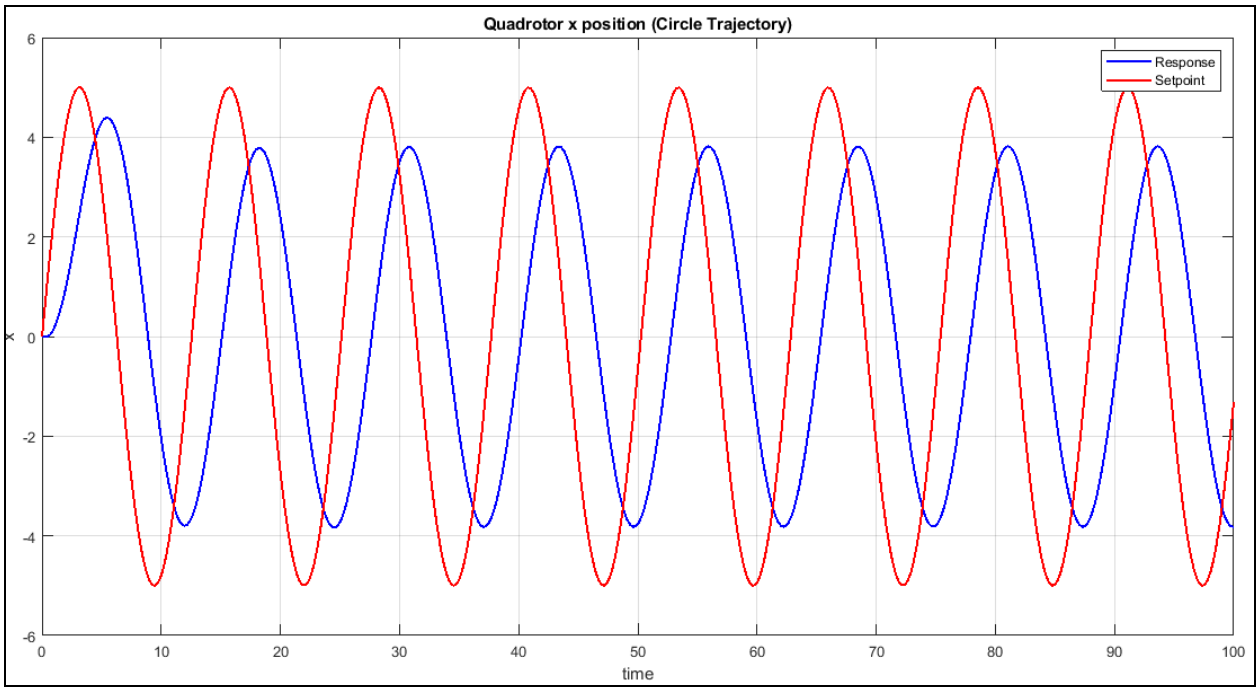

Figure 5.2.35. *X Position Response.*

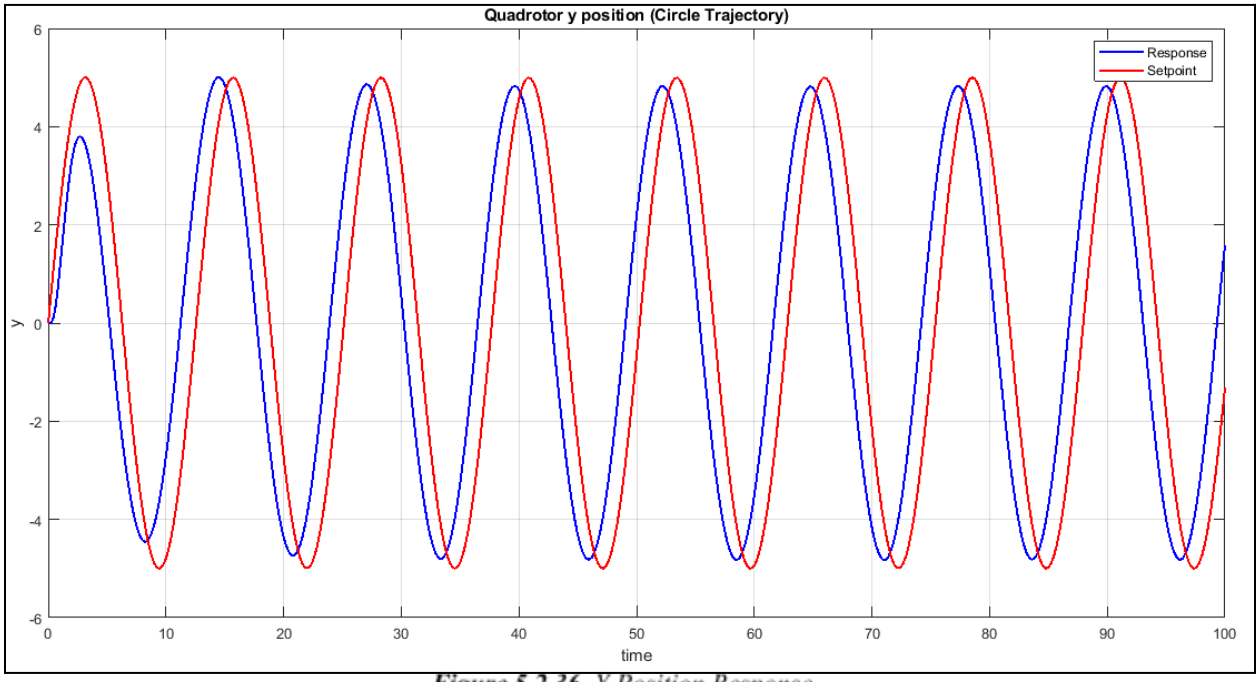

Figure 5.2.36. *Y Position Response.*

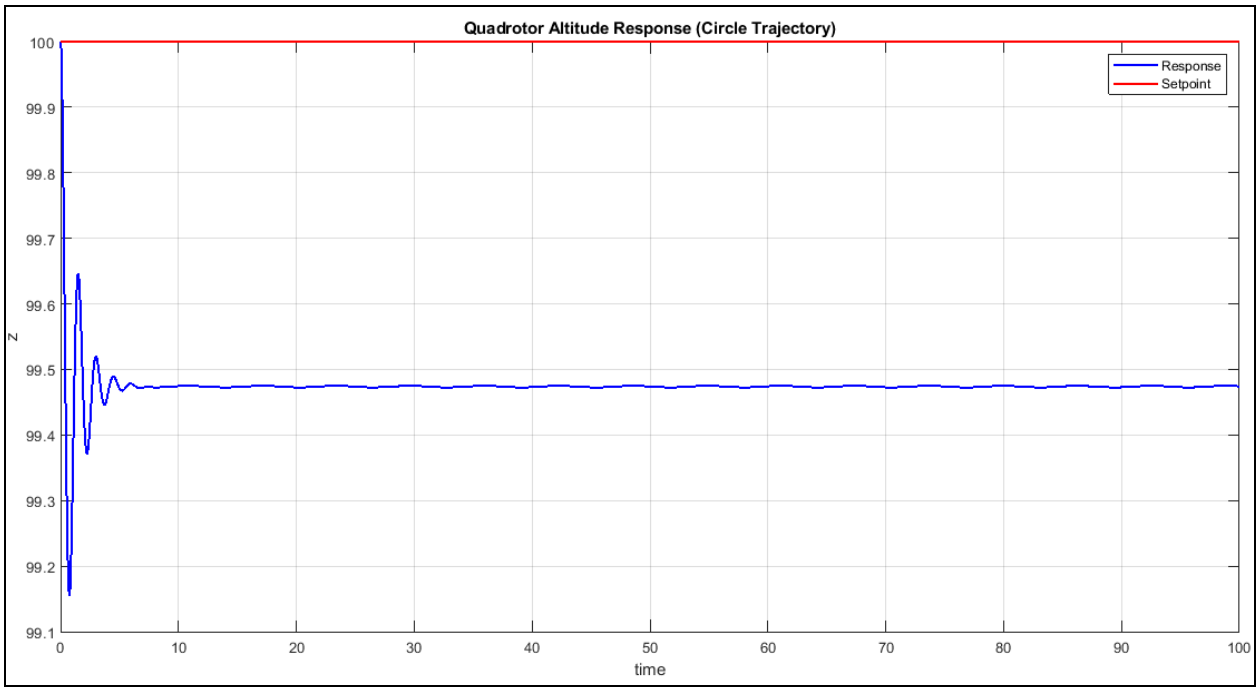

Figure 5.2.37. *Z Position Response.*

Inner Loop Response

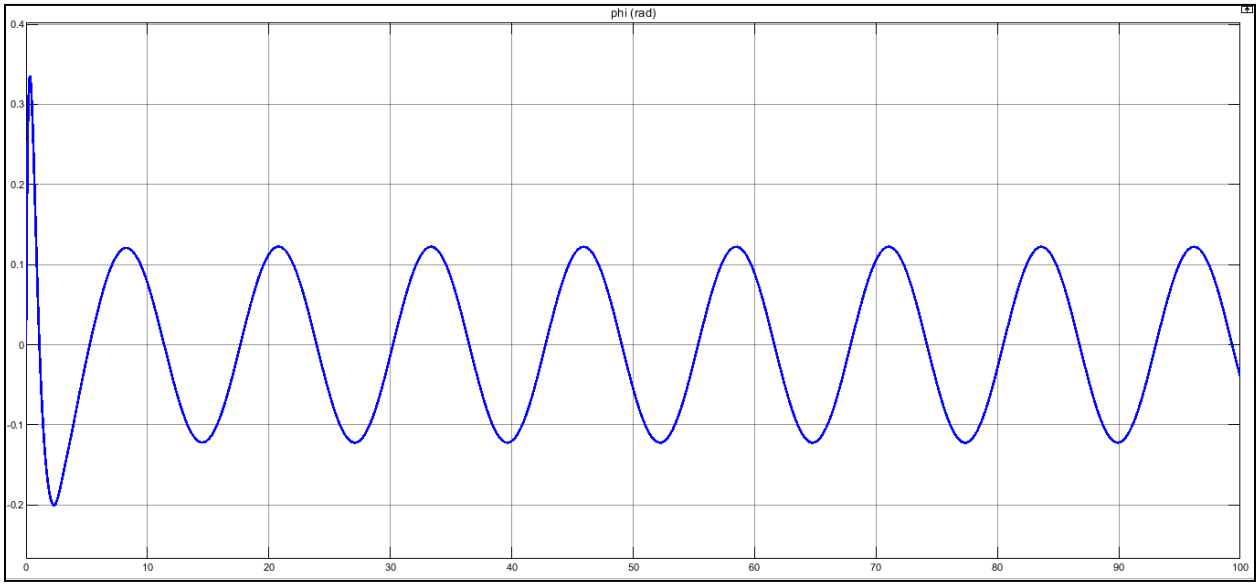

Figure 5.2.38. *Phi Angle Response.*

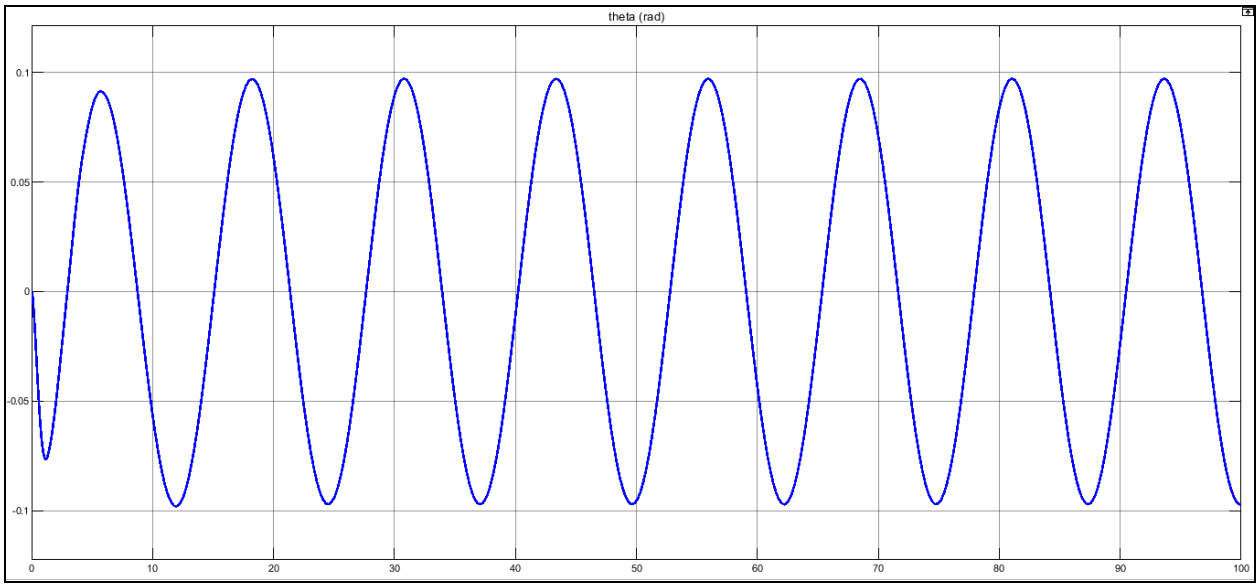

Figure 5.2.39. *Theta Angle Response.*

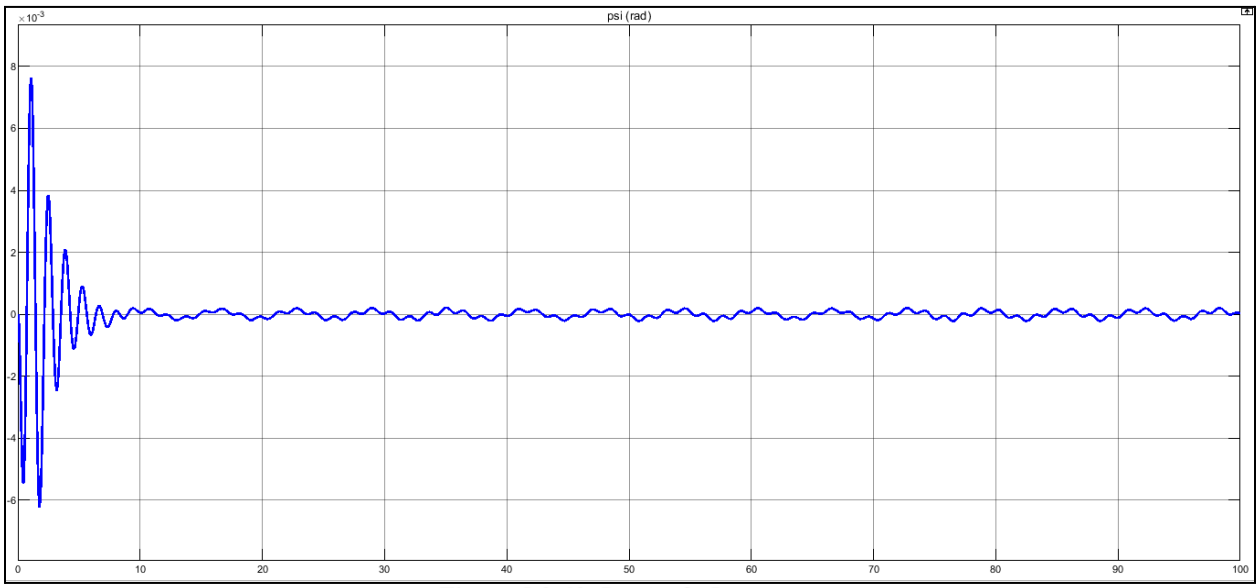

Figure 5.2.40. *Psi Angle Response.*

5.2.2.3.2 Infinity Trajectory Results

The quadrotor was also tested on an infinity-like trajectory. Results are shown below.

Outer Loop Response

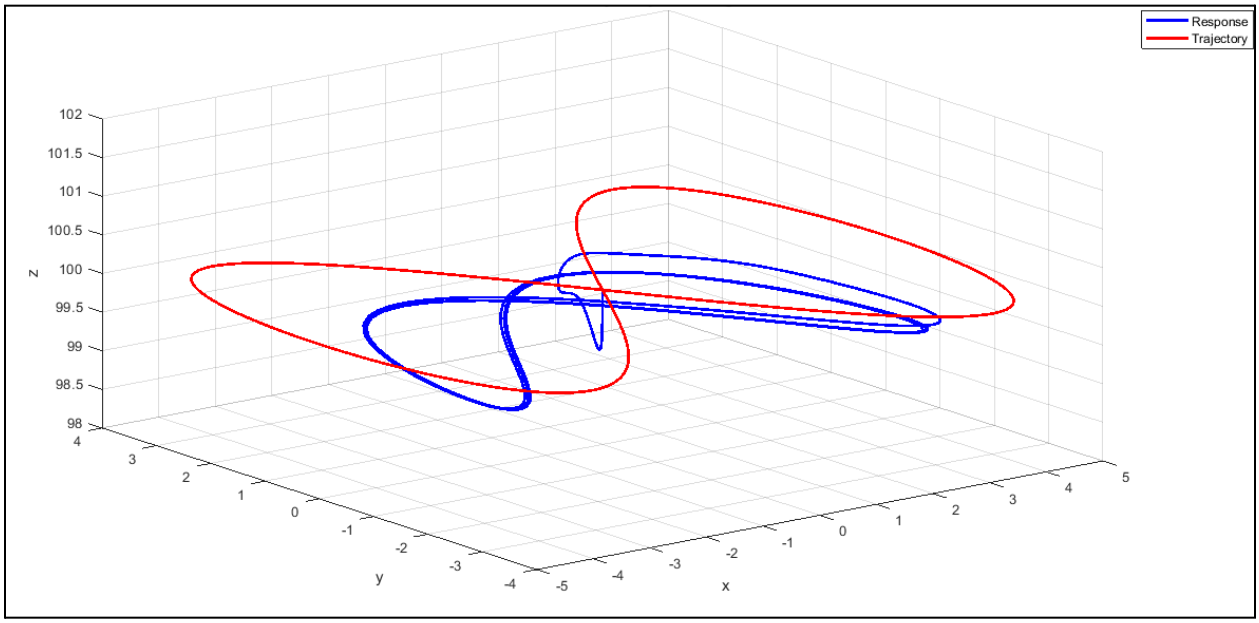

Figure 5.2.41. *3D Response of the Quadrotor.*

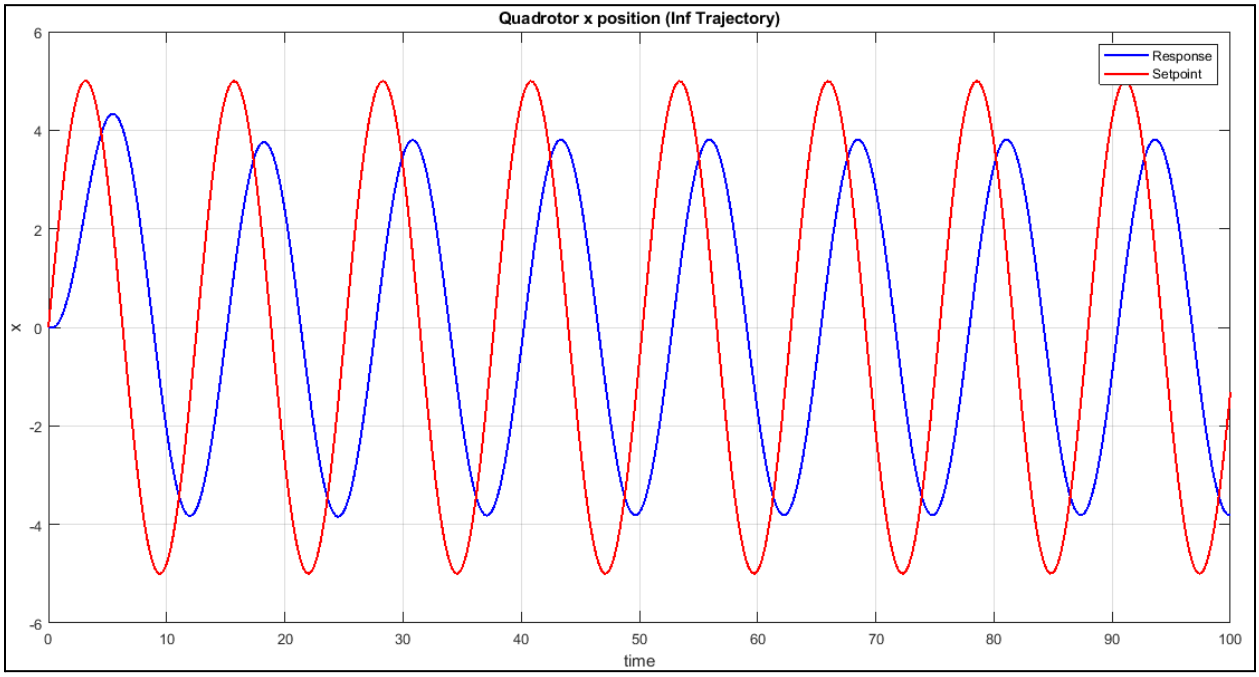

Figure 5.2.42. *X Position Response.*

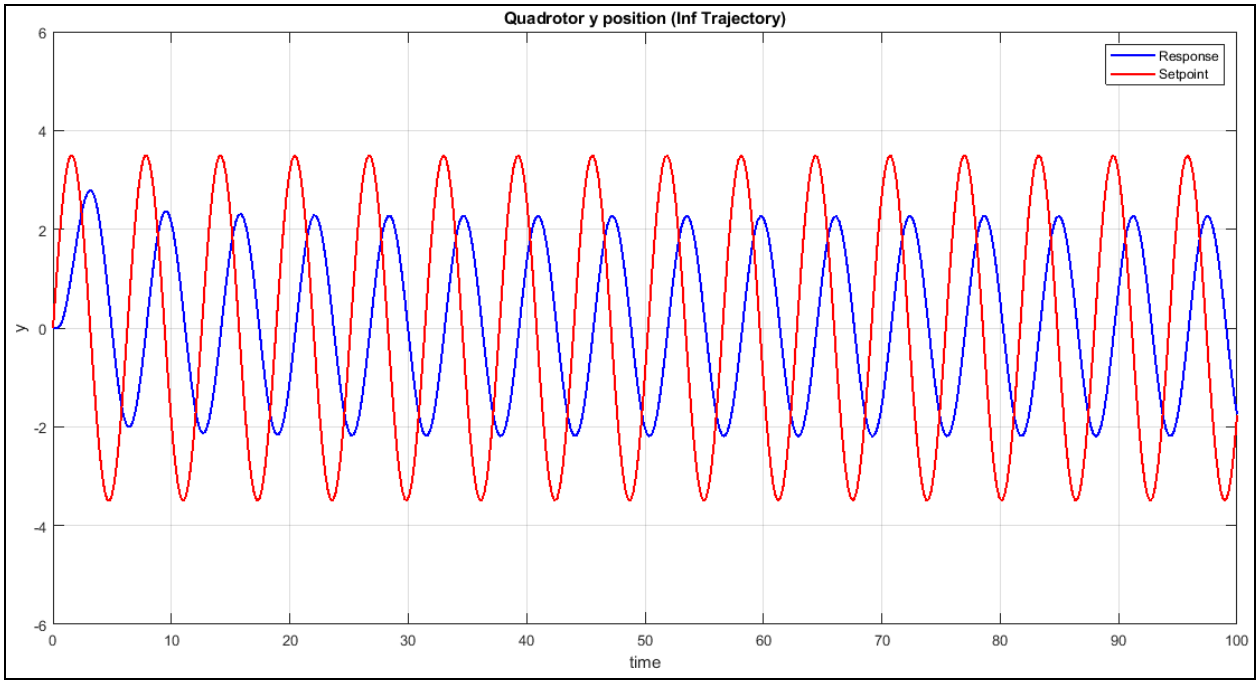

Figure 5.2.43. *Y Position Response.*

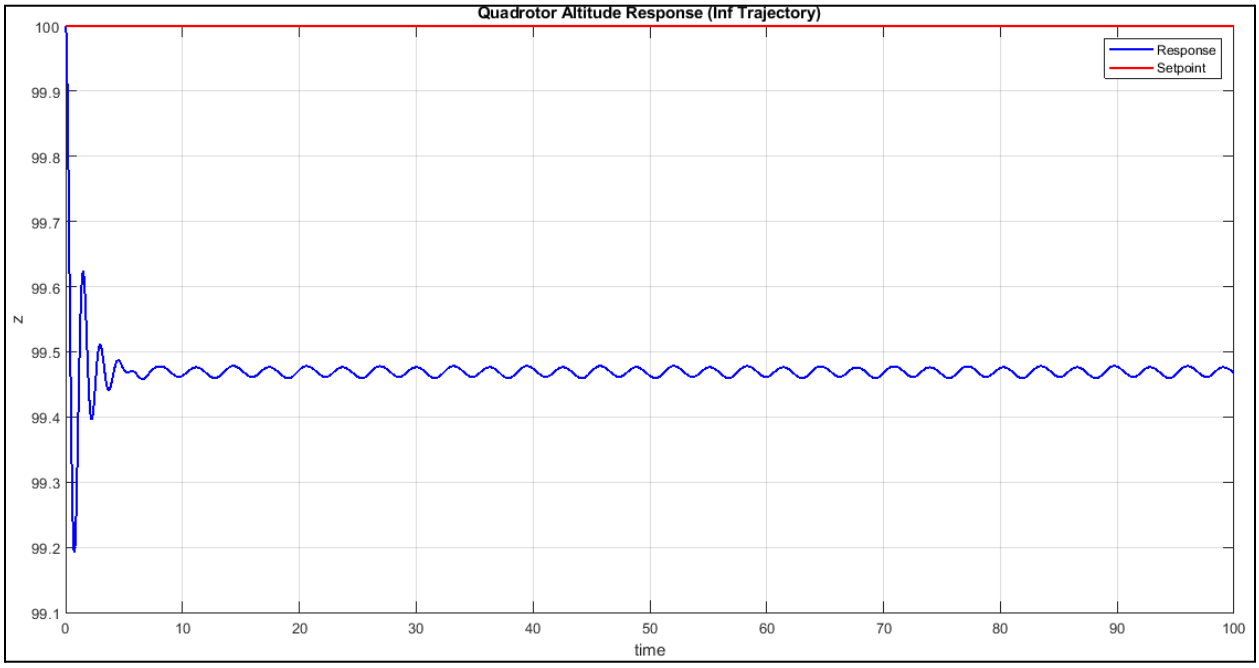

Figure 5.2.44. *Z Position Response.*

Inner Loop Response

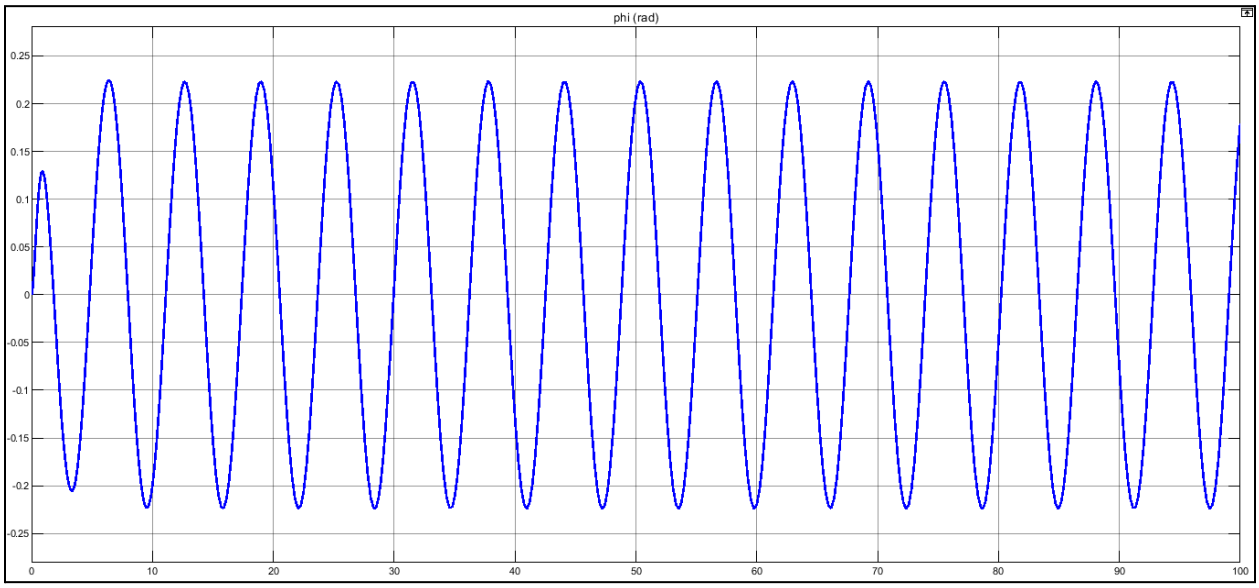

Figure 5.2.45. *Phi Angle Response.*

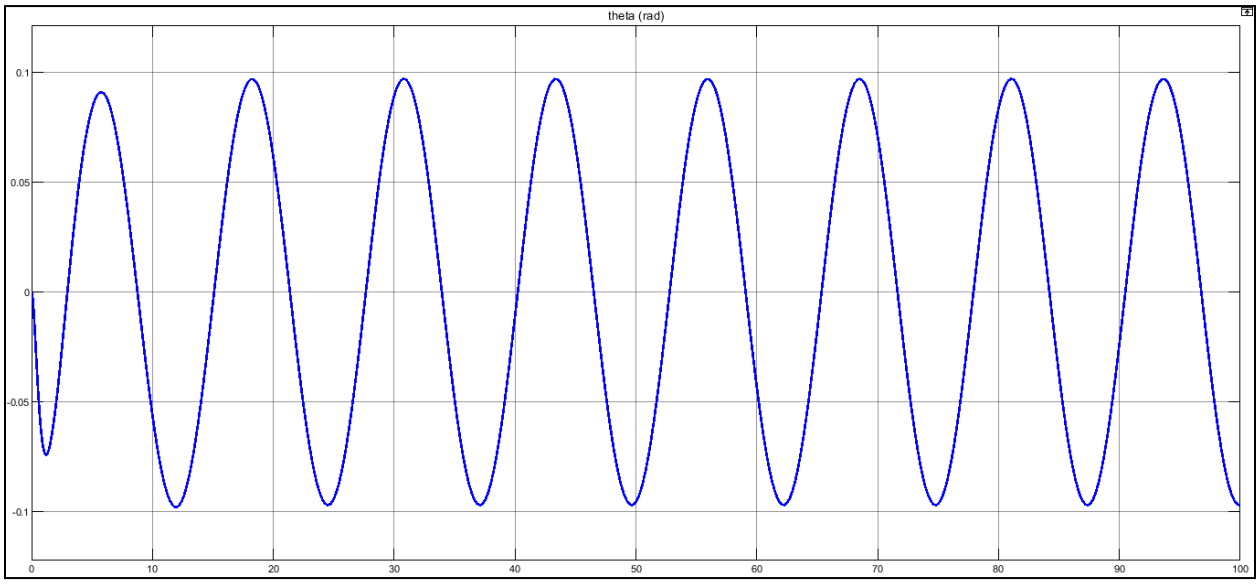

Figure 5.2.46. *Theta Angle Response.*

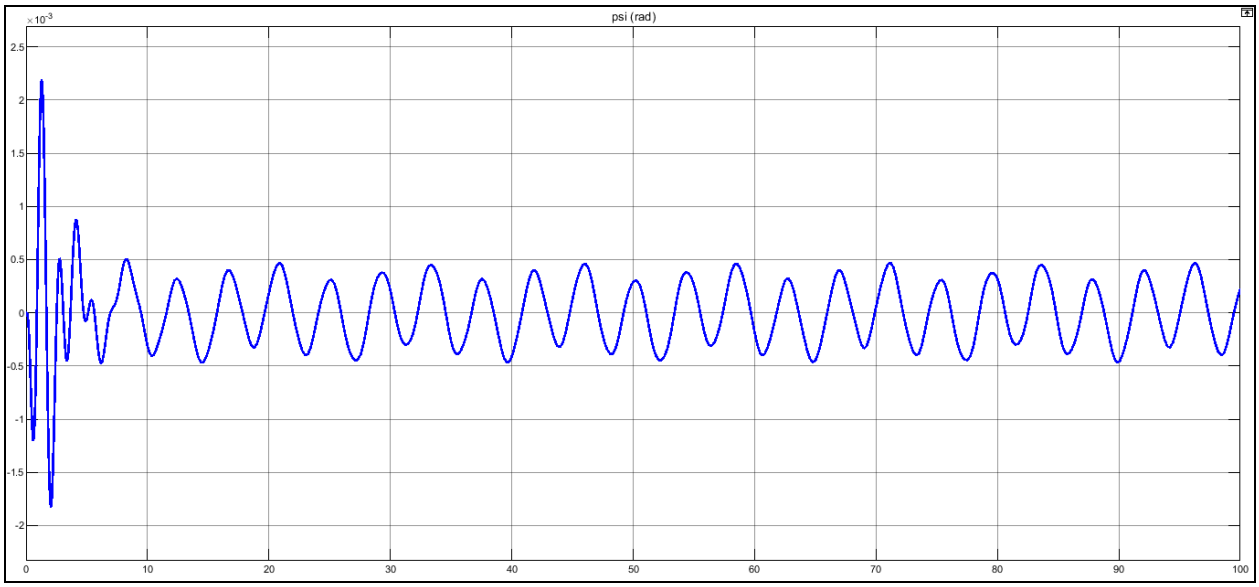

Figure 5.2.47. *Psi Angle Response.*

Discussion

The PID controller achieved satisfactory results tracking the circle trajectory, with errors in x, y, and z of 12%, 4%, and 0.5% respectively. The error in the x position could be reduced by further tuning of the controller so as to achieve better results.

5.2.3. Comparison between PID and NMPC

This comparison section focuses on contrasting the performance of the previously implemented cascaded PID controller and the aforementioned Nonlinear Model Predictive Controller approach for a quadrotor system. Three key aspects considered for comparison include reference tracking, disturbance rejection, and noise resilience.

Regarding reference tracking, the PID controller, known for its simplicity, exhibits satisfactory performance for basic reference trajectories due to its linear nature. However, when faced with more complex reference trajectories or nonlinear dynamics, the limitations of the PID controller become apparent (Spitzer, 2022). On the other hand, NMPC, with its ability to optimize control inputs over a finite prediction horizon, offers improved tracking performance, especially for intricate reference trajectories and challenging operating conditions (Spitzer, 2022) (Islam, & Okasha, 2019).

In terms of disturbance rejection, the PID controller struggles to effectively mitigate disturbances that impact the quadrotor's dynamics. Although the integral term can reduce

steady-state errors caused by disturbances, it may not adequately handle sudden or significant disturbances. In contrast, NMPC explicitly incorporates disturbance models or adapts the prediction model during the optimization process, enabling superior disturbance rejection. This feature makes NMPC more suitable for scenarios with anticipated or present disturbances.

Furthermore, the impact of noise addition on control performance is essential to consider. PID controllers, due to their linear nature, may exhibit limited resilience to noise, as they are unable to explicitly account for uncertainties (Islam, & Okasha, 2019). Conversely, NMPC demonstrates enhanced robustness against noise by incorporating state estimation techniques and employing robust optimization methods. This ability to effectively handle uncertainties and adapt control inputs contributes to improved control performance and stability in the presence of noise(Spitzer, 2022).

By evaluating the reference tracking capability, disturbance rejection, and noise resilience, a comprehensive analysis is conducted to assess the advantages and limitations of both PID control and NMPC for quadrotor control. This comparison provides valuable insights into the suitability of each control approach based on the specific requirements and challenges of the quadrotor system under investigation (Islam, & Okasha, 2019).

5.2.3.1. Reference Tracking

For the reference tracking comparison, both controllers are given the same reference trajectory to follow under the same constraints. Controllers are expected to follow the given reference trajectory with minimal overshoot and delay. Following an infinity-shaped trajectory while hovering at 100 m ($x = 5\sin(0.5t)$, $y = 3.5\sin(t)$, and $z = 100$).

The NMPC follows the trajectory with minor errors such as; an overshoot in the x direction of 15.02% and a steady state error in the z direction of 0.3 m (0.3%), which is shown in the states response for such a trajectory in Fig. 5.2.5. Noticeably, the NMPC control action makes the quadrotor motion very smooth following the trajectory however having sharp maneuvers to make. Besides, the control action response shown in Fig. 5.2.6 shows no

aggressive control action.

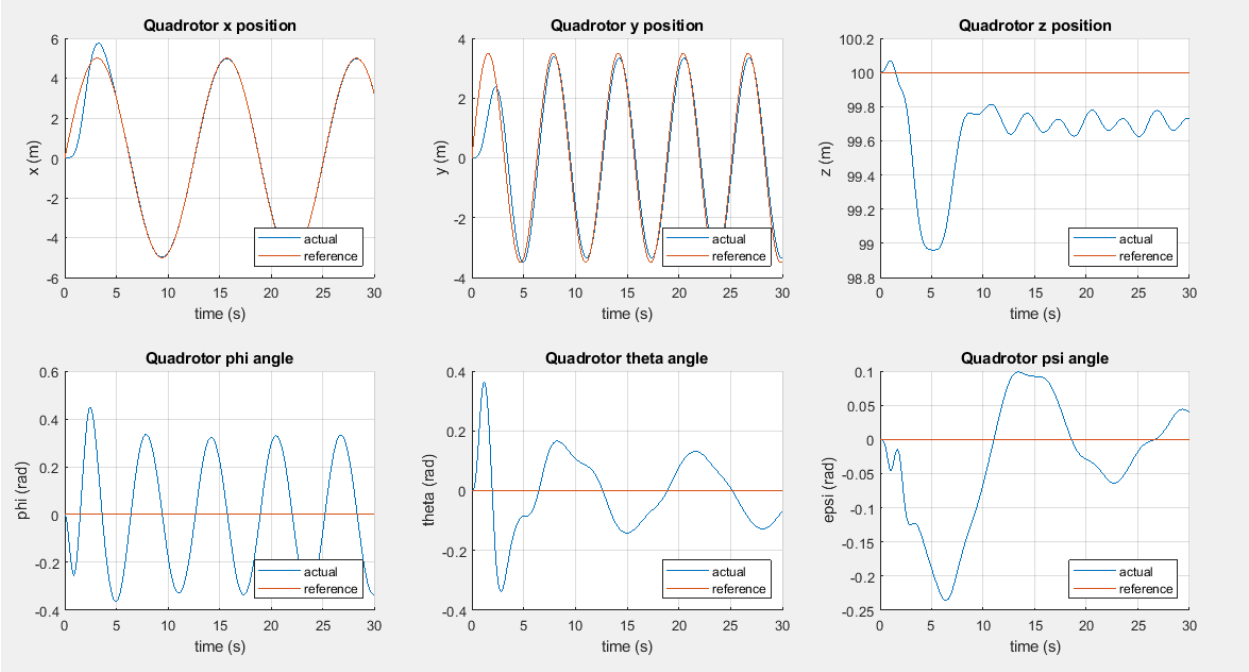

Fig. 5.2.48. The states' responses for an infinity-shaped trajectory while hovering at 100 m.

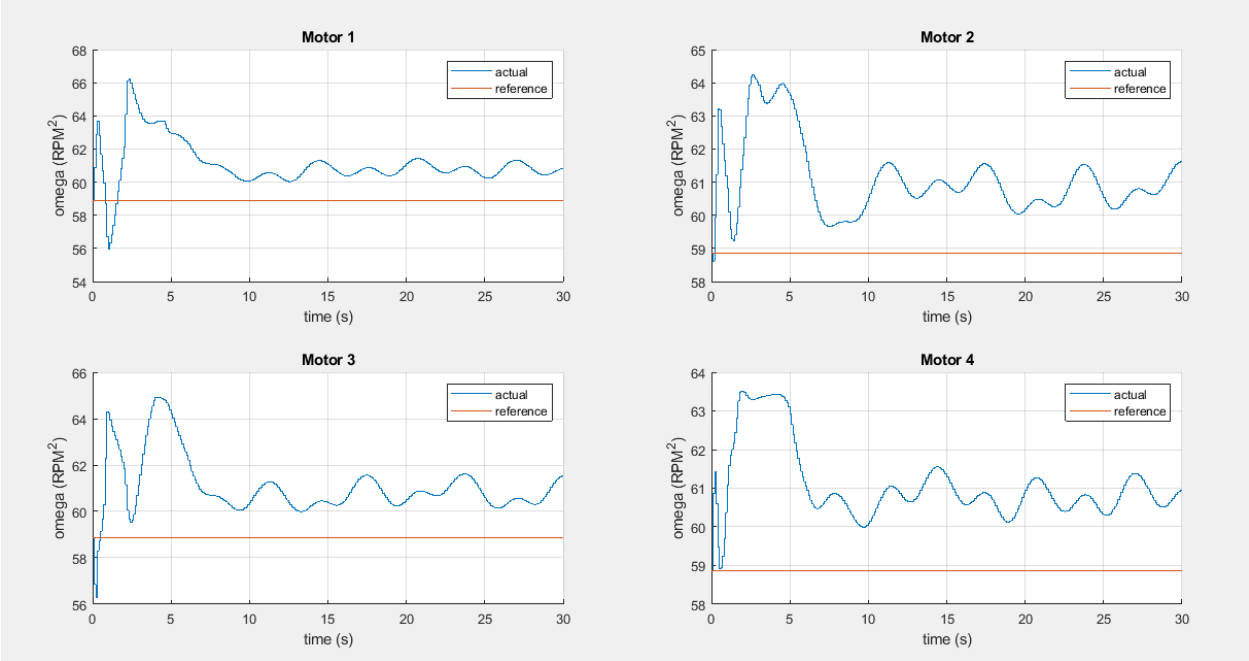

Fig. 5.2.49. The NMPC control actions to follow an infinity-shaped trajectory while hovering at 100 m.

As for the PID response, the controller did not cause any overshoot in tracking the position setpoints in x, y, and z. However, steady-state errors of around 12% occurred in both x

and y. As for the altitude response, the controller managed to maintain the aircraft at the desired setpoint but with an error of 0.5%. Accordingly, NLMPC outperformed the PID controller in tracking an aggressive trajectory.

5.2.3.2. Disturbance Rejection

For the disturbance rejection comparison, a minor step disturbance is introduced to one of the angles while hovering, the system then is observed to check the controllers' recovery of the hovering state rapidly and steadily. The responses of the PID and NMPC are expected to resist the disturbance introduced and recover the quadrotor hover state smoothly with no aggressive responses.

Introducing a 10-degree step disturbance to the phi angle while hovering at 100 m, the state response for 40 s of the NMPC control actions, as shown in Fig.5.2.7, and 5.2.8, indicates that the system y and z positions have settled already but the x position and the theta angle seem to be diverging, however, it still within very small values (magnitudes of $10^{-14} and\ 10^{-15}$), so a longer response is needed to judge.

Running another 10-degree step disturbance to the phi angle while hovering at 100 m for 1000s, the state response in Fig. 5.2.9 shows that the x position, z position, phi angle, and epsi angle have already totally settled, meanwhile, the y position and the theta angle are oscillating within very low magnitudes (vibrating). It is also worth noting that the motor speed (the NMPC actions) has already been set to its nominal hovering value very early in the simulation, as Fig. 5.2.10 states. This means that the vibrations happening in the system are not caused by the controller but rather inherited from the system model itself.

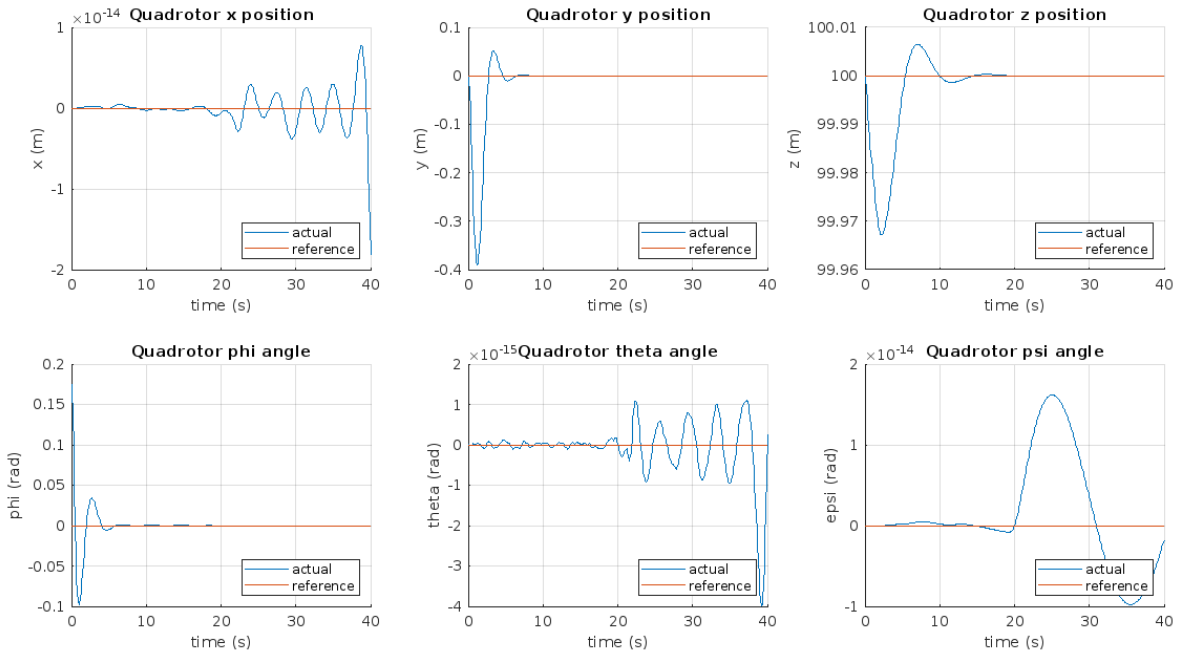

Fig. 5.2.50. The states' responses for a 10 deg disturbance in the phi angle while hovering at 100 m for 40 s.

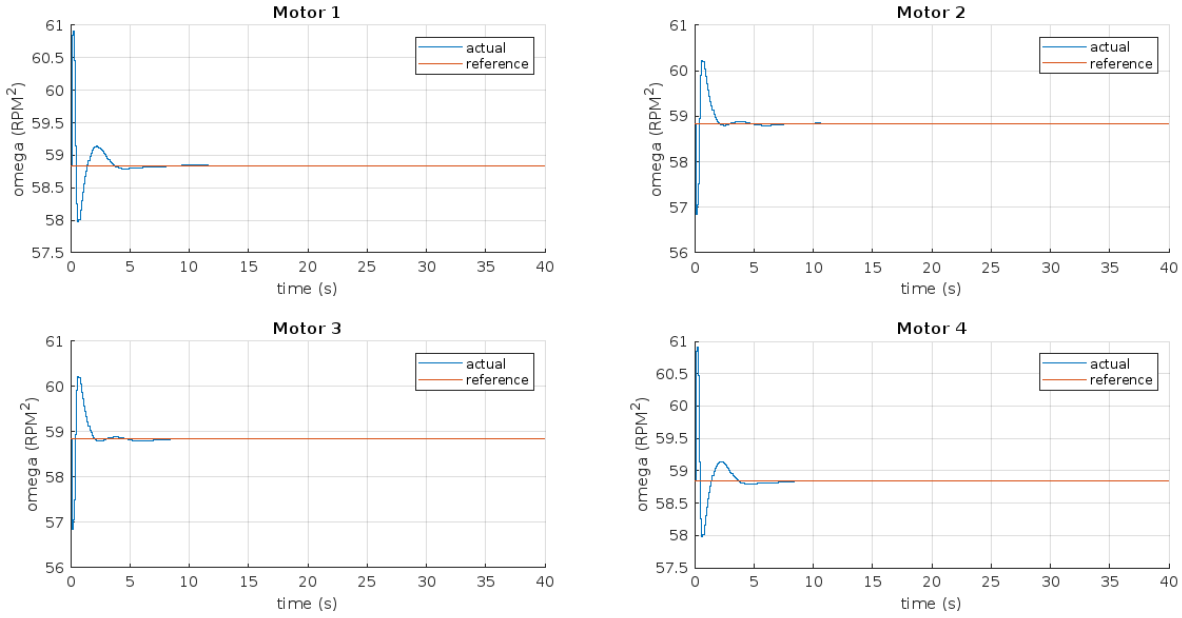

Fig. 5.2.51. The control action responses for a 10 deg disturbance in the phi angle while hovering at 100 m for 40 s.

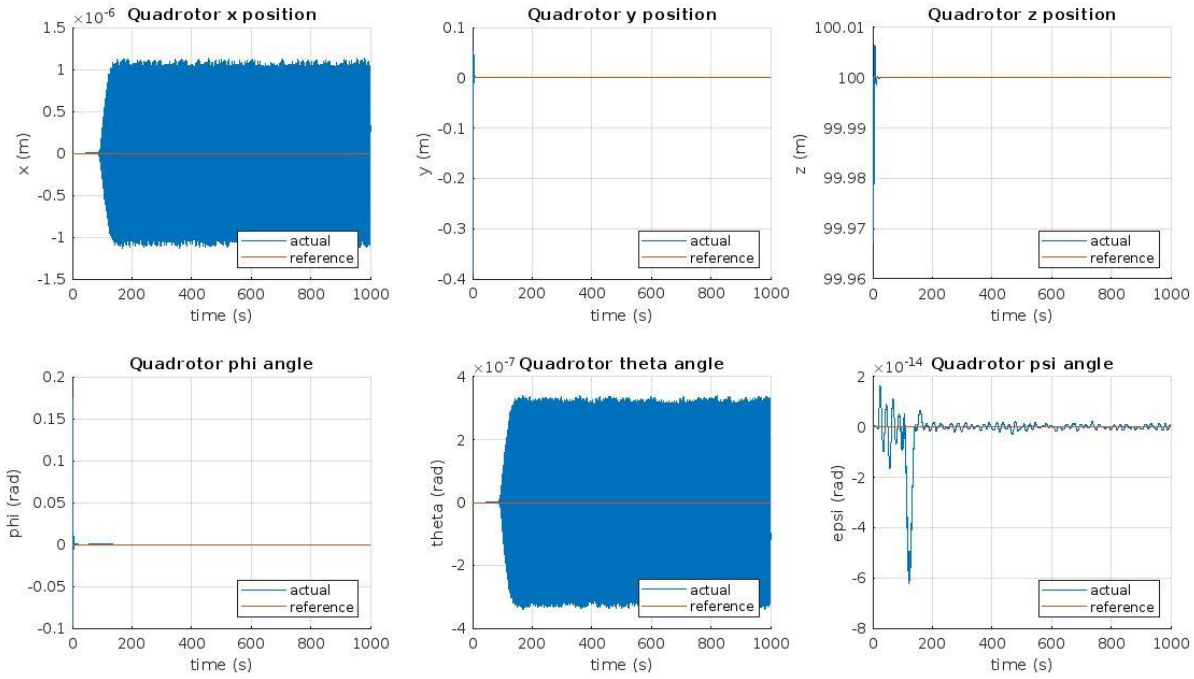

Fig.5.2.52. The states' responses for a 10 deg disturbance in the phi angle while hovering at 100 m for 1000 s.

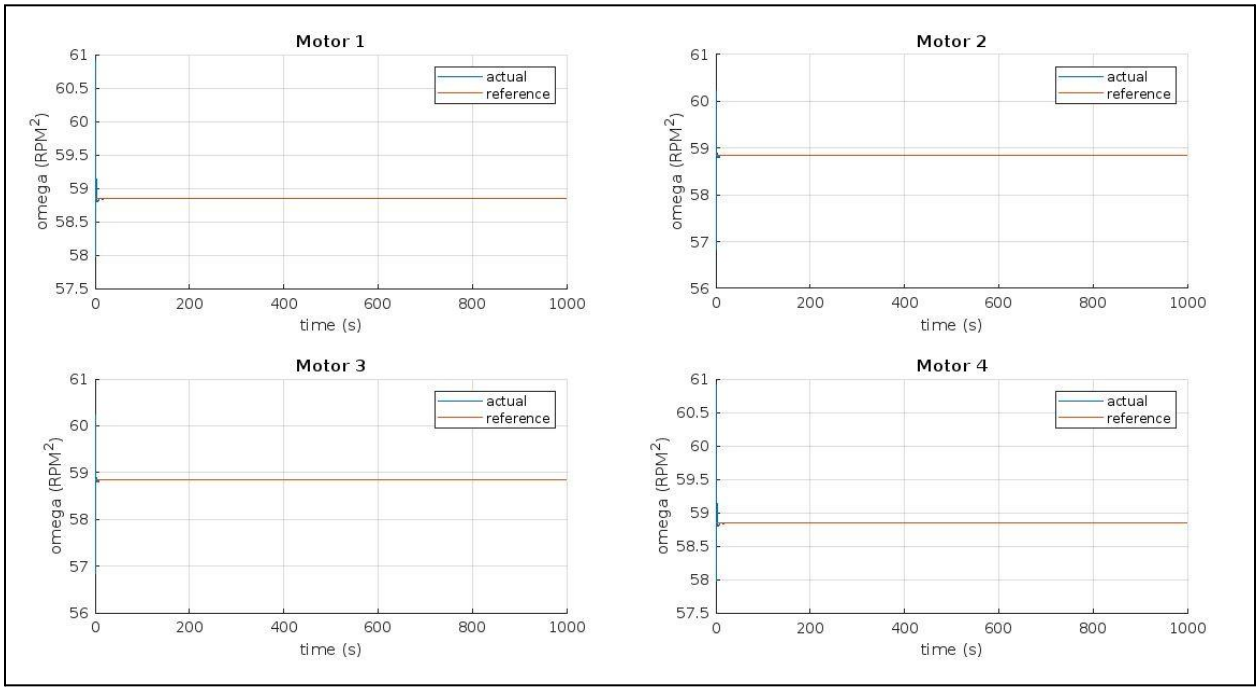

Fig.5.2.53. The control action responses for a 10 deg disturbance in the phi angle while hovering at 100 m for 1000 s.

The same procedure is repeated for the theta angle with a disturbance of 10 degrees. Fig.5.2.11, and 5.2.12 draw similar results as that of the phi angle disturbance, but this time the y position, epsi, and phi angles are the ones that seem to be diverging and need a longer response to judge while the other parameters settle already. Again, in Fig. 5.2.13, 5.2.14 when the simulation response is extended to 1000 s. It highlights the same conclusions reached with the phi angle disturbance responses, that the epsi keeps oscillating within a very low magnitude (vibrating) and again it is obvious that it is from the system model and not caused by a control action.

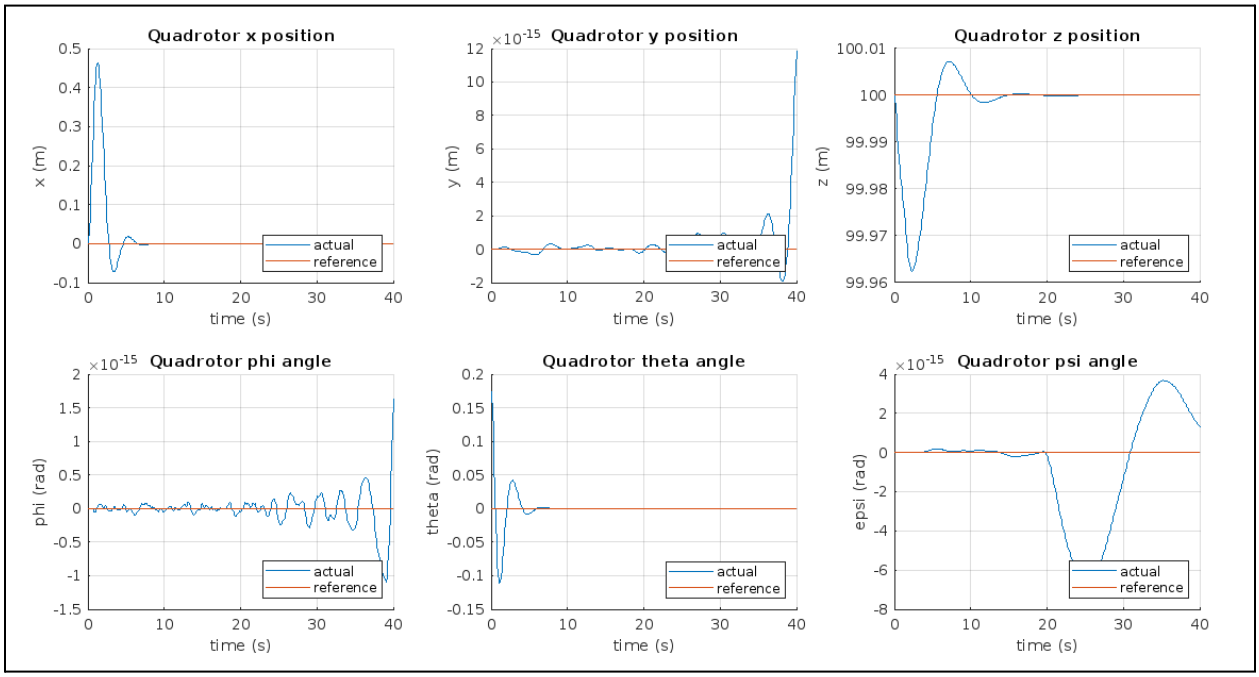

Fig.5.2.54. The states' responses for a 10 deg disturbance in the theta angle while hovering at 100 m for 40 s.

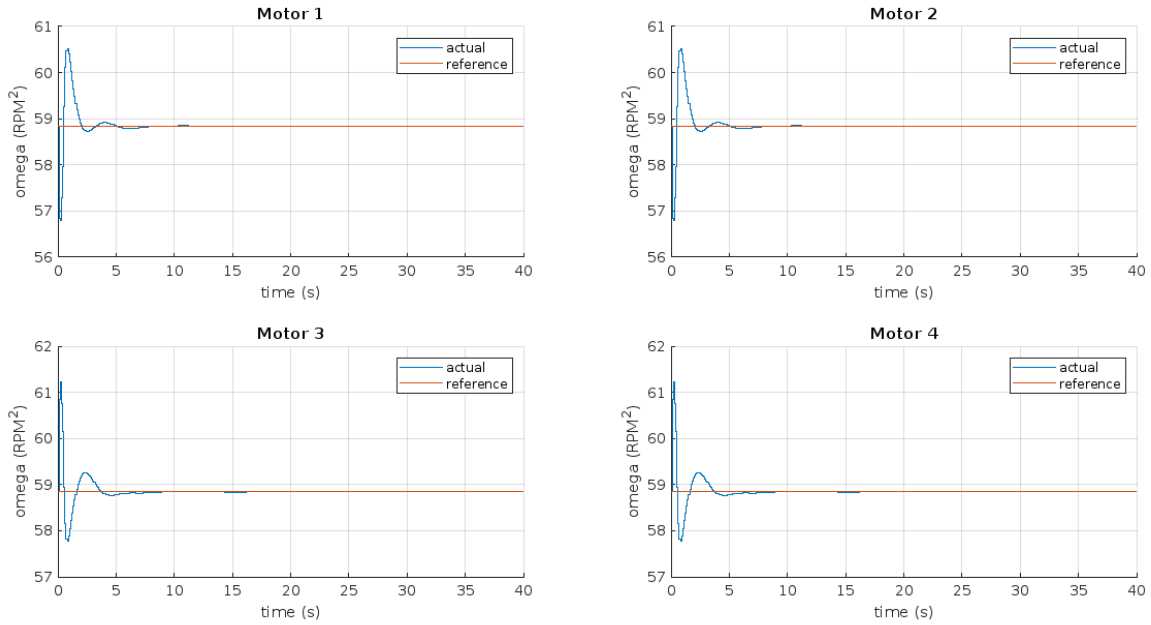

Fig.5.2.56. The control action responses for a 10 deg disturbance in the theta angle while hovering at 100 m for 40 s.

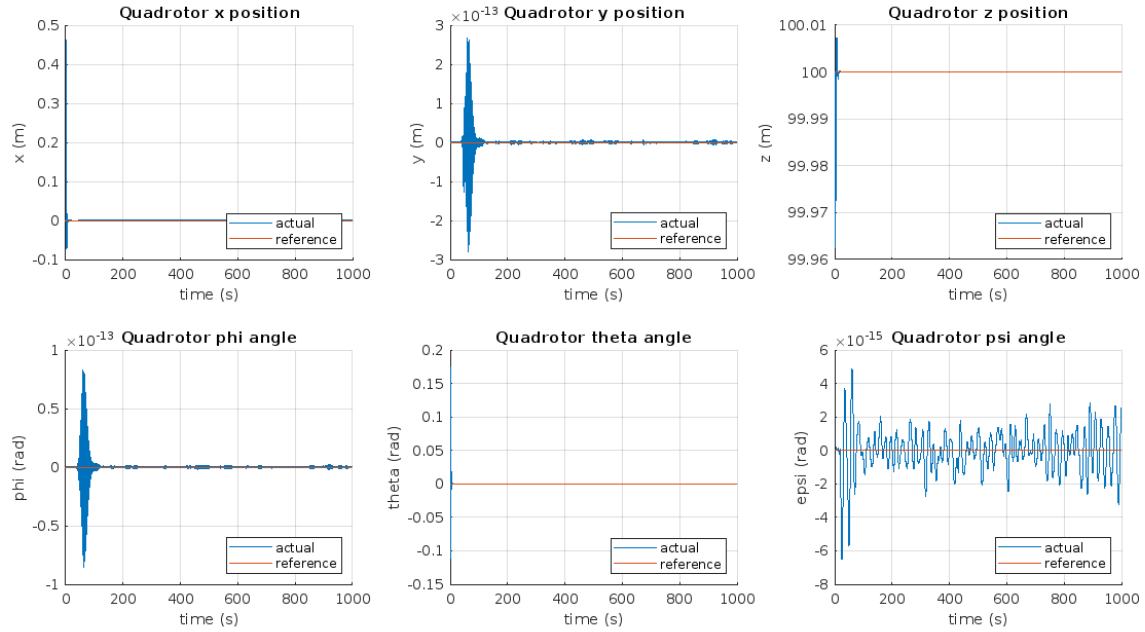

Fig.5.2.57. The states' responses for a 10 deg disturbance in the theta angle while hovering at 100 m for 1000 s.

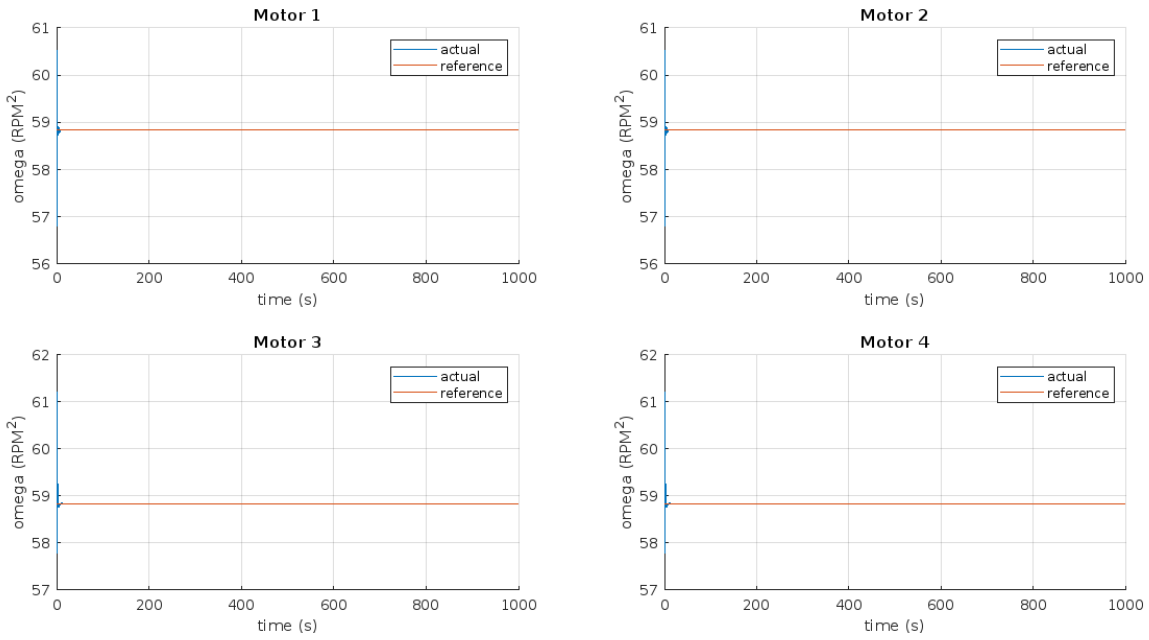

Fig.5.2.58. The control action responses for a 10 deg disturbance in the theta angle while hovering at 100 m for 1000 s.

Once more the procedure done with the phi and theta disturbances is repeated for the espi and a similar conclusion could be drawn, but this time for the x position and theta angle. The results are highlighted in Fig. 5.2.15, 5.2.16, 5.2.17, and 5.2.18 below.

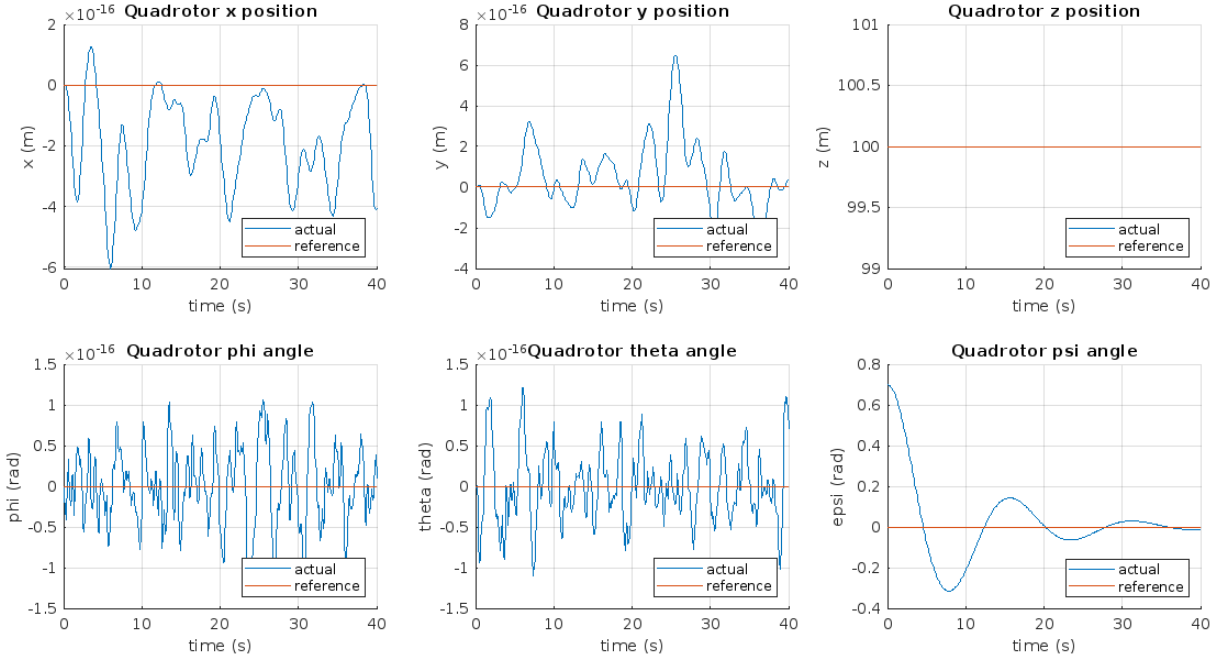

Fig.5.2.59. The states' responses for a 10 deg disturbance in the epsi angle while hovering at 100 m for 40 s.

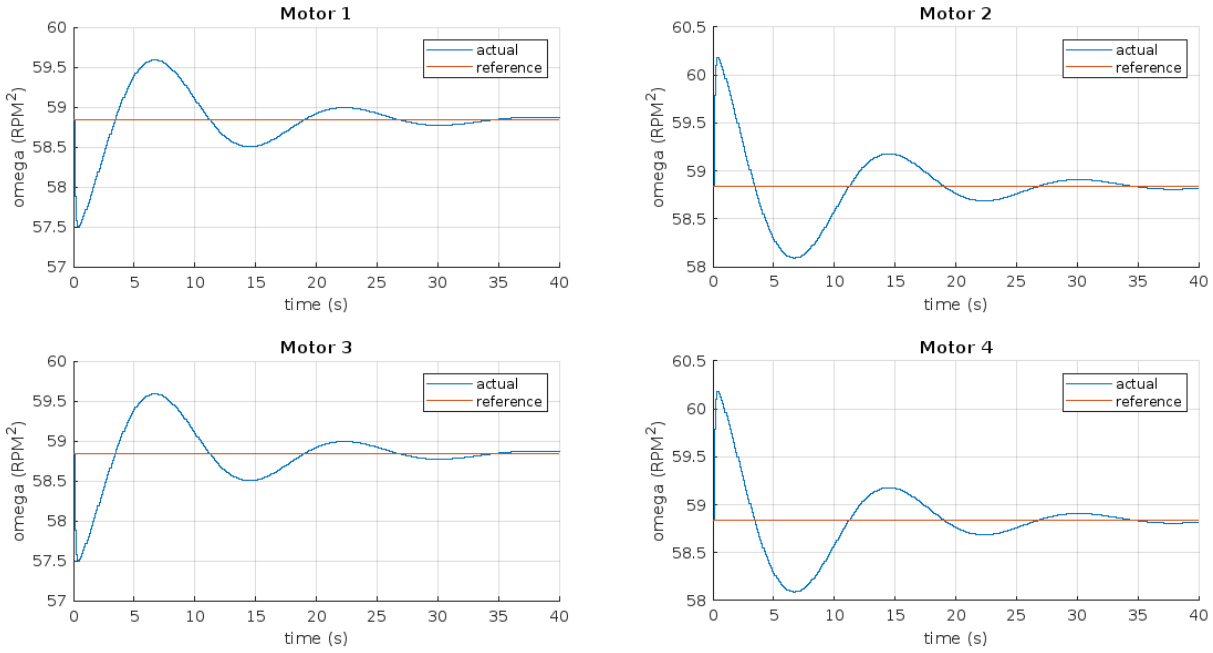

Fig.5.2.60. The control action responses for a 10 deg disturbance in the epsi angle while hovering at 100 m for 40 s.

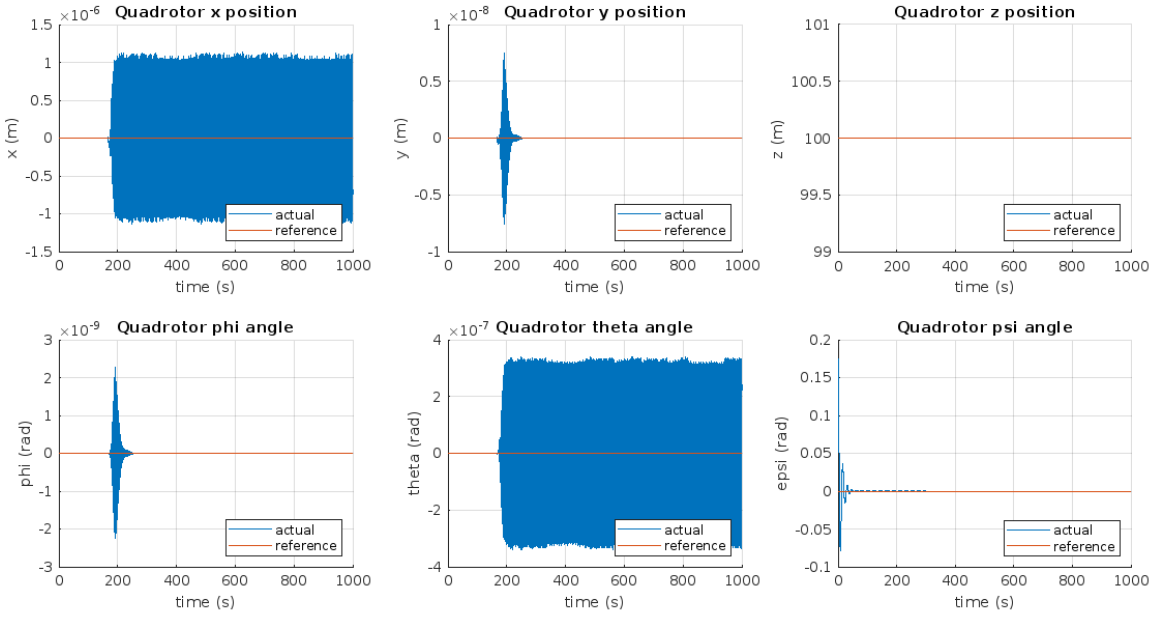

Fig.5.2.61. The states' responses for a 10 deg disturbance in the epsi angle while hovering at 100 m for 1000 s.

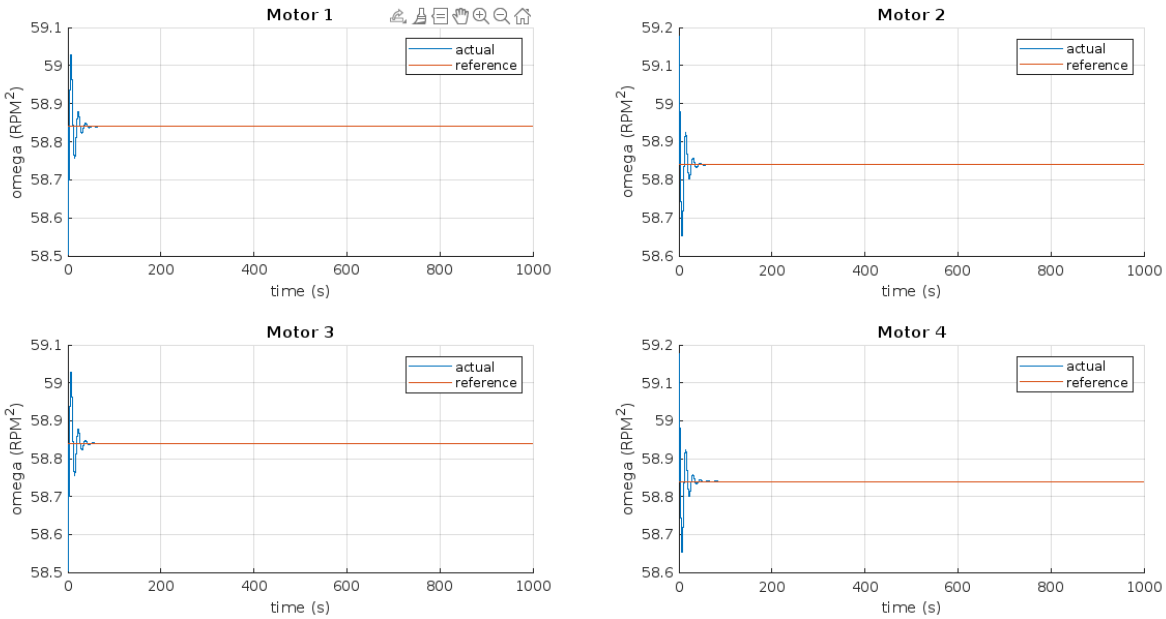

Fig.5.2.62. The control action responses for a 10 deg disturbance in the epsi angle while hovering at 100 m for 1000 s.

Another final disturbance for the three angles together shows, in Fig.5.2.19, and 5.2.20, that the quadrotor totally recovers its initial state of hovering without any vibrations as before.

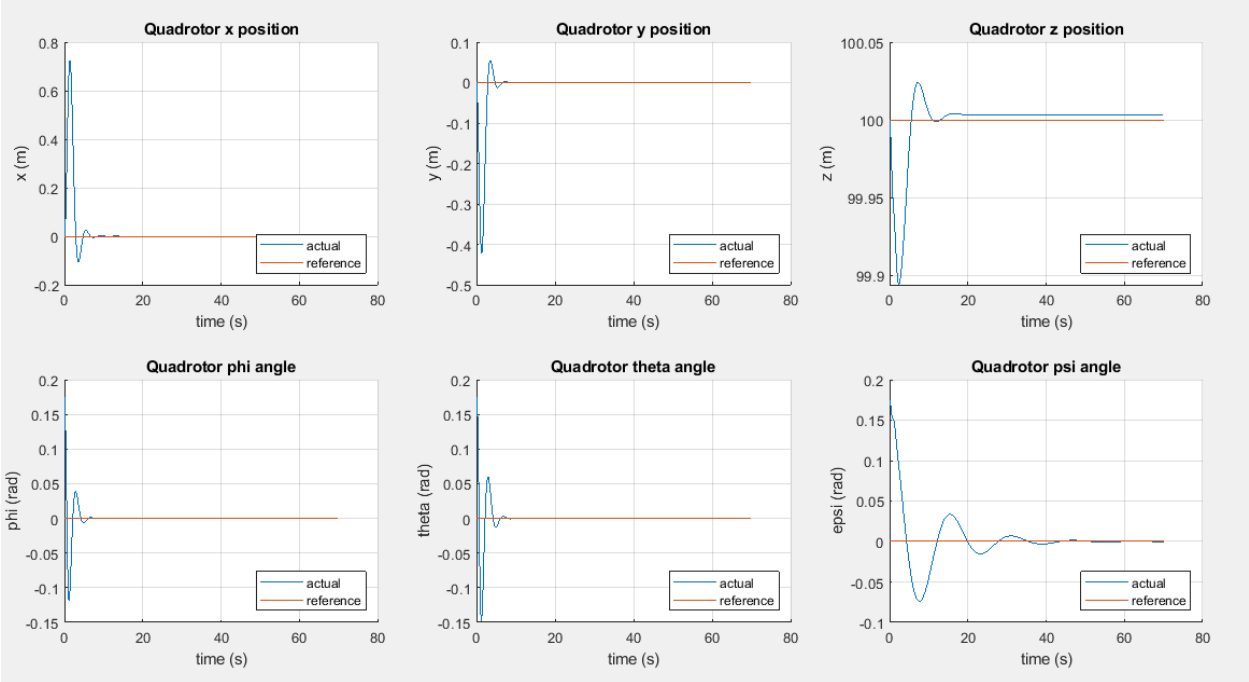

Fig.5.2.63. The states' responses for a 10 deg disturbance in all three angles while hovering at 100 m for 70 s.

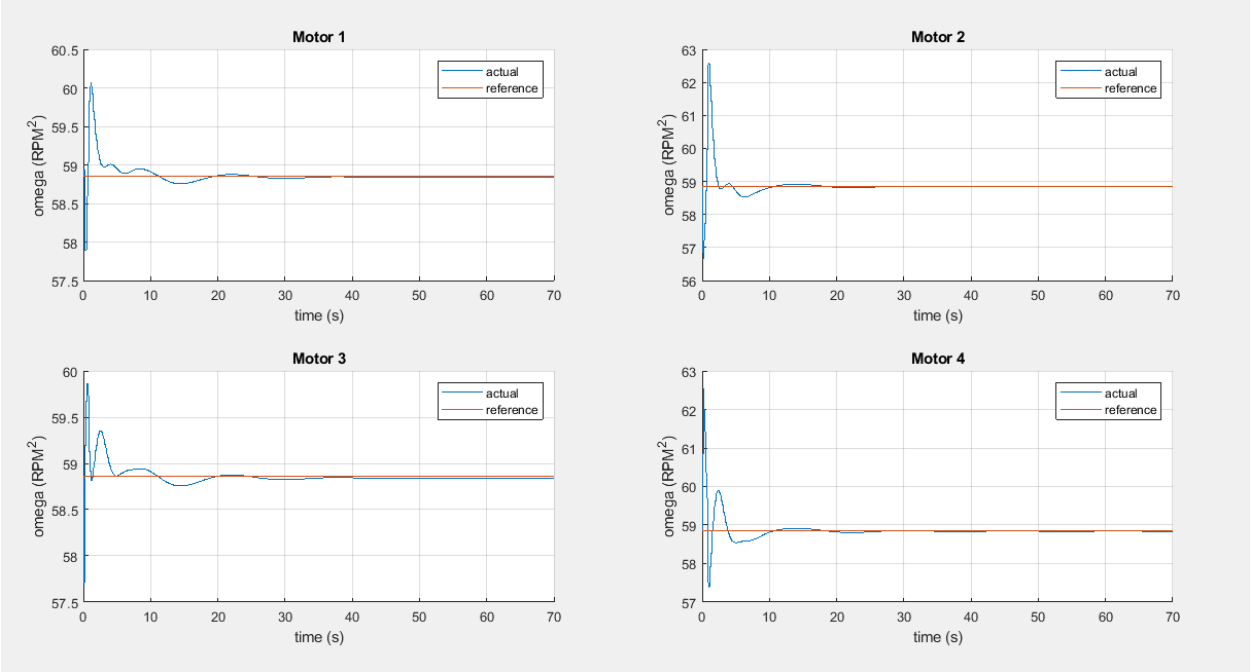

Fig.5.2.64. The control action responses for a 10 deg disturbance in all three angles while hovering at 100 m for 70 s.

As for the PID controller, it was able to perform well while experiencing an initial disturbance in both roll and pitch angles. The controller was able to recover quickly and return to the desired steady-state value. Accordingly, both PID and NLMPC performed well in disturbance rejection.