ROCHESTER INSTITUTE OF TECHNOLOGY

Model-Free Control of an Unmanned Aircraft Quadcopter Type System

By: Christian Monti

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Mechanical Engineering

Advisor: Dr. Agamemnon Crassidis

DEPARTMENT OF MECHANICAL ENGINEERING

KATE GLEASON COLLEGE OF ENGINEERING

Rochester, NY

July 28th, 2020

Model-Free Control of an Unmanned Aircraft Quadcopter Type System

By: Christian Monti

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Mechanical Engineering

Department of Mechanical Engineering

Kate Gleason College of Engineering

Rochester Institute of Technology

Approved by:

Agamemnon Crassidis — July 28th, 2020

Dr. Agamemnon Crassidis *Date*
Thesis Advisor
Department of Mechanical Engineering

Daniel Kaputa — July 28th, 2020

Dr. Daniel Kaputa *Date*
Thesis Committee Member
Department of Electrical, Computer, and Telecommunications Engineering Technology

July 28th, 2020

Dr. Jason Kolodziej *Date*
Thesis Committee Member
Department of Mechanical Engineering

Mark H. Kempski — July 30th, 2020

Dr. Mark Kempski *Date*
Thesis Committee Member
Department of Mechanical Engineering

07/30/2020

Dr. Michael Schrlau *Date*
Department Representative
Department of Mechanical Engineering

ABSTRACT

A model-free control algorithm based on the sliding mode control method for unmanned aircraft systems is proposed. The mathematical model of the dynamic system is not required to derive the sliding mode control law for this proposed method. The knowledge of the system's order, state measurements and control input gain matrix shape and bounds are assumed to derive the control law to track the required trajectories. Lyapunov's Stability criteria is used to ensure closed-loop asymptotic stability and the error estimate between previous control inputs is used to stabilize the system. A smoothing boundary layer is introduced into the system to eliminate the high frequency chattering of the control input and the higher order states. The $[B]$ matrix used in the model-free algorithm based on the sliding mode control is derived for a quadcopter system. A simulation of a quadcopter is built in Simulink and the model-free control algorithm based on sliding mode control is implemented and a PID control law is used to compare the performance of the model-free control algorithm based off of the RMS (Root-Mean-Square) of the difference between the actual state and the desired state as well as average power usage. The model-free algorithm outperformed the PID controller in all simulations with the quadcopter's original parameters, double the mass, double the moments of inertia, and double both the mass and the moments of inertia while keep both controllers exactly the same for each simulation.

Table of Contents

List of Figures

List of Tables

NOMENCLATURE

UAS	Unmanned Aerial System
UAV	Unmanned Aerial Vehicle
SMC	Sliding Mode Control
MFSMC	Model-Free Sliding Mode Control
PID	Proportional – Integral – Derivative
NED	North – East – Down
$[B]$	Control Input Matrix
s	Sliding Surface
λ	Slope of the Sliding Surface
K	Switching Gain
Φ	Boundary Layer
h	Altitude
ϕ	Roll Angle
θ	Pitch Angle
ψ	Yaw/Heading Angle
u	Control Input

1.0 INTRODUCTION

System automation is becoming more popular and versatile every day leading to a tremendous rise in the field of control systems. Traditional control methods such as Proportional-Integral-Derivative (PID) control is considered the most popular algorithm used to drive system states by compensating the errors. However, PID control and other traditional control schemes require a known system model for tuning along with limitations in linearizing system models affecting overall performance.

Due to its robustness the Sliding Mode Control (SMC) is used to compensate for systems with modelling uncertainties and is directly applicable for both linear and nonlinear systems. The SMC method theory transforms the control system into a set of 1^{st}-order problems which is easier to control obtaining better overall tracking performance. The method has two phases, a reaching phase and a sliding phase. The reaching phase drives the system towards the "sliding phase" and the sliding phase slides the states towards equilibrium. Asymptotic tracking stability is ensured through Lyapunov's Direct Method when the state trajectories are not on the sliding surface. A discontinuous term is added to the control law to compensate for system uncertainties and disturbances. However, the traditional SMC algorithm requires knowledge of the mathematical model of the system and hence it is unique to each system which restricts its use and time consuming to derive and implement.

There is a clear need (and advantage) to develop a model-free sliding mode controller which is based on previous control inputs, system measurements, control input gains and the system's order with no knowledge of the system model. Reis and Crassidis [1] had developed a model-free sliding model controller for a Single-Input-Single-Output (SISO) linear and nonlinear 1^{st} and

2^{nd}-order system for a unitary and non-unitary control input gain. Simulations were performed indicating perfect tracking was achieved while the sliding condition was also satisfied. Similar investigation was conducted on squared and non-squared Multi-Input-Multi-Output (MIMO) 1^{st} and 2^{nd}-order systems by El Tin and Crassidis [2]. Simulation results showed perfect system tracking was achieved for squared MIMO 1^{st} and 2^{nd}-order type problems. In addition, for non–squared MIMO systems the states which were weighted more exhibited better perfect tracking performance compared to the stated weighted less. The sliding condition was satisfied for both cases proving tracking stability was achieved throughout.

Sreeraj and Crassidis [3] developed a model-free algorithm based on sliding mode control and implemented it successfully for controlling pitch and roll states on a quadrotor mounted to a gimbal. There is a need to implement this algorithm to an Unmanned Aircraft System (UAS) that is not mounted to a gimbal controlling altitude and is allowed to fly in free-flight which is the overall goal of this proposed work.

2.0 LITERATURE REVIEW

Sliding mode control has become popular and has received attention in the last few years due to its robustness and its ability to handle system modeling uncertainties and disturbances. The SMC method has successfully been applied to robotic manipulators, power systems, Unmanned Aerial Vehicles (UAVs), and other sophisticated systems. Different approaches have been developed for the SMC method, but the main principle remains requiring a system model for its development.

2.1 Generic Sliding Mode Control Schemes

Laghrouche et al. [4] introduced a higher-order sliding mode controller on optimal linear quadratic control applied to a minimum-phase nonlinear SISO systems. The problem was divided into three steps. Firstly, a higher order sliding mode problem was created to eliminate the chattering effect, followed by characterizing the nonlinear uncertainties as bounded non-structured parametric uncertainties considering the system as an uncertain linear system. Lastly am optimal sliding mode controller is derived by minimizing a quadratic cost function over finite amount of time. This SMC was tested on a kinematic model of an automobile to steer it from an initial position to a trajectory defined by the user. A sliding mode control of the fourth order was used with a time varying sliding surface. The system achieved perfect tracking with the error converging to zero with no chattering.

Runcharoon and Srichatrapimuk [5] presented the altitude control of a quadrotor using a SMC system. The Euler angles was used to define the altitude (φ = roll, θ = pitch, ψ = yaw) and describe the orientation of the quadrotor. A PD controller was used to control the altitude (z) and the position (x,y) while the equations characterizing the position and altitude wee linearized to quantify the PD control gains. Euler angle assumptions $\varphi \approx \psi \approx 0$ on the x- axis, $\theta \approx \psi \approx 0$ on the

y-axis and $\varphi \approx \theta \approx 0$ were applied. A boundary layer was added to the dynamic equations eliminate the chattering about the sliding surface. The simulation was able to drive the quadrotor to the desired position and desired orientation and prove the stability of the system and the control inputs.

Runcharoon and Srichatrapimuk [5] used a SMC and PD to achieve a stable system. It showed an improvement compared to a general PID control system, but it's limited the full potential of a SMC which does not required linearization. The presence of under-actuated systems led to the use of two different control methods as it requires manipulation if used with the SMC process.

Sen et al. [6] introduced an adaptive method SMC for quadrotor helicopters and dealt with the estimation of the system uncertainties and perturbation bounds. As these bounds are unknown, they are overestimated thus leading to excessive gain. This gain is directly proportional to the magnitude of chattering, and hence by estimating these bounds and updating the control law, this is magnitude is reduced. The method was tested on a quadrotor helicopter type application. A stable closed-loop system with perfect position tracking with no chattering was achieved. The uncertainties bounds were unknown, which were later estimated.

Xu and Ozguner [7] presented a method to stabilize underactuated systems using SMC. The authors used the method proposed by Olfati – Saber [8] to transform the system into a cascade normal form. Xu and Ozguner [9] applied this method on two nonlinear underactuated MIMI systems: 1. Translational Oscillator with Rotational Actuator (TORA) and 2. a quadrotor UAV. The quadrotor system was similar to the one used by Runcharoon and Srichatrapimuk [5] while the TORA was controlled using a rate bounded PID controller and a sliding mode controller [9]. A simulation proved a stable system and convergence to the desired position.

Pai [10] demonstrated that the SMC showed robust tracking in discrete time systems by applying a discrete-time integral SMC on uncertain linear systems. An auxiliary control function was introduced to define the discrete-time sliding mode controller to stabilize the system. The switching surface was designed by extending the integral switching function from continuous to discrete time SMC ensuring that a quasi-sliding mode was reached [11, 12]. In practice only the switching surface is approached by discrete-time SMC systems and hence quasi-sliding mode is assured. The method was applied to a discrete-time system and it showed excellent tracking performance with the presence of uncertainties along with stability of the closed loop system. The integral switching surface in the design process eliminated the reaching phase and chattering was absent as due to the absence of a switching gain.

Lee [13] also introduced a discrete-time SMC using a fast output sampling. In this paper the system's closed loop eigenvalues were arbitrarily defined while designing the control system focusing on stability and transient response. A boundary layer was introduced in the control law to reduce the chattering effect. The approach was tested on a discrete-time controller for a continuous time plant model with a serial type lightly damped resonance. Outstanding step response tracking was achieved which eventually proved that the system's closed loop eigenvalues can be arbitrarily assigned.

Ferrara et al. [14] presented the problem of applying SMC in systems with saturating actuators. A sub-optimal sliding mode controller with modifications was used to avoid control input saturation. The problem was the uncertainty in convergence of the sliding variable to zero in a finite amount of time when saturation occurs during reaching phase. The proposed modification decreases the control input once it reaches the saturation value (reaching phase). The control input increases again if the switching value was not reached implying the control input remains

at the saturation value until a new switching value is reached. The work proved the system states converged to the origin in finite time and was verified with a simulation example while avoiding the saturation limits.

2.2 Model-Free Sliding Mode Control Schemes

As the dynamical systems become exceeding complex, a significant advantage is gained in developing a model-free approach in the design of a sliding mode control law both in development time and control law efficiency.

Martinez-Guerra et al. [15] presented a Sliding Mode Observer (SMO) referred to as master-slave synchronization to determine certain synchronization problem with chaotic behavior. The method required an accurate knowledge of the nonlinear system dynamics. Therefore, a model-free SMO with a promotional correction of the sign function of the synchronization error was introduced. The method was successfully applied to the Lorenz system, a nonlinear system exhibiting chaotic behavior when tuned to certain gains.

Salgado-Jimenez et al. [16] applied a model-free higher-order SMC on a one degree-of-freedom underwater vehicle for position control. The new method used only the exponential convergence of the desired trajectory, eliminating the need of knowledge of the system dynamics or parameters. Chattering was avoided to restrict damage to the actuators lifetime by using a higher-order SMC. However, the controller is integrated to a PD control scheme whose desired gain values and performances are tuned. Two trajectories were tested: 1. Sine Wave and, 2. Triangular Wave. A smooth response was achieved in both cases while the vehicle followed the desired path.

Raygosa-Barahona et al. [17] introduced a model-free backstepping technique with integral SMC to develop a model-free SMC system for an underactuated underwater Remotely Operated Vehicle (ROV).A model-free controller was obtained by designing a regressor free 2nd-order sliding mode controller as the auxiliary input control at each iteration. The sliding mode is integrated with a PID control and is applied to a ROV to track a helix trajectory. However, the PID controller needs to be tuned in order to achieve the desired performance. The vehicle converged to the desired trajectory with no chattering.

Munoz-Vazquez et al. [18] introduced a method to control the position of a quadrotor using passive Velocity Field (VF) navigation with unknown system dynamics. The controller has three subsystems: 1. model-free control subsystem – responsible for maintaining the sliding mode condition at all time, 2. VF subsystem – responsible for designing the velocity field to define the desired path, and 3. sliding surface subsystem – responsible for assembling invariant manifolds of position and orientating sliding surfaces. The controller was tested in a 3D environment without and with obstacles to prove the effectiveness of the VF to navigate around the obstacles in cluttered environments. The system displayed perfect tracking with no chattering however, the VF needed to be designed in order to derive the controller scheme.

Crassidis and Mizov [19] [20] presented a model-free pure sliding mode control scheme to achieve perfect tracking for linear and nonlinear systems along with asymptotic stability. The controller is designed based on previous control inputs, state measurements and the knowledge of the system order. A boundary layer was introduced into the control law to remove the chattering effect. This reduced the tracking precision but gave a smooth control effort which is required. The method was tested on a 1st and 2nd-order linear and nonlinear system. All systems tested in a simulation effort showed perfect tracking and asymptotic stability was also observed.

Reis and Crassidis [1] extended the previous model-free SMC approach to SISO application systems with non-unitary input gains. The effect of noise on system on system inaccuracies was also investigated. Firstly, a nonlinear mass-spring-damper system with non-unitary control input gain without sensor noise was simulated followed by a state measurement noise using a Gaussian distribution of noise. The variance, mean and probability distribution were obtained from the sensor's datasheet. Perfect tracking was obtained in both cases and chattering was eliminated by using a boundary layer.

El Tin and Crassidis [2] further extended the model-free SMC method into MIMO systems and examined the effects of an actuator-induced time delay. The derivation and implementation for square MIMO system was similar to the approach proposed in Reis and Crassidis [1]. For an underactuated MIMO system a transformation matrix was introduced to square the control input gain matrix to derive the control law. Perfect tracking was achieved for squared MIMO systems while only certain outputs achieved perfect tracking in the underactuated cases. The transformation matrix allowed the control of the output for tracking. The method was further applied to a single input nonlinear two mass-spring-damper type system and quadrotor. The first system achieved perfect tracking on all states with the control effort maximized, but the latter observed perfect tracking on certain outputs while the control efforts and certain outputs displayed high frequency activity. This was due to the aggressiveness of the controller to track the required trajectory entirely. The presence of an actuator time delay had an adverse effect when it exceeded a certain value, hence the control law required to be modified to account for the presence of this time delay.

Levant [21] also presented a unique method of model-free sliding mode control based on Higher Order Slide Mode [HOSM] theory. The controller form is based usually on an insignificant relay

controller where the sliding surface satisfies the higher order sliding modes. No information about the plant and only the relative order is required for the controller since the sliding surface and the sliding surface derivatives are based only on the states. The control effort is calculated by integration eliminating the chattering effect without the need for a smoothing step. The method eliminates chattering in the ideal scenario, but chattering might still occur due to the excitation of parasitic dynamics [22]. The parasitic dynamics (such as actuator or sensor delays) were assumed as unmodeled dynamics. The control law was used to steer a four-wheeled vehicle onto a desired trajectory.

Precup [23] developed two distinct methods of model-free sliding mode control based on dynamic data-driven linear estimation of the system model. For a first-order system, the sliding surface is defined as:

$$S = \tilde{x}(t) + \int_0^t \tilde{x}(\tau)d\tau \tag{2.1}$$

where S is the sliding surface, $\tilde{x}(t)$ is the difference between the actual state and the desired state as a function of time t, and $\int_0^t \tilde{x}(\tau)d\tau$ is the integral of the difference between the actual state and the desired state over a time t. The system model is assumed to be:

$$\dot{y} = F(t) + \alpha u(t) \tag{2.2}$$

where $\dot{y}$ is the state derivative, $u(t)$ is the input, α is the tunable parameter and is selected to keep the magnitude of $\dot{y}$ and $u(t)$ the same, and $F(t)$ is the system matrix and is approximated to:

$$\hat{F}(t) = \hat{y}(t) - u(t) \tag{2.3}$$

where $\hat{y}(t)$ is the state estimate and is determined from y by a first order derivative and a low-pass filter. Replacing the discontinuous sign function and adding a thickness boundary layer and a saturation function, the SMC control law is

$$u = \alpha^{-1}(-\hat{F}(t) + \dot{x}_d(t) - \lambda^{-1}\tilde{x}(t) - e_{est\,max} - \lambda^{-1}sat\left(\frac{S}{\varphi}\right)) \tag{2.4}$$

where $\dot{x}_d(t)$ is the desired state, λ is the slope of the sliding surface, φ is the boundary layer thickness, and $e_{est\,max}$ is a maximum error estimate which satisfies the inequality

$$\varphi^{-1}|S(t)|\eta > 2\lambda e_{est\,max} \tag{2.5}$$

where η is a small positive constant.

The method is similar to those used in [1, 2, 20] as it is the adaption of conventional SMC with the system model estimated from only the states and inputs. However, in this method the algebraic loops are avoided using a differentiator rather than a direct state measurement.

3.3 Sliding Mode Control for Underactuated Systems

Many practical applications require control of MIMO systems and in this work using the SMC method. Dehghani and Menhaj [24] developed a state-space model for a leader-follower system which omits the effects of flight dynamics. The control inputs are considered translational acceleration in three dimensions. This removed the problem of cross-coupling of the control effort and the underactuated nature of a conventional aircraft wing. However, resolving problems arising from developing control laws for underactuated MIMO systems is an important aspect.

Qian, Yi and Zhan [25] developed a multi-surface SMC for a single-input-multi-output underactuated system. The method was based on nested sliding variables including all system states. The number of sliding surfaces depended on the number of states. It allows tracking of

multiple outputs with a single input. Tunable coefficients were weighted to the outputs to lower leveled sliding surface when constructing higher leveled sliding surfaces. The method was validated by simulation on a single and double inverted pendulum for stabilization. The effect for chattering was not mentioned in this work.

Schkoda [26] developed a squaring transformation matrix using optimal control theory to converge on the weighting of different outputs in the transformation. The method is similar to the one developed by Raygosa in [16] where virtual control inputs were mapped to the actual control inputs through a transformation matrix. The transformation matrix in [26] leads to a square input matric which is inverted to derive the SMC control law for MIMO systems.

One major area of application of underactuated systems is in unmanned aircraft systems. There are two main configurations of for UASs: fixed wing and quadrotor. Fixed wing UAS is similar to a conventional aircraft. The four control inputs are rudder (vertical tail), elevator (horizontal tail), ailerons and forward thrust. A quadrotor is a type of helicopter with four equal sized rotors distributed equally in horizontal plane around the center of mass. Typical rotors use rotors with fixed blade pitch. The next section reviews some methods used to develop SMC for UAS addressing the issue of under actuation.

2.4 Sliding Mode Control for Unmanned Aircraft Systems

Norton et al. [27] developed a fixed wing UAS system with 12 state outputs but only 4 inputs (rudder, elevator, aileron deflections and thrust). The under actuation is eliminated by applying a diffeomorphism to the differential equations of the systems. After coordinate transformations, the differential equations are given as four three dimensional equations z_i, with four sliding variables S_i defined as

$$S_i = z_i - z_{i_d}\ i = 1, 2, 3, 4 \tag{2.6}$$

Here, S_i is the difference between the actual state and the desired state, z_i is the current trajectory in the transformed coordinates, and z_{i_d} is the desired trajectory to be tracked in the transformed coordinates. Like classic SMC, the sliding variables are differentiated and substituted in the system model. The control laws are developed for thrust and surface deflections along with three virtual controllers to compensate modal uncertainties.

Abulahamitbhai [28] also developed a six (Degree-of-Freedom) DOF state space model for a fixed wing UAS with 12 states. Unlike assigning a sliding surface to each state in [12], only four sliding surfaces are developed, one for each input (rudder, elevator, aileron deflections and thrust). The two-6-dimensional state variables (position and velocity) are transformed into a 4-dimensional space with the sliding surface. Transformation matrix using weights for individual states like the one used in [27] is used.

Duan, Mora-Camino and Miquel [29] compared the performance of a decoupled longitudinal fixed-wing UAS model using dynamic inversion and backstepping methods. A full 6 DOP aircraft model with actuator dynamics was simulated but only the longitudinal results were examined. The backstepping method gave smoother responses but the control law was very complex.

Brezoescu, Lozano and Casillo [30] worked on the tracking control in the lateral direction of a fixed wing aircraft. The single input was the derivative of the yaw rate. The outputs were yaw (heading) angle and orthogonal distance from the required path. Even though the system expressed was underactuated, the control law derived able to regulate both outputs effectively.

Villanueva et al. [31] developed a 6 DOF state-space model for a quadrotor with 12 states. Four different control modes (manual, altitude hold, position hold and waypoint following) was derived using the super twisting method of SMC. The under actuation of the quadrotor was resolved by adding a pseudo-control inputs to roll and pitch that are dependent on positions in the horizontal plane, which was the same method used by Munoz-Vazquez et al. in [17] where pitch and roll were removed as the explicit outputs of the system and the remaining 4 states and 4 inputs make a fully actuated system. The method is different to what was done in [2] where all 6 DOF were retained and a transformation matrix was used for the underactuated system using tracking weights.

Derafa, Benallegue and Fridman [32] also applied super twisting on a quadrotor and tested the system in the real world. The controller was developed for attitude tracking and stabilization. The desired values are given as functions of desired positions. The system modelled in this method becomes a fully actuated system.

Sreeraj and Crassidis [3] applied a model-free algorithm based on sliding mode control on a hardware quadcopter mounted on a gimbal controlling pitch and roll, while thrust and yaw were controlled using tradition PID. It was concluded that the model-free algorithm provided better tracking while also consuming less power than the traditional PID.

3.0 MODEL-FREE SLIDING MODE CONTROL DERIVATION AND APPLICATION

The Model-Free Sliding Mode Control (MFSMC) algorithm is derived in this section. The algorithm is then applied to a 2nd-order linear SISO system and a 2nd-order nonlinear square system.

3.1 Model-Free Sliding Mode Control Algorithm

The system description of an n^{th}-order MIMO autonomous system is given by:

$$\vec{x}_p^n = \vec{f}_p\left(\vec{x}_p^n\right) + [B]_{p\times m}\vec{u}_m \tag{3.1}$$

The system description can be rewritten as:

$$\vec{x}_p^n = \vec{x}_p^n + [B]\vec{u}_m - \vec{u}_{m_{k-1}} - [B]\vec{u}_m + [B]\vec{u}_{m_{k-1}} \tag{3.2}$$

where p and m are the number of inputs and output respectively, $\vec{x}_p^n$ is the system states, $[B]$ is the control input gains, and $\vec{u}_{m_{k-1}}$ is the previous control input to the system. The error between the current and previous control input is defined as:

$$\vec{\varepsilon}_m = \vec{u}_{m_{k-1}} - \vec{u}_{m_k} \tag{3.3}$$

Implementation of Eq. (3.3) would cause an algebraic loop in the algorithm. To remove the algebraic loop a control input error estimation is implemented and is defined as:

$$\hat{\vec{\varepsilon}}_m = \vec{u}_{m_{k-2}} - \vec{u}_{m_{k-1}} \tag{3.4}$$

where $\vec{u}_{m_{k-2}}$ is the second previous control input. The control input error estimation is assumed to be within the bounds of:

$$(1-\sigma_l)\hat{\vec{\varepsilon}}_m \le \vec{\varepsilon}_m \le (1+\sigma_u)\hat{\vec{\varepsilon}}_m \tag{3.5}$$

where σ_l is the lower bound and σ_u is the upper bound of the control input estimation error. The control input gain $[B]$ is assumed to be:

$$b_{ij_{lower}} \leq b_{ij} \leq b_{ij_{upper}} \tag{3.6}$$

were $b_{ij_{lower}}$ is the lower bound of the ij^{th} entry, b_{ij} is the true value of the ij^{th} entry, and $b_{ij_{upper}}$ is the upper bound of the ij^{th} entry. For systems that are 1st-order the sliding surface can be defined as:

$$\vec{s} = \tilde{\vec{x}} + \lambda \int_0^t \tilde{\vec{x}}(\tau) \cdot d\tau \tag{3.7}$$

where $\vec{s}$ represents the sliding surface, $\tilde{\vec{x}}$ represents the difference between the actual state and the desired state, λ represents the slope of the sliding surface, and *t* represents time. For systems that are 2nd-order or higher can be defined as:

$$\vec{s} = (\frac{d}{dt} + \lambda)^{n-1} \tilde{\vec{x}} \tag{3.8}$$

For a 2nd-order system eq. (3.8) becomes:

$$\vec{s} = \dot{\tilde{\vec{x}}} + \lambda \tilde{\vec{x}} \tag{3.9}$$

where $\dot{\tilde{\vec{x}}}$ represents the difference between the derivative of the actual state and the derivative of the desired state. To ensure the states remain on the sliding surface the derivative of the sliding surface is set to zero:

$$\dot{\vec{s}} = \ddot{\tilde{\vec{x}}} + \lambda \dot{\tilde{\vec{x}}} = 0 \tag{3.10}$$

where $\dot{\vec{s}}$ represents the derivative of the sliding surface, and $\dddot{\vec{x}}$ represents the difference between the second derivative of the actual state and the second derivative of the desired state.

Substituting Eq. (3.4) into Eq. (3.2):

$$\vec{x}_p^n = \vec{x}_p^n + [B]\vec{u}_m - [B]\vec{u}_{m_{k-1}} + [B]\vec{\varepsilon} \tag{3.11}$$

substituting Eq. (3.11) into Eq. (3.10):

$$\dot{\vec{s}} = \dddot{\vec{x}} + \lambda\ddot{\vec{x}} + [B]\vec{u} - [B]\vec{u}_{k-1} + [B]\vec{\varepsilon} = 0 \tag{3.12}$$

solving for $\vec{u}$ and adding a discontinuous term to handle model uncertainties results in:

$$\vec{u} = [B]^{-1}\left[\ddot{\vec{x}} + \lambda\dot{\vec{x}} + \eta \cdot sgn(\vec{s})\right] + \vec{u}_{k-1} - \vec{\varepsilon} \tag{3.13}$$

3.2 Lyapunov's Direct Method

Lyapunov's Direct Method is used to ensure asymptotic stability during the reaching phase. A candidate Lyapunov function is one that is positive definite and can be defined as:

$$\vec{V}(\vec{x}) = \frac{1}{2}\vec{s}^2 \tag{3.14}$$

where $\vec{V}(\vec{x})$ is the Lyapunov function. Differentiating the Lyapunov function results in:

$$\dot{\vec{V}}(\vec{x}) = \vec{s}\dot{\vec{s}} \tag{3.15}$$

substituting Eq. (3.12) into Eq. (3.15) and setting it to be negative definite to ensure global asymptotic stability results in:

$$\dot{\vec{V}}(\vec{x}) = \vec{s}\left(\ddot{\vec{x}} + \lambda\dot{\vec{x}} + [B]\vec{u} - [B]\vec{u}_{k-1} + [B]\vec{\varepsilon}\right) \leq 0 \tag{3.16}$$

substituting Eq. (3.13) into Eq. (3.16) results in:

$$\dot{\vec{V}}(\vec{x}) = \vec{s}\left(\ddot{\vec{x}} + \lambda\dot{\vec{x}} + [B]\left(-[B]^{-1}\left[\ddot{\vec{x}} + \lambda\dot{\vec{x}} + \eta \cdot sgn(\vec{s})\right] + \vec{u}_{k-1} - \vec{\varepsilon}\right) - [B]\vec{u}_{k-1} + [B]\vec{\varepsilon}\right) \leq 0$$

(3.17)

which simplifies to:

$$\dot{\vec{V}}(\vec{x}) = \vec{s}\left(-\eta \cdot sgn(\vec{s})\right) = -\eta|\vec{s}| \leq 0 \tag{3.18}$$

introducing a switching gain $\vec{K}$ into Eq. (3.13) ensures that the state trajectories remain asymptotically stable during the reaching phase and results in:

$$\vec{u} = -[B]^{-1}\left[\ddot{\vec{x}} + \lambda\dot{\vec{x}} + \vec{K}sgn(\vec{s})\right] + \vec{u}_{k-1} - \vec{\varepsilon} \tag{3.19}$$

since $[B]$ and $\vec{\varepsilon}$ are not known exactly they're substituted with $[\hat{B}]$ and $\hat{\vec{\varepsilon}}$, where these are estimated values, so Eq. (3.19) becomes:

$$\vec{u} = -[\hat{B}]^{-1}\left[\ddot{\vec{x}} + \lambda\dot{\vec{x}} + \vec{K}sgn(\vec{s})\right] + \vec{u}_{k-1} - \hat{\vec{\varepsilon}} \tag{3.20}$$

substituting Eq. (3.4) into Eq. (3.20) results in:

$$\vec{u} = -[\hat{B}]^{-1}\left[\ddot{\vec{x}} + \lambda\dot{\vec{x}} + \vec{K}sgn(\vec{s})\right] + 2\vec{u}_{k-1} - \vec{u}_{k-2} \tag{3.21}$$

$\left[\hat{B}\right]$ is determined by calculating the geometric mean of the upper and lower bounds:

$$\left[\hat{B}\right] = ([B]_{upper}[B]_{lower})^{\frac{1}{2}} \tag{3.22}$$

and the auxiliary variable $[\beta]$ is defined as:

$$[\beta] = [B]_{upper}[\hat{B}]^{-1} \tag{3.23}$$

using the sliding condition shown in Eq. (3.21) asymptotic stability is ensured if:

$$\vec{s}\dot{\vec{s}} \leq -\eta|\vec{s}| \tag{3.24}$$

The upper bounds of the estimations made are assumed to be conservative. Solving for the most extreme case of the inequality in Eq. (3.24) results in the switching gain being:

$$\vec{K} = \left|\dddot{\vec{x}}\right| |[\beta] - [I]| + \lambda \left|\ddot{\vec{x}}\right| |[\beta] - [I]| + \left|\left[\hat{B}\right]\sigma_u(\vec{u}_{k-2} - \vec{u}_{k-1})\right| + [\beta]\eta \tag{3.25}$$

Implementing the current presented control law shown in Eq. (3.21) produces high frequency chattering on the output from the controller which can damage actuators on real-world systems. To alleviate the chattering, a boundary layer is introduced to act as a 1st-order filter with no phase loss:

$$\vec{u} = -[\hat{B}]^{-1}\left[\dddot{\vec{x}} + \lambda\ddot{\vec{x}} + (\vec{K} - \dot{\vec{\Phi}})sat\left(\frac{\vec{s}}{\vec{\Phi}}\right)\right] + 2\vec{u}_{k-1} - \vec{u}_{k-2} \tag{3.26}$$

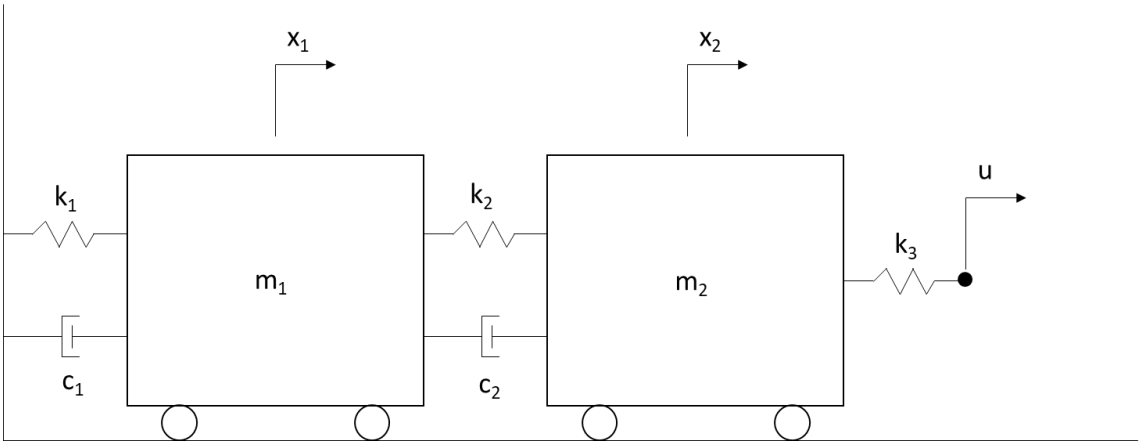
where $\vec{\Phi}$ represents the boundary layer and $\dot{\vec{\Phi}}$ represents the derivative of the boundary layer and is described by:

$$\dot{\vec{\Phi}} + \lambda\vec{\Phi} = \vec{K}(\vec{x}_d) \tag{3.27}$$

where $\vec{x}_d$ represents the desired state. The initial condition for Eq. (3.27) is:

$$\vec{\Phi}(0) = \frac{\eta}{\lambda} \tag{3.28}$$

3.3 Example on a 2nd-order Linear SISO System



The system shown above is a two-mass system connected by springs *k* and dampers *c*, and *u* represents a force input on a spring connected to m_2. The system is represented by the following equations of motion:

$$m_1\ddot{x}_1 + c_1\dot{x}_1 + k_1x_1 + c_2(\dot{x}_1 - \dot{x}_2) + k_2(x_1 - x_2) = 0 \tag{3.29}$$

$$m_2\ddot{x}_2 + c_2(\dot{x}_2 - \dot{x}_1) + k_2(x_2 - x_1) + k_3x_2 = k_3u \tag{3.30}$$

where:

$$m_1 = 5\ kg \quad m_2 = 20\ kg \quad c_1 = 8\frac{Ns}{m} \quad c_2 = 20\frac{Ns}{m} \quad k_1 = 50\frac{N}{m} \quad k_2 = 80\frac{N}{m} \quad k_3 = 150\frac{N}{m}$$

it is assumed that m_2 is within the bounds of:

$$15 \leq m_2 \leq 25\ kg$$

with a desired tracking signal of:

$$x_{2_d}(t) = \sin\left(\frac{\pi}{2}t\right) \tag{3.31}$$

The controller parameters used are:

σ	0.65
η	0.2
λ	20

Table 1: Controller Parameters for the 2nd-order Linear SISO System

Using the control law derived in Eq. (3.26) results in the following performance:

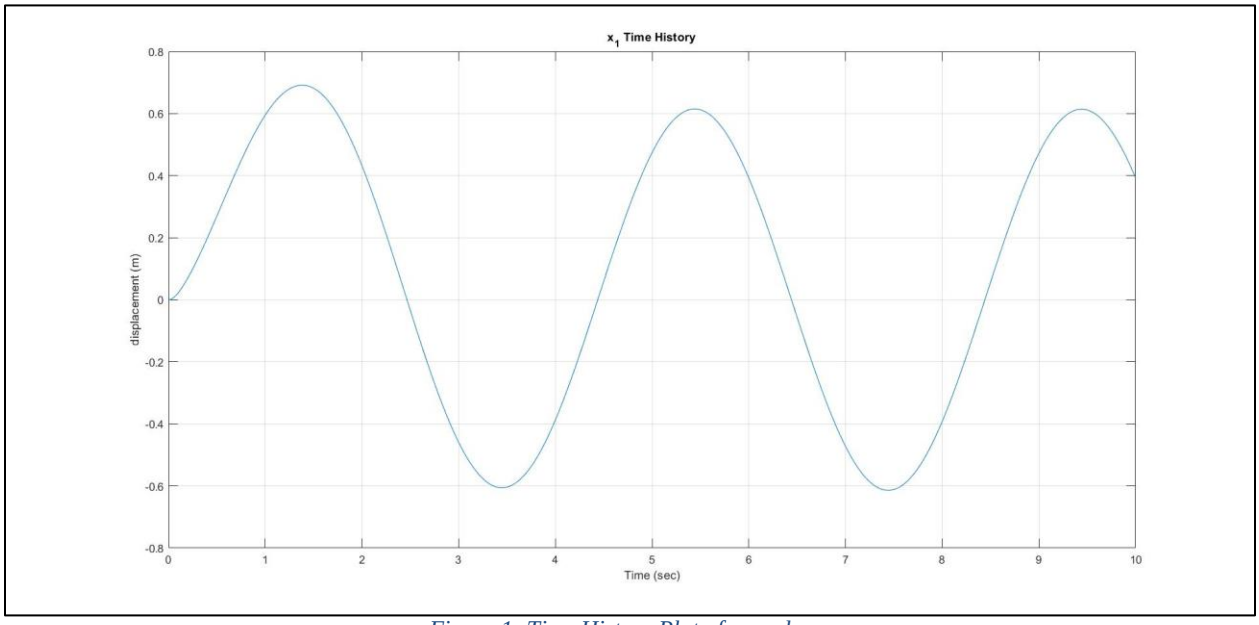


Figure 1: Time History Plot of x_2 and x_{2d}

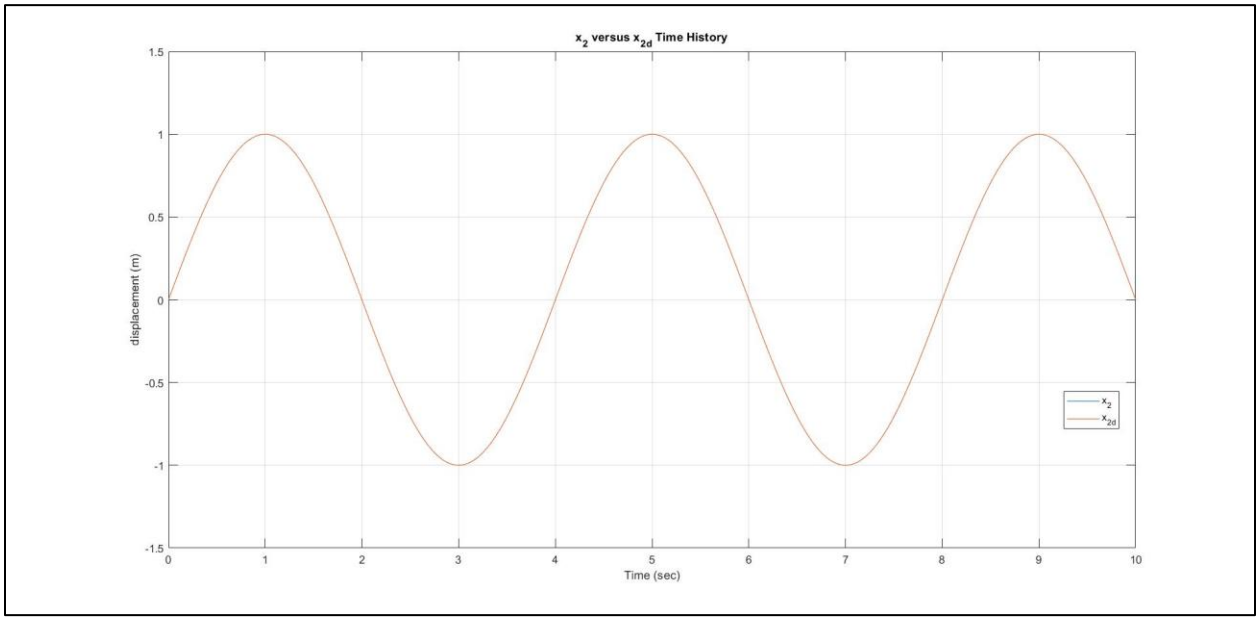


Figure 2: Time History Plot of x_1

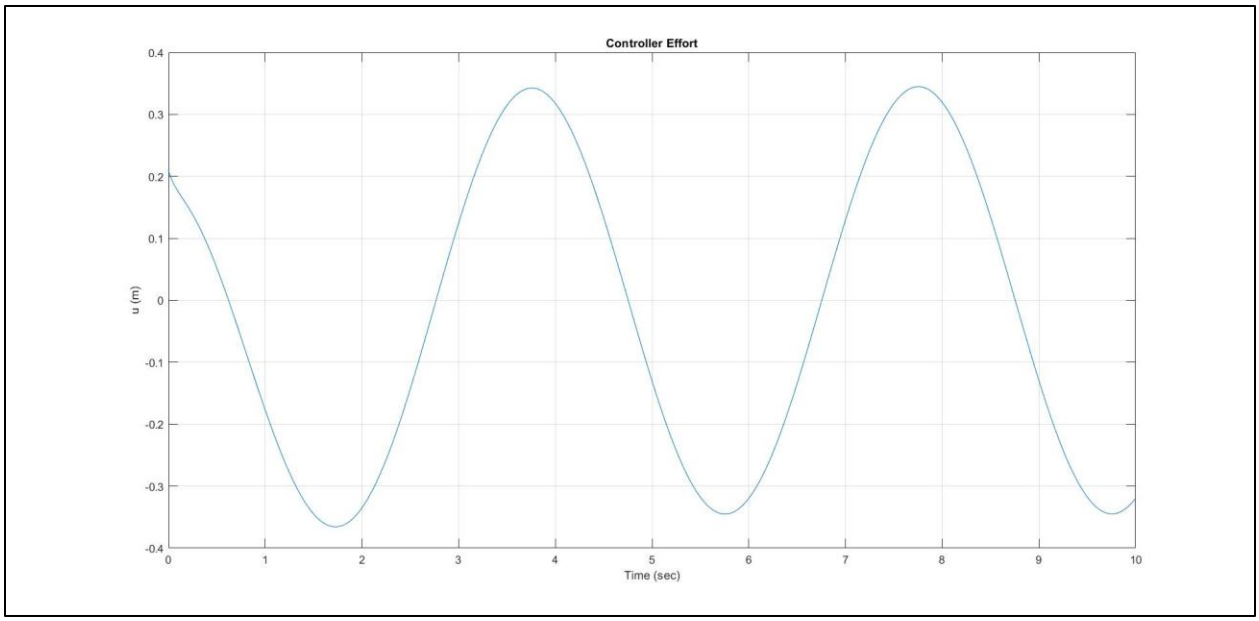


Figure 3: Time History Plot of the Controller Effort

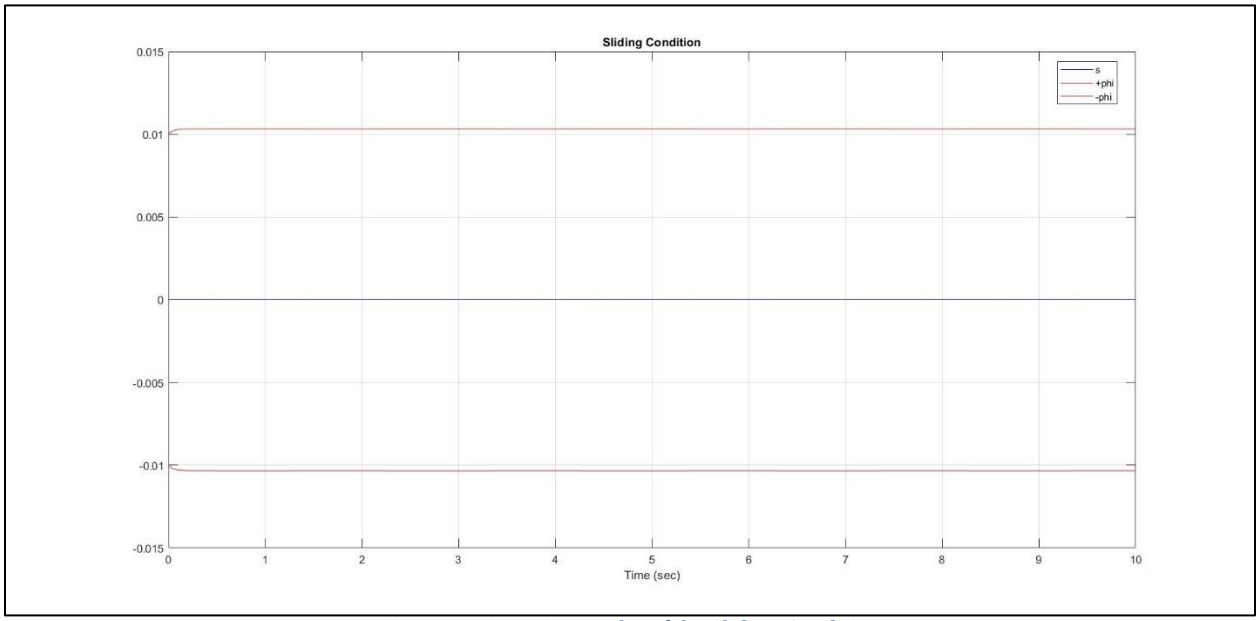
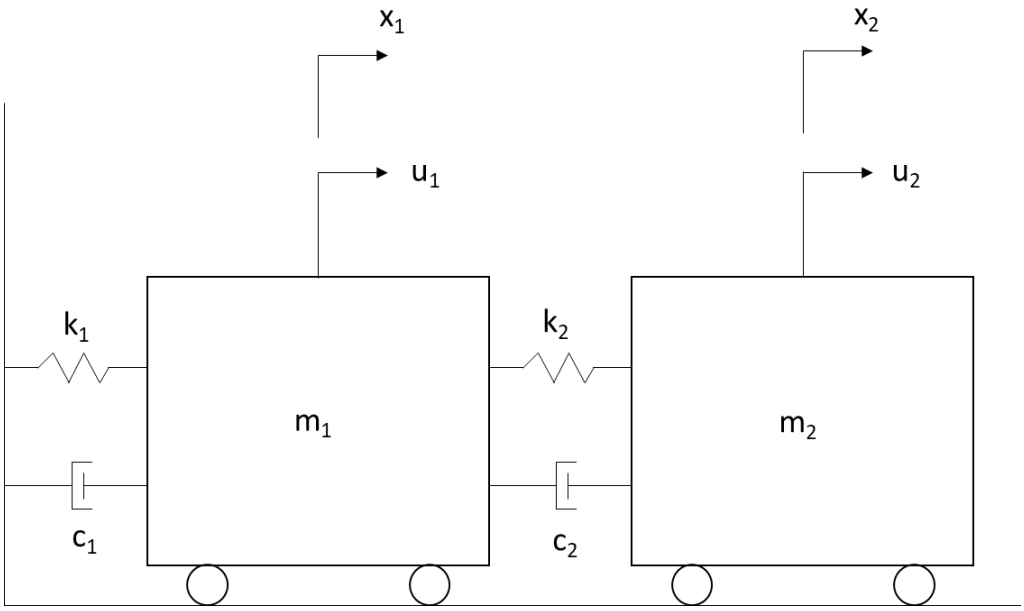


Figure 4: Time History Plot of the Sliding Condition

The RMS of the difference between the actual and desired state, from Figure 1, resulted in $2.3844*10^{-8}$ meters. This RMS value shows that almost perfect tracking is achieved. The sliding condition, shown in Figure 4, is observed to be satisfied for the whole simulation which guarantees asymptotic stability and the controller effort, shown in Figure 3, is observed to be smooth without high frequency chattering.

3.4 Example on a 2nd-order Nonlinear Square MIMO System



The system shown above is a two-mass system connected by nonlinear springs *k* and nonlinear dampers *c*, and *u* represents a force input on each mass. The system is represented by the following equations of motion:

$$m_1\ddot{x}_1 + c_1\dot{x}_1|\dot{x}_1| + k_1x_1^3 + c_2(\dot{x}_1 - \dot{x}_2)|\dot{x}_1 - \dot{x}_2| + k_2(x_1 - x_2)^3 = u_1 \tag{3.32}$$

$$m_2\ddot{x}_2 + c_2(\dot{x}_2 - \dot{x}_1)|\dot{x}_2 - \dot{x}_1| + k_2(x_2 - x_1)^3 = u_2 \tag{3.33}$$

where:

$$m_1 = 10\ kg \quad m_2 = 20\ kg \quad c_1 = 20\frac{Ns^2}{m^2} \quad c_2 = 45\frac{Ns^2}{m^2} \quad k_1 = 120\frac{N}{m^3} \quad k_2 = 80\frac{N}{m^3}$$

it is assumed that m_1 and m_2 are within the bounds of:

$$5 \leq m_1 \leq 15\ kg$$

$$15 \leq m_2 \leq 25\ kg$$

with a desired tracking signal of:

$$x_{1_d}(t) = \sin\left(\frac{\pi}{2}t\right) \tag{3.34}$$

$$x_{2_d}(t) = \sin\left(\frac{\pi}{2}t\right) \tag{3.35}$$

The controller parameters used are:

	m_1	m_2
σ_u	0.65	0.65
λ	20	20
η	0.2	0.2

Table 2: Controller Parameters for the 2nd-order Nonlinear Square MIMO System

Using the control law derived in Eq. (3.26) results in the following performance:

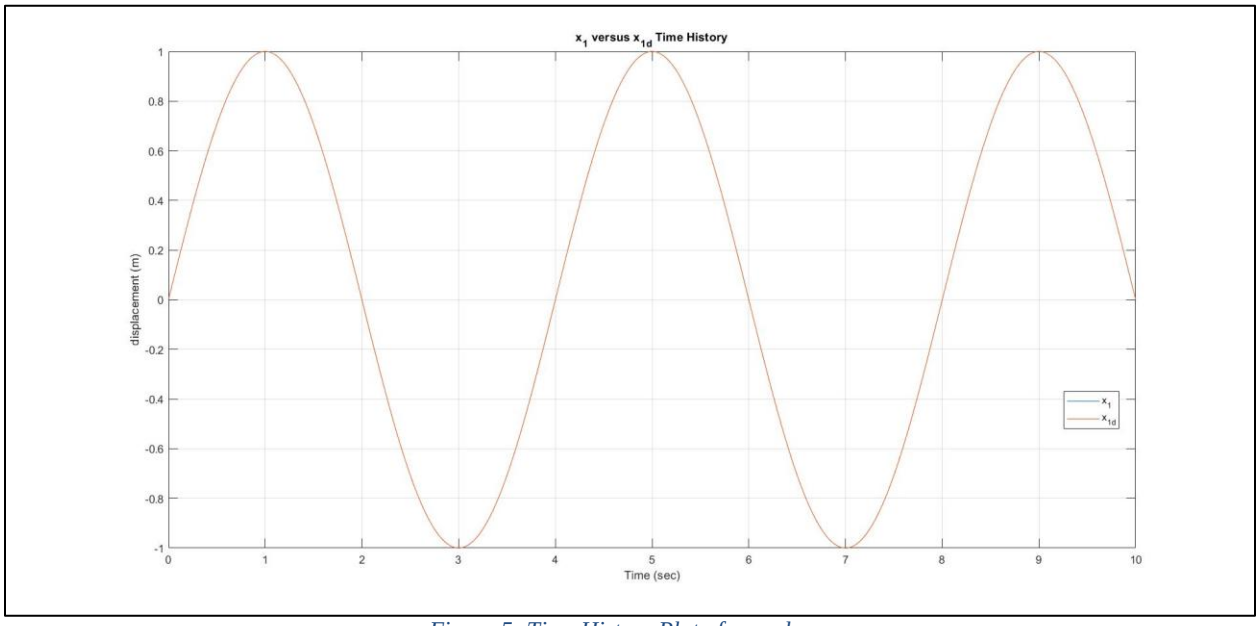


Figure 5: Time History Plot of x_1 and x_{1d}

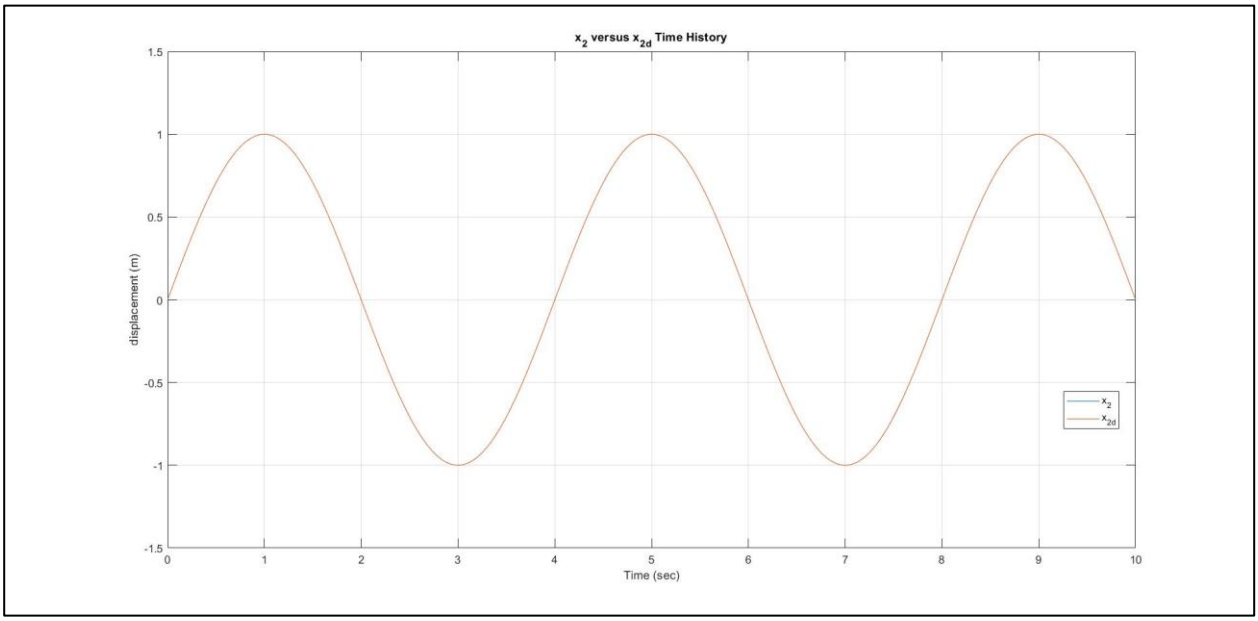


Figure 6: Time History Plot of x_2 and x_{2d}

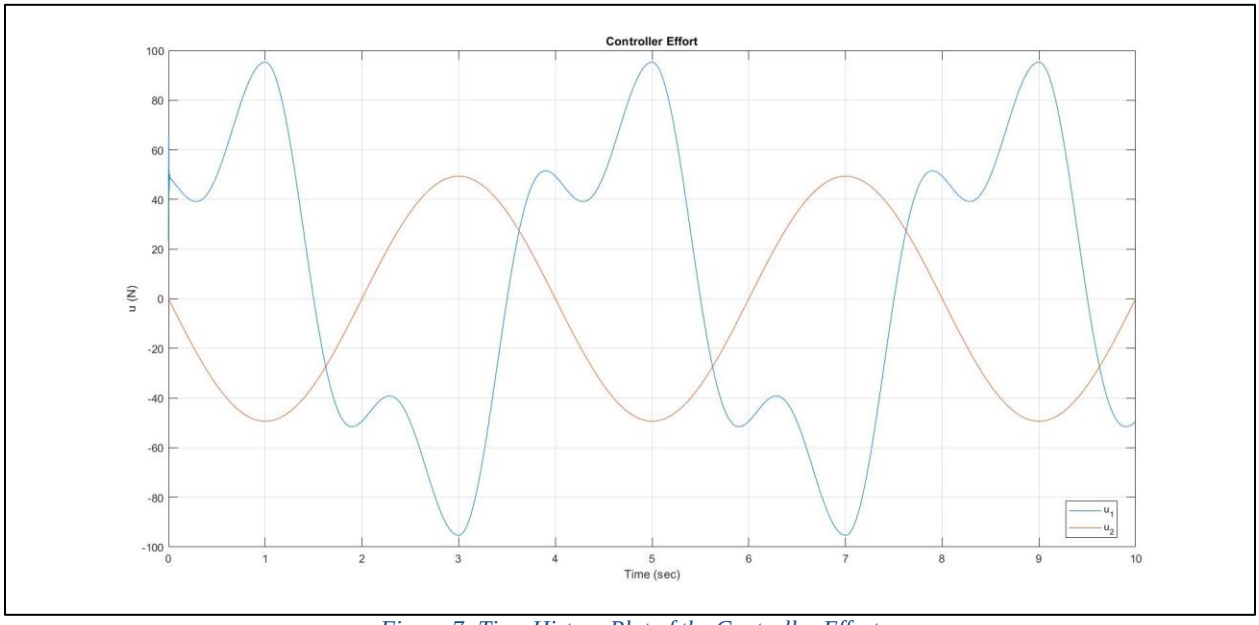


Figure 7: Time History Plot of the Controller Effort

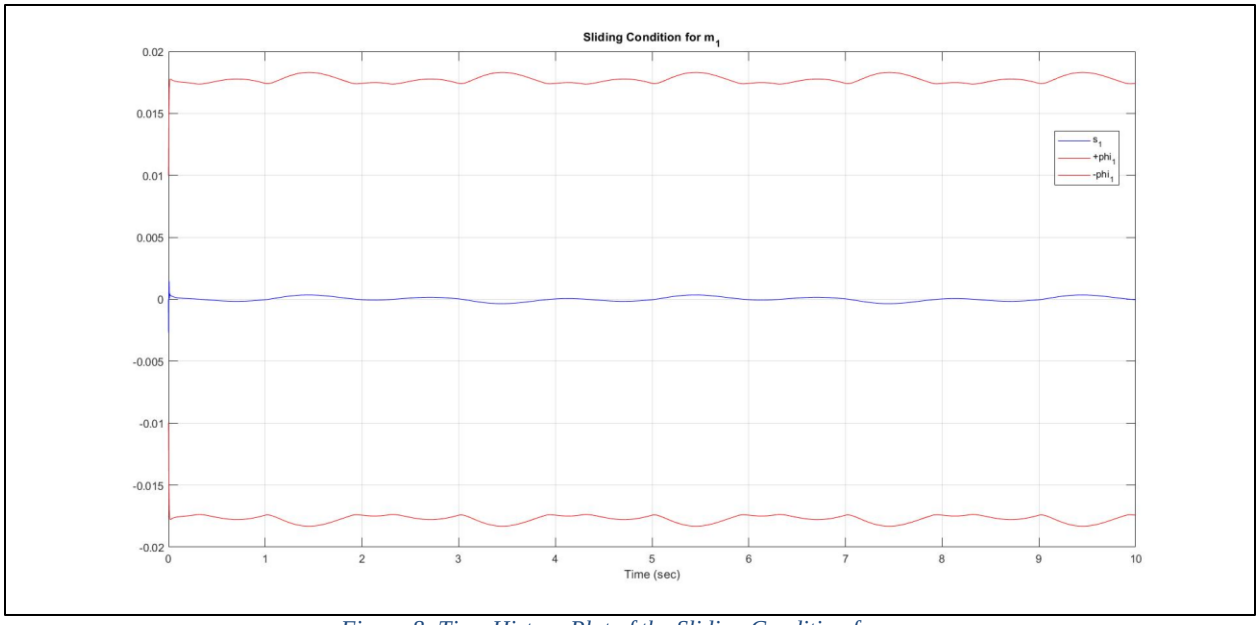


Figure 8: Time History Plot of the Sliding Condition for m_1

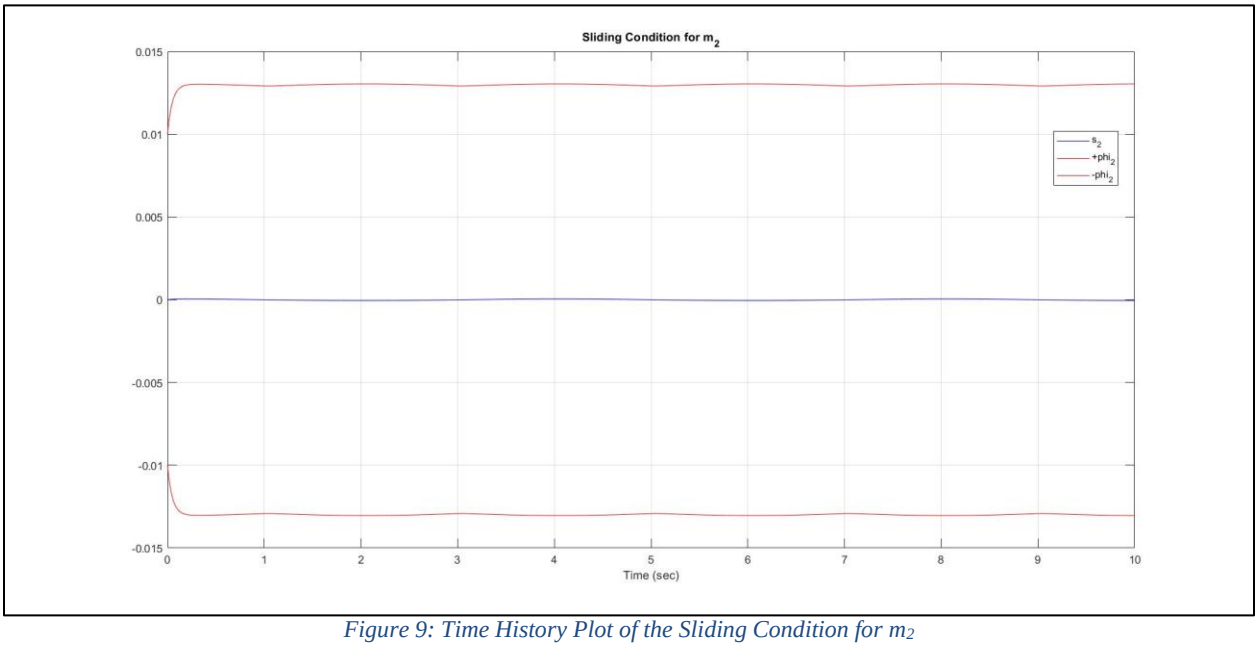


Figure 9: Time History Plot of the Sliding Condition for m_2

The RMS of the difference between the actual and desired state for m_1 and m_2, from Figures 5 and 6, resulted in $8.6823*10^{-6}$ and $1.7000*10^{-6}$ meters, respectfully. This RMS value shows that almost perfect tracking is achieved for both masses. The sliding condition, shown in Figures 8 and 9, is observed to be satisfied for the whole simulation which guarantees asymptotic stability

and the controller effort, shown in Figure 7, is observed to be smooth without high frequency chattering for both controllers.

4.0 METHODOLOGY

The objective of this work is to compare the performance of MFSMC to an optimal PID controller by comparing tracking performance and average power utilization. First, a model of a quadcopter system is built in Simulink using equations that govern 6 DOF airborne systems. The $[B]$ matrix is derived next and then the MFSMC algorithm can be executed. Another simulation using PID controllers is executed; the PID controllers are to be tuned using MATLAB's "fminsearch" function in the Optimization Toolbox while trying to minimize tracking error and control effort.

Both controllers are simulated under the craft's original parameters, doubling only the craft's mass, doubling only the craft's moments of inertia, and doubling both the craft's mass and moments of inertia. For all simulations, the control algorithms are not altered to observe each controllers' performance with significant uncertainty in model parameters.

The controllers are compared by observing each controller's ability to track altitude, roll, pitch, and yaw using RMS of the difference between the desired state and the actual state, as well as observing each controller's average power usage. Lastly, the conclusions and future work are summarized.

5.0 QUADCOPTER SIMULATION

A simulation of a quadcopter is built and controlled by a PID controller and the MFSMC algorithm. The equations of motion are derived for the system, then a derivation for the $[B]$ matrix is derived. The PID controllers are tuned using MATLAB's fminsearch function from the Optimization Toolbox. For each controller design the RMS for altitude, roll, pitch, and yaw is determined and compared. The electrical power consumed and mechanical power output from the system are also compared for each controller.

5.1 Quadcopter Plant Model

Quadcopters are a 6 DOF system with four motors with propellers to generate thrust at each motor. The local frame of the system is described by X, Y, and Z for translational displacement and ϕ, θ, and ψ for angular displacement which are the rotations about X, Y, and Z, respectively, and is represented by Figure 10:

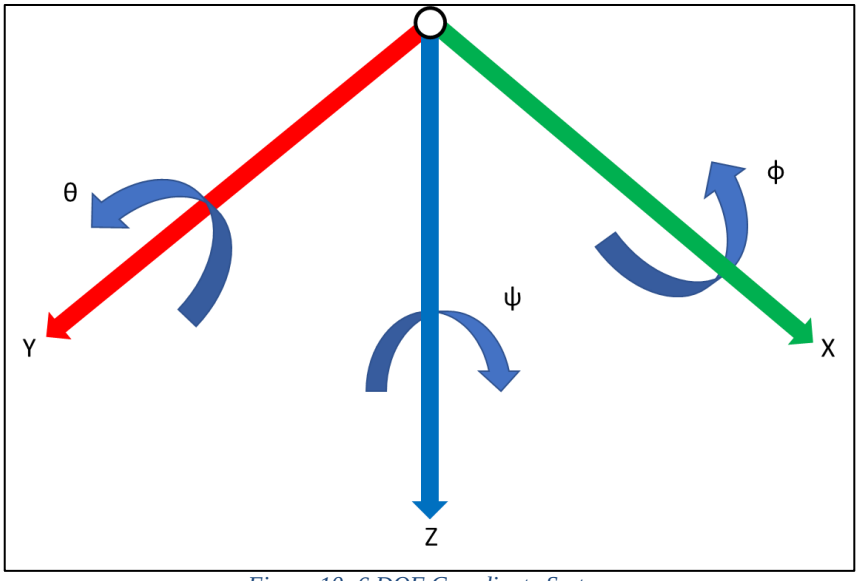


Figure 10: 6 DOF Coordinate System

The states of the quadcopter are described below in Table 3:

Symbol	Description	Units
θ	Pitch Angle	rad
$\dot{\theta}$	Pitch Angular Velocity	rad/sec
$\ddot{\theta}$	Pitch Angular Acceleration	rad/sec^2
ϕ	Roll Angle	rad
$\dot{\phi}$	Roll Angular Velocity	rad/sec
$\ddot{\phi}$	Roll Angular Acceleration	rad/sec^2
ψ	Yaw Angle	rad
$\dot{\psi}$	Yaw Angular Velocity	rad/sec
$\ddot{\psi}$	Yaw Angular Acceleration	rad/sec^2
X	Position along X	m
$\dot{X}$	Velocity along X	m/sec
$\ddot{X}$	Acceleration along X	m/sec^2
Y	Position along Y	m
$\dot{Y}$	Velocity along Y	m/sec
$\ddot{Y}$	Acceleration along Y	m/sec^2
Z	Position along Z	m
$\dot{Z}$	Velocity along Z	m/sec
$\ddot{Z}$	Acceleration along Z	m/sec^2
Ω_1	Motor 1 Angular Velocity	rad/sec
Ω_2	Motor 2 Angular Velocity	rad/sec
Ω_3	Motor 3 Angular Velocity	rad/sec
Ω_4	Motor 4 Angular Velocity	rad/sec

Table 3: Quadcopter States

5.1.1 Reference Frames and Orientations

The two reference frames are assumed to be the local NED (North – East – Down) reference frame and the body-reference frame. The NED reference frame is an inertial frame of reference and the body-reference frame is non-inertial and is assumed to be fixed on the quadcopter as it moves through 3-D space. For the NED reference frame, the X-axis is pointed towards North, the Y-axis is point towards the East, and the Z-axis is pointed down, as shown in Figure 11:

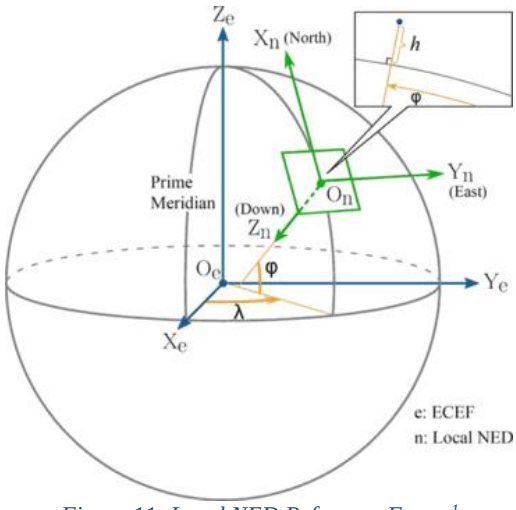


Figure 11: Local NED Reference Frame[1]

The body-reference frame is dependent on the orientation of the quadcopter. The quadcopter is assumed to have an "x" orientation, shown in Figure 12:

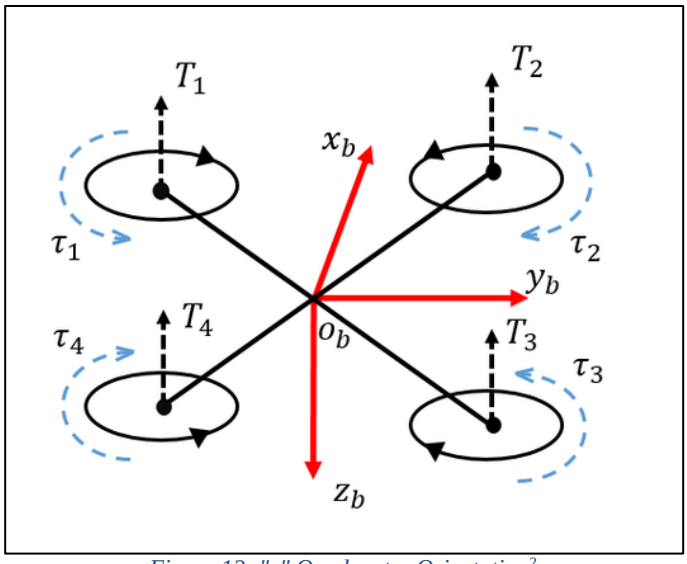


Figure 12: "x" Quadcopter Orientation[2]

[1] https://www.google.com/search?q=local+NED&rlz=1C1RLNS_enUS783US783&sxsrf=ALeKk02C1YPuCchd8dqEhMO6O9qryf9PNw:1595960235660&source=lnms&tbm=isch&sa=X&ved=2ahUKEwiP-6-Xx_DqAhWym-AKHWxnDgYQ_AUoA3oECA4QBQ&biw=1396&bih=657#imgrc=9hkvsjuyY8CA2M

[2] https://www.google.com/search?q=quadcopter+x+configuration+coordinate+frame&tbm=isch&ved=2ahUKEwiAg5bO0PDqAhVLON8KHfkkD3IQ2-cCegQIABAA&oq=quadcopter+x+configuration+coordinate+frame&gs_lcp=CgNpbWcQA1Di9QdYmP8HYNn_B2gAcAB4AIABSIgBjAOSAQE2mAEAoAEBqgELZ3dzLXdpei1pbWfAAQE&sclient=img&ei=jnUgX4CEHsvw_Ab5ybyQBw&bih=596&biw=711&rlz=1C1RLNS_enUS783US783#imgrc=K7SjDHxYctCHMM&imgdii=ZmUcCazlCuDdaM

5.1.2 Plant Model Equations

A mathematical model to represent the quadcopter is shown in this section. The assumptions for the model are:

- The quadcopter is a rigid body
- The geometry is symmetrical
- The quadcopter has uniform mass distribution
- Thrust is proportional to the square of the propeller speed
- Drag is proportional to the square of the vehicle speed
- There are no external disturbances to the system

The thrust at each motor is:

$$F_1 = b\Omega_1^2 \tag{5.1}$$

$$F_2 = b\Omega_2^2 \tag{5.2}$$

$$F_3 = b\Omega_3^2 \tag{5.3}$$

$$F_4 = b\Omega_4^2 \tag{5.4}$$

where F_i is the force at the i^{th} motor, Ω_i is the motor speed of the i^{th} motor, and b is the thrust factor. The moments about the x-axis (roll) from each motor is:

$$M_{1,x} = \left(l \sin\left(\frac{\pi}{4}\right)\right) b\Omega_1^2 \tag{5.5}$$

$$M_{2,x} = -\left(l \sin\left(\frac{\pi}{4}\right)\right) b\Omega_2^2 \tag{5.6}$$

$$M_{3,x} = -\left(l \sin\left(\frac{\pi}{4}\right)\right) b\Omega_3^2 \tag{5.7}$$

$$M_{4,x} = \left(l \sin\left(\frac{\pi}{4}\right)\right) b\Omega_4^2 \tag{5.8}$$

where $M_{i,x}$ is the rolling moment due to the i^{th} motor, and l is the length of the center of gravity to the motor. The moments about y-axis (pitch) from each motor are:

$$M_{1,y} = \left(l \sin\left(\frac{\pi}{4}\right)\right) b\Omega_1^2 \tag{5.9}$$

$$M_{2,y} = \left(l \sin\left(\frac{\pi}{4}\right)\right) b\Omega_2^2 \tag{5.10}$$

$$M_{3,y} = -\left(l \sin\left(\frac{\pi}{4}\right)\right) b\Omega_3^2 \tag{5.11}$$

$$M_{4,y} = -\left(l \sin\left(\frac{\pi}{4}\right)\right) b\Omega_4^2 \tag{5.12}$$

where $M_{i,y}$ is the pitching moment due to the i^{th} motor. The torques from each motor are:

$$T_1 = d\Omega_1^2 \tag{5.13}$$

$$T_2 = -d\Omega_2^2 \tag{5.14}$$

$$T_3 = d\Omega_3^2 \tag{5.15}$$

$$T_4 = -d\Omega_4^2 \tag{5.16}$$

where T_i is the torque produced by the i^{th} motor. Summing all the forces and moments yields the equations:

$$F_{thrust} = F_1 + F_2 + F_3 + F_4 \tag{5.17}$$

$$M_{roll} = M_{1,x} + M_{2,x} + M_{3,x} + M_{4,x} \tag{5.18}$$

$$M_{pitch} = M_{1,y} + M_{2,y} + M_{3,y} + M_{4,y} \tag{5.19}$$

$$M_{yaw} = T_1 + T_2 + T_3 + T_4 \tag{5.20}$$

sub Eqs. (5.1 – 5.16) into Eqs. (5.17 – 5.20):

$$F_{thrust} = b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \tag{5.21}$$

$$M_{roll} = lb \sin\left(\frac{\pi}{4}\right)(\Omega_1^2 - \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \tag{5.22}$$

$$M_{pitch} = lb \sin\left(\frac{\pi}{4}\right)(\Omega_1^2 + \Omega_2^2 - \Omega_3^2 - \Omega_4^2) \tag{5.23}$$

$$M_{yaw} = d(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2) \tag{5.24}$$

The equations of motion for a quadcopter system is given by:

$$\dot{u} = -g \sin(\theta) + vr - wq - \frac{C_d \rho A_u}{2m} u^2 \tag{5.25}$$

$$\dot{v} = g \sin(\phi) \cos(\theta) - ur + wp - \frac{C_d \rho A_v}{2m} v^2 \tag{5.27}$$

$$\dot{w} = g \cos(\phi) \cos(\theta) + uq - vp + \frac{C_d \rho A_w}{2m} w^2 - \frac{F_{thrust}}{m} \tag{5.28}$$

$$\begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{xy} & I_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz} \end{bmatrix} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} qr(I_{yy} - I_{zz}) + (q^2 - r^2)I_{yz} - prI_{xy} + pqI_{xz} + M_x \\ pr(I_{zz} - I_{xx}) + (r^2 - p^2)I_{xz} - pqI_{yz} + qrI_{xy} + M_y \\ pq(I_{xx} - I_{yy}) + (p^2 - q^2)I_{xy} - qrI_{xz} + prI_{yz} + M_z \end{bmatrix} \tag{5.29}$$

where u, v, w are the velocity of the craft along the x, y, z-body axes, respectively, g is the acceleration due to gravity near the Earth's surface, p, q, r are the roll rate, pitch rate, yaw rate, respectively, C_d is the drag coefficient, ρ is the density of atmosphere, m is the mass of the craft, A_u, A_v, A_w are the cross-sectional areas normal to the x, y, z-body axes, respectively, and I_{ij} is the moment of inertia about the i, j-axes. The moments are given by:

$$M_x = M_{roll} + J_m q \Omega_r \tag{5.30}$$

$$M_y = M_{pitch} + J_m p \Omega_r \tag{5.31}$$

$$M_z = M_{yaw} \tag{5.32}$$

where:

$$\Omega_r = \Omega_1 - \Omega_2 + \Omega_3 - \Omega_4 \tag{5.33}$$

and from the assumptions it is known that:

$$I_{xy} = 0 \tag{5.34}$$

$$I_{xz} = 0 \tag{5.35}$$

$$I_{yz} = 0 \tag{5.36}$$

sub Eqs. (5.30 – 5.36) into Eq. (5.37):

$$\dot{p} = \frac{qr(I_{yy} - I_{zz})}{I_{xx}} + \frac{M_{roll} + J_m q \Omega_r}{I_{xx}} \tag{5.37}$$

$$\dot{q} = \frac{pr(I_{zz} - I_{xx})}{I_{yy}} + \frac{M_{pitch} + J_m p \Omega_r}{I_{yy}} \tag{5.38}$$

$$\dot{r} = \frac{pq(I_{xx} - I_{yy})}{I_{zz}} + \frac{M_{yaw}}{I_{zz}} \tag{5.39}$$

sub Eqs. (5.22 – 5.24) into Eqs. (5.37 – 5.39):

$$\dot{p} = \frac{qr(I_{yy} - I_{zz})}{I_{xx}} + \frac{lb \sin\left(\frac{\pi}{4}\right)\left(\Omega_1^2 - \Omega_2^2 - \Omega_3^2 + \Omega_4^2\right) + J_m q \Omega_r}{I_{xx}} \tag{5.37}$$

$$\dot{q} = \frac{pr(I_{zz}-I_{xx})}{I_{yy}} + \frac{lb\sin\left(\frac{\pi}{4}\right)\left(\Omega_1^2+\Omega_2^2-\Omega_3^2-\Omega_4^2\right)+J_m p\Omega_r}{I_{yy}} \tag{5.38}$$

$$\dot{r} = \frac{pq(I_{xx}-I_{yy})}{I_{zz}} + \frac{d\left(\Omega_1^2-\Omega_2^2+\Omega_3^2-\Omega_4^2\right)}{I_{zz}} \tag{5.39}$$

sub Eq. (5.21) into Eq. (5.28):

$$\dot{w} = g\cos(\phi)\cos(\theta) + uq - vp + \frac{C_d\rho A_w}{2m}w^2 - \frac{b\left(\Omega_1^2+\Omega_2^2+\Omega_3^2+\Omega_4^2\right)}{m} \tag{5.40}$$

The Euler equations are given by:

$$\dot{\phi} = p + q\sin(\phi)\tan(\theta) + r\cos(\phi)\tan(\theta) \tag{5.41}$$

$$\dot{\theta} = q\cos(\phi) - r\sin(\phi) \tag{5.42}$$

$$\dot{\psi} = (q\sin(\phi) + r\cos(\phi))\sec(\theta) \tag{5.43}$$

Accounting for motor dynamics gives the following equations:

$$L\frac{di_i}{dt} = V_{i_{cmd}} - Ri_i - K_{emf}\Omega_i \tag{5.44}$$

$$J_m\frac{d\Omega_i}{dt} = K_t i_i - b_m\Omega_i \tag{5.45}$$

where L is the inductance, i_i is the current through the i^{th} motor, $V_{i_{cmd}}$ is the voltage command to the i^{th} motor, R is the electrical resistance of the motor, K_{emf} is the back *emf* voltage generated per the angular velocity of the motor, J_m is the moment of inertia of the motor, K_t is the moment generated per amount of current, and b_m is the moment generated from friction per the angular velocity of the motor. Lastly, to obtain the local NED coordinates the following matrix operation is utilized:

$$\begin{bmatrix}\ddot{X}_{NED}\\ \ddot{Y}_{NED}\\ \ddot{Z}_{NED}\end{bmatrix}=\begin{bmatrix}\cos(\psi)\cos(\theta) & -\sin(\psi)\cos(\phi)+\cos(\psi)\sin(\theta)\sin(\phi) & \sin(\psi)\sin(\phi)-\cos(\psi)\sin(\theta)\cos(\phi)\\ \sin(\psi)\cos(\theta) & \cos(\psi)\cos(\phi)+\sin(\psi)\sin(\theta)\sin(\phi) & -\cos(\psi)\sin(\phi)+\sin(\psi)\sin(\theta)\cos(\phi)\\ -\sin(\theta) & \cos(\theta)\sin(\phi) & \cos(\theta)\cos(\phi)\end{bmatrix}\begin{bmatrix}\dot{u}\\ \dot{v}\\ \dot{w}\end{bmatrix} \quad (5.46)$$

5.2 Deriving the $[B]$ Matrix

In this section the $[B]$ matrix used in the MFSMC algorithm is derived. Restating Eqs. (5.41 – 5.43):

$$\dot{\phi} = p + q\sin(\phi)\tan(\theta) + r\cos(\phi)\tan(\theta) \quad (5.41)$$

$$\dot{\theta} = q\cos(\phi) - r\sin(\phi) \quad (5.42)$$

$$\dot{\psi} = (q\sin(\phi) + r\cos(\phi))\sec(\theta) \quad (5.43)$$

differentiating Eqs. (5.41 – 5.43) gives:

$$\ddot{\phi} = \dot{p} + \dot{\theta}\sec^2\theta\,(q\sin\phi + r\cos\phi) + \dot{\phi}\tan\theta\,(q\cos\phi - r\sin\phi) + \tan\theta\,(\dot{q}\sin\phi + \dot{r}\cos\phi) \quad (5.47)$$

$$\ddot{\theta} = \dot{q}\cos\phi - \dot{r}\sin\phi - \dot{\phi}(q\sin\phi + r\cos\phi) \quad (5.48)$$

$$\ddot{\psi} = \sec\theta\left[\dot{\theta}\tan\theta\,(q\sin\phi + r\cos\phi) + \dot{\phi}(q\cos\phi - r\sin\phi) + \dot{q}\sin\phi + \dot{r}\cos\phi\right] \quad (5.49)$$

sub Eqs. (5.37 – 5.39) into Eqs. (5.47 – 5.49):

$$\ddot{\phi} = qr\frac{(I_{yy}-I_{zz})}{I_{xx}} + \frac{1}{I_{xx}}M_x + \tan(\theta)\sin(\phi)\left(pr\frac{(I_{zz}-I_{xx})}{I_{yy}} + \frac{1}{I_{yy}}M_y\right) +$$

$$\tan(\theta)\cos(\phi)\left(pq\frac{(I_{xx}-I_{yy})}{I_{zz}} + \frac{1}{I_{zz}}M_z\right) + \dot{\theta}\sec(\theta)^2\left(q\sin(\phi) + r\cos(\phi)\right) +$$

$$\dot{\phi}\tan(\theta)\,(q\cos(\phi) - r\sin(\phi)) \quad (5.50)$$

$$\ddot{\theta} = \cos(\phi)\left(pr\frac{(I_{zz}-I_{xx})}{I_{yy}} + \frac{1}{I_{yy}}M_y\right) - \sin(\phi)\left(\sec(\theta)\sin(\phi)\right) - \dot{\phi}(q\sin(\phi) + r\cos(\phi))$$

(5.51)

$$\ddot{\psi} =$$

$$\sec(\theta)\sin(\phi)\left(pr\frac{(I_{zz}-I_{xx})}{I_{yy}} + \frac{1}{I_{yy}}M_y\right) + \sec(\theta)\cos(\phi)\left(pq\frac{(I_{xx}-I_{yy})}{I_{zz}} + \frac{1}{I_{zz}}M_z\right) + \sec(\theta)\left[\dot{\theta}\tan\theta\,(q\sin\phi + r\cos\phi) + \dot{\phi}(q\cos\phi - r\sin\phi)\right]$$

(5.52)

rearranging Eqs. (5.41 – 5.43) gives:

$$p = \dot{\phi} - \dot{\psi}\sin\theta$$

(5.53)

$$q = \dot{\psi}\sin\phi\cos\theta + \dot{\theta}\cos\phi$$

(5.54)

$$r = \dot{\psi}\cos\phi\cos\theta - \dot{\theta}\sin\phi$$

(5.55)

so that:

$$pq = \dot{\psi}\left(\dot{\phi}\sin(\phi)\cos(\theta) - \sin(\theta)\left(\dot{\psi}\sin(\phi)\cos(\theta) + \dot{\theta}\cos(\phi)\right)\right) + \dot{\phi}\dot{\theta}\cos(\phi)$$

(5.56)

$$pr = \dot{\psi}\left(\dot{\phi}\cos(\phi)\cos(\theta) - \sin(\theta)\left(\dot{\psi}\cos(\phi)\cos(\theta) + \dot{\theta}\sin(\phi)\right)\right) - \dot{\phi}\dot{\theta}\sin(\phi)$$

(5.57)

$$qr = \dot{\psi}\left(\dot{\psi}\sin(\phi)\cos(\phi)\cos(\theta)^2 - \sin(\theta)\left(\dot{\psi}\cos(\phi)\cos(\theta) + \dot{\theta}\sin(\phi)\right)\right) - \dot{\theta}^2\sin(\phi)\cos(\phi)$$

(5.58)

sub Eqs. (5.53 – 5.58) into Eqs. (5.50 – 5.52):

$$\ddot{\phi} = \frac{(I_{yy}-I_{zz})}{I_{xx}}\left[\dot{\psi}\left(\dot{\psi}\sin(\phi)\cos(\phi)\cos(\theta)^2 - \dot{\theta}\cos(\theta)\cos(2\theta)\right) - \dot{\theta}^2\sin(\phi)\cos(\phi)\right] +$$

$$\tan(\theta)\sin(\phi)\frac{(I_{zz}-I_{xx})}{I_{yy}}\left[\dot{\psi}\left(\dot{\phi}\cos(\phi)\cos(\theta) - \sin(\theta)\left(\dot{\psi}\cos(\phi)\cos(\theta) - \dot{\theta}\sin(\phi)\right)\right) -\right.$$

$$\dot{\phi}\dot{\theta}\sin(\phi)\Big] + \tan(\theta)\cos(\phi)\frac{(I_{xx}-I_{yy})}{I_{zz}}\Big[\dot{\psi}\Big(\dot{\phi}\sin(\phi)\cos(\theta) - \sin(\theta)\big(\dot{\psi}\sin(\phi)\cos(\theta) -$$

$$\dot{\theta}\cos(\phi)\big)\Big) + \dot{\phi}\dot{\theta}\cos(\phi)\Big] + 2\dot{\psi}\dot{\theta}\sin(\phi)^2\sec(\theta) + \dot{\phi}\dot{\theta}\tan(\theta) + \frac{1}{I_{xx}}M_x + \frac{\tan(\theta)\sin(\phi)}{I_{yy}}M_y +$$

$$\frac{\tan(\theta)\cos(\phi)}{I_{zz}}M_z \tag{5.59}$$

$$\ddot{\theta} = \cos(\phi)\frac{(I_{zz}-I_{xx})}{I_{yy}}\Big[\dot{\psi}\Big(\dot{\phi}\cos(\phi)\cos(\theta) - \sin(\theta)\big(\dot{\psi}\cos(\phi)\cos(\theta) - \dot{\theta}\sin(\phi)\big)\Big) -$$

$$\dot{\phi}\dot{\theta}\sin(\phi)\Big] - \sin(\phi)\frac{(I_{xx}-I_{yy})}{I_{zz}}\Big[\dot{\psi}\Big(\dot{\phi}\sin(\phi)\cos(\theta) - \sin(\theta)\big(\dot{\psi}\sin(\phi)\cos(\theta) + \dot{\theta}\cos(\phi)\big)\Big) +$$

$$\dot{\phi}\dot{\theta}\cos(\phi)\Big] - \dot{\psi}\dot{\phi}\cos(\theta) + \frac{\cos(\phi)}{I_{yy}}M_y + \frac{\sin(\phi)}{I_{zz}}M_z \tag{5.60}$$

$$\ddot{\psi} = \sec(\theta)\sin(\phi)\frac{(I_{zz}-I_{xx})}{I_{yy}}\Big[\dot{\psi}\Big(\dot{\phi}\cos(\phi)\cos(\theta) - \sin(\theta)\big(\dot{\psi}\cos(\phi)\cos(\theta) - \dot{\theta}\sin(\phi)\big)\Big) -$$

$$\dot{\phi}\dot{\theta}\sin(\phi)\Big] - \sec(\theta)\cos(\phi)\frac{(I_{xx}-I_{yy})}{I_{zz}}\Big[\dot{\psi}\Big(\dot{\phi}\sin(\phi)\cos(\theta) - \sin(\theta)\big(\dot{\psi}\sin(\phi)\cos(\theta) +$$

$$\dot{\theta}\cos(\phi)\big)\Big) + \dot{\phi}\dot{\theta}\cos(\phi)\Big] + \dot{\theta}\big(\dot{\psi}\tan(\theta) + \dot{\phi}\sec(\theta)\big) + \frac{\sec(\theta)\sin(\phi)}{I_{yy}}M_y + \frac{\sec(\theta)\cos(\phi)}{I_{zz}}M_z$$

(5.61)

To reduce the size of Eqs. (5.59 – 5.61) placeholder variables are defined:

$$\Gamma_1 = \frac{(I_{yy}-I_{zz})}{I_{xx}}\big[\dot{\psi}\big(\dot{\psi}\sin(\phi)\cos(\phi)\cos(\theta)^2 - \dot{\theta}\cos(\theta)\cos(2\theta)\big) - \dot{\theta}^2\sin(\phi)\cos(\phi)\big] +$$

$$\tan(\theta)\sin(\phi)\frac{(I_{zz}-I_{xx})}{I_{yy}}\Big[\dot{\psi}\Big(\dot{\phi}\cos(\phi)\cos(\theta) - \sin(\theta)\big(\dot{\psi}\cos(\phi)\cos(\theta) - \dot{\theta}\sin(\phi)\big)\Big) -$$

$$\dot{\phi}\dot{\theta}\sin(\phi)\Big] + \tan(\theta)\cos(\phi)\frac{(I_{xx}-I_{yy})}{I_{zz}}\Big[\dot{\psi}\Big(\dot{\phi}\sin(\phi)\cos(\theta) - \sin(\theta)\big(\dot{\psi}\sin(\phi)\cos(\theta) -$$

$$\dot{\theta}\cos(\phi)\big)\Big) + \dot{\phi}\dot{\theta}\cos(\phi)\Big] + 2\dot{\psi}\dot{\theta}\sin(\phi)^2\sec(\theta) + \dot{\phi}\dot{\theta}\tan(\theta) + \frac{\tan(\theta)\sin(\phi)}{I_{yy}}M_y +$$

$$\frac{\tan(\theta)\cos(\phi)}{I_{zz}}M_z \tag{5.62}$$

$$\Gamma_2 = \cos(\phi)\frac{(I_{zz}-I_{xx})}{I_{yy}}\Big[\dot{\psi}\Big(\dot{\phi}\cos(\phi)\cos(\theta) - \sin(\theta)\big(\dot{\psi}\cos(\phi)\cos(\theta) - \dot{\theta}\sin(\phi)\big)\Big) -$$

$$\dot{\phi}\dot{\theta}\sin(\phi)\Big] - \sin(\phi)\frac{(I_{xx}-I_{yy})}{I_{zz}}\Big[\dot{\psi}\Big(\dot{\phi}\sin(\phi)\cos(\theta) - \sin(\theta)\big(\dot{\psi}\sin(\phi)\cos(\theta) + \dot{\theta}\cos(\phi)\big)\Big) +$$

$$\dot{\phi}\dot{\theta}\cos(\phi)\Big] - \dot{\psi}\dot{\phi}\cos(\theta) + \frac{\sin(\phi)}{I_{zz}}M_z \tag{5.63}$$

$$\Gamma_3 = \sec(\theta)\sin(\phi)\frac{(I_{zz}-I_{xx})}{I_{yy}}\Big[\dot{\psi}\Big(\dot{\phi}\cos(\phi)\cos(\theta) - \sin(\theta)\big(\dot{\psi}\cos(\phi)\cos(\theta) - \dot{\theta}\sin(\phi)\big)\Big) -$$

$$\dot{\phi}\dot{\theta}\sin(\phi)\Big] - \sec(\theta)\cos(\phi)\frac{(I_{xx}-I_{yy})}{I_{zz}}\Big[\dot{\psi}\Big(\dot{\phi}\sin(\phi)\cos(\theta) - \sin(\theta)\big(\dot{\psi}\sin(\phi)\cos(\theta) +$$

$$\dot{\theta}\cos(\phi)\big)\Big) + \dot{\phi}\dot{\theta}\cos(\phi)\Big] + \dot{\theta}\big(\dot{\psi}\tan(\theta) + \dot{\phi}\sec(\theta)\big) + \frac{\sec(\theta)\sin(\phi)}{I_{yy}}M_y \tag{5.64}$$

then sub Eqs. (5.62 – 5.64) into Eqs. (5.59 – 5.61):

$$\ddot{\phi} = \Gamma_1 + \frac{1}{I_{xx}}M_x \tag{5.65}$$

$$\ddot{\theta} = \Gamma_2 + \frac{\cos(\phi)}{I_{yy}}M_y \tag{5.66}$$

$$\ddot{\psi} = \Gamma_3 + \frac{\sec(\theta)\cos(\phi)}{I_{zz}}M_z \tag{5.67}$$

sub Eqs. (5.30 – 5.32) into Eqs. (5.65 -5.67):

$$\ddot{\phi} = \Gamma_1 + \frac{1}{I_{xx}}(M_{roll} + J_m q\Omega_r) \tag{5.68}$$

$$\ddot{\theta} = \Gamma_2 + \frac{\cos(\phi)}{I_{yy}}(M_{pitch} + J_m q\Omega_r) \tag{5.69}$$

$$\ddot{\psi} = \Gamma_3 + \frac{\sec(\theta)\cos(\phi)}{I_{zz}} M_{yaw} \tag{5.70}$$

sub Eqs. (5.22 – 5.24) into Eqs. (5.68 – 5.70):

$$\ddot{\phi} = \Gamma_1 + \frac{J_m q \Omega_r}{I_{xx}} + \frac{lb \sin\left(\frac{\pi}{4}\right)\left(\Omega_1^2 - \Omega_2^2 - \Omega_3^2 + \Omega_4^2\right)}{I_{xx}} \tag{5.71}$$

$$\ddot{\theta} = \Gamma_2 + \cos(\phi) \frac{J_m q \Omega_r}{I_{yy}} + \cos(\phi) \frac{lb \sin\left(\frac{\pi}{4}\right)\left(\Omega_1^2 + \Omega_2^2 - \Omega_3^2 - \Omega_4^2\right)}{I_{yy}} \tag{5.72}$$

$$\ddot{\psi} = \Gamma_3 + \sec(\theta)\cos(\phi) \frac{d\left(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2\right)}{I_{zz}} \tag{5.73}$$

Using the motor dynamics, a relationship is built between the motor speed and voltage command. Recall Eqs. (5.44 – 5.45):

$$L \frac{di_i}{dt} = V_{i_{cmd}} - R i_i - K_{emf} \Omega_i \tag{5.44}$$

$$J_m \frac{d\Omega_i}{dt} = K_t i_i - b_m \Omega_i \tag{5.45}$$

assuming steady-state of the motors Eqs. (5.44 – 5.45) reduce to:

$$0 = V_{i_{cmd}} - R i_i - K_{emf} \Omega_i \tag{5.74}$$

$$0 = K_t i_i - b_m \Omega_i \tag{5.75}$$

combining Eqs. (5.74 – 5.75) and solving for Ω_i yields:

$$\Omega_i = \frac{K_t}{K_t K_{emf} + R b_m} V_{i_{cmd}} \tag{5.76}$$

sub Eq. (5.76) into Eq. (5.40) and Eqs. (5.71 – 5.73):

$$\dot{w} = g\cos(\phi)\cos(\theta) + uq - vp + \frac{C_d \rho A_w}{2m} w^2 - \frac{b\left(\frac{K_t}{K_t K_{emf} + Rb_m}\right)^2 \left(V_1^2 + V_2^2 + V_3^2 + V_4^2\right)}{m} \quad (5.71)$$

$$\dot{\phi} = \Gamma_1 + \frac{J_m q \Omega_r}{I_{xx}} + \frac{lb\sin\left(\frac{\pi}{4}\right)\left(\frac{K_t}{K_t K_{emf} + Rb_m}\right)^2 \left(V_1^2 - V_2^2 - V_3^2 + V_4^2\right)}{I_{xx}} \quad (5.72)$$

$$\ddot{\theta} = \Gamma_2 + \cos(\phi)\frac{J_m q \Omega_r}{I_{yy}} + \cos(\phi)\frac{lb\sin\left(\frac{\pi}{4}\right)\left(\frac{K_t}{K_t K_{emf} + Rb_m}\right)^2 \left(V_1^2 + V_2^2 - V_3^2 - V_4^2\right)}{I_{yy}} \quad (5.73)$$

$$\ddot{\psi} = \Gamma_3 + \sec(\theta)\cos(\phi)\frac{d\left(\frac{K_t}{K_t K_{emf} + Rb_m}\right)^2 \left(V_1^2 - V_2^2 + V_3^2 - V_4^2\right)}{I_{zz}} \quad (5.74)$$

since altitude is being tracked, a coordinate transformation from $\dot{w}$ to $\ddot{h}$ is needed, where $\ddot{h}$ is the acceleration of the craft's altitude:

$$\ddot{h} = \cos(\phi)\cos(\theta)\left(-g - uq + vp - \frac{C_d \rho A_w}{2m} w^2\right) + \cos(\phi)\cos(\theta)\frac{b\left(\frac{K_t}{K_t K_{emf} + Rb_m}\right)^2 \left(V_1^2 + V_2^2 + V_3^2 + V_4^2\right)}{m} \quad (5.75)$$

Next the following control inputs are defined based off Eqs. (5.72 – 5.75):

$$u_h = V_1^2 + V_2^2 + V_3^2 + V_4^2 \quad (5.76)$$

$$u_\phi = V_1^2 - V_2^2 - V_3^2 + V_4^2 \quad (5.77)$$

$$u_\theta = V_1^2 + V_2^2 - V_3^2 - V_4^2 \quad (5.78)$$

$$u_\psi = V_1^2 - V_2^2 + V_3^2 - V_4^2 \quad (5.79)$$

where u_h, u_ϕ, u_θ, and u_ψ are the controller outputs for altitude, roll, pitch, and yaw, respectively. Sub Eqs. (5.76 – 5.79) into Eqs. (5.72 – 5.75):

$$\ddot{h} = \cos(\phi)\cos(\theta)\left(-g - uq + vp - \frac{C_d \rho A_w}{2m} w^2\right) + \cos(\phi)\cos(\theta)\frac{b\left(\frac{K_t}{K_t K_{emf} + R b_m}\right)^2}{m} u_h \tag{5.80}$$

$$\ddot{\phi} = \Gamma_1 + \frac{J_m q \Omega_r}{I_{xx}} + \frac{lb \sin\left(\frac{\pi}{4}\right)\left(\frac{K_t}{K_t K_{emf} + R b_m}\right)^2}{I_{xx}} u_\phi \tag{5.81}$$

$$\ddot{\theta} = \Gamma_2 + \cos(\phi)\frac{J_m q \Omega_r}{I_{yy}} + \cos(\phi)\frac{lb \sin\left(\frac{\pi}{4}\right)\left(\frac{K_t}{K_t K_{emf} + R b_m}\right)^2}{I_{yy}} u_\theta \tag{5.82}$$

$$\ddot{\psi} = \Gamma_3 + \sec(\theta)\cos(\phi)\frac{d\left(\frac{K_t}{K_t K_{emf} + R b_m}\right)^2}{I_{zz}} u_\psi \tag{5.83}$$

from Eqs. (5.80 – 5.83) the $[B]$ matrix can be derived to be:

$$[B] = \begin{bmatrix} \frac{bK_t^2}{m(K_t K_{emf} + R b_m)^2} & 0 & 0 & 0 \\ 0 & \frac{lb \sin\left(\frac{\pi}{4}\right) K_t^2}{I_{xx}(K_t K_{emf} + R b_m)^2} & 0 & 0 \\ 0 & 0 & \frac{lb \sin\left(\frac{\pi}{4}\right) K_t^2}{I_{yy}(K_t K_{emf} + R b_m)^2} & 0 \\ 0 & 0 & 0 & \frac{dK_t^2}{I_{zz}(K_t K_{emf} + R b_m)^2} \end{bmatrix} \tag{5.84}$$

solving Eqs. (5.76 – 5.79) gives:

$$V_1 = \frac{1}{2}\sqrt{u_h + u_\phi + u_\theta + u_\psi} \tag{5.85}$$

$$V_2 = \frac{1}{2}\sqrt{u_h - u_\phi + u_\theta - u_\psi} \tag{5.86}$$

$$V_3 = \frac{1}{2}\sqrt{u_h - u_\phi - u_\theta + u_\psi} \tag{5.87}$$

$$V_4 = \frac{1}{2}\sqrt{u_h + u_\phi - u_\theta - u_\psi} \tag{5.88}$$

5.3 Simulation Study

In this section a simulation of a quadcopter is performed by tracking various signals, which were determined by Sreeraj [3], that were strategized to not saturate the controller output but also to adequately test the performance of the system using the MFSMC algorithm, using the equations in sections 5.1 – 5.2, and a PID controller which is optimized using MATLAB's "fminsearch" function from the Optimization Toolbox. The parameters of the craft are altered in some simulations without altering the either control algorithms to observe performance with significant uncertainty in the model parameters. The RMS of the difference between actual state and the desired state as well as the average power consumed are used to compare each controller.

For the MFSMC algorithm, the following parameters were used:

	lambda (λ)	eta (η)
u_h	30	0.75
u_ϕ	30	0.35
u_θ	30	0.7
u_ψ	30	0.3

Table 4: MFSMC Controller Parameters

For the PID controller, the following coefficients were used:

	u_h	u_ϕ	u_θ	u_ψ
K_P	5.74384E+04	-6.50040E+00	1.70906E+02	2.49697E+02
K_I	2.04123E+03	-1.96484E-02	1.33983E+01	2.95907E+01
K_D	3.03855E+03	4.87282E+02	2.07731E+03	2.48264E+03

Table 5: PID Controller Parameters

5.3.1 Original Craft Parameters

In this simulation, all the craft's parameters are not altered. The performance of the PID controller is presented first:

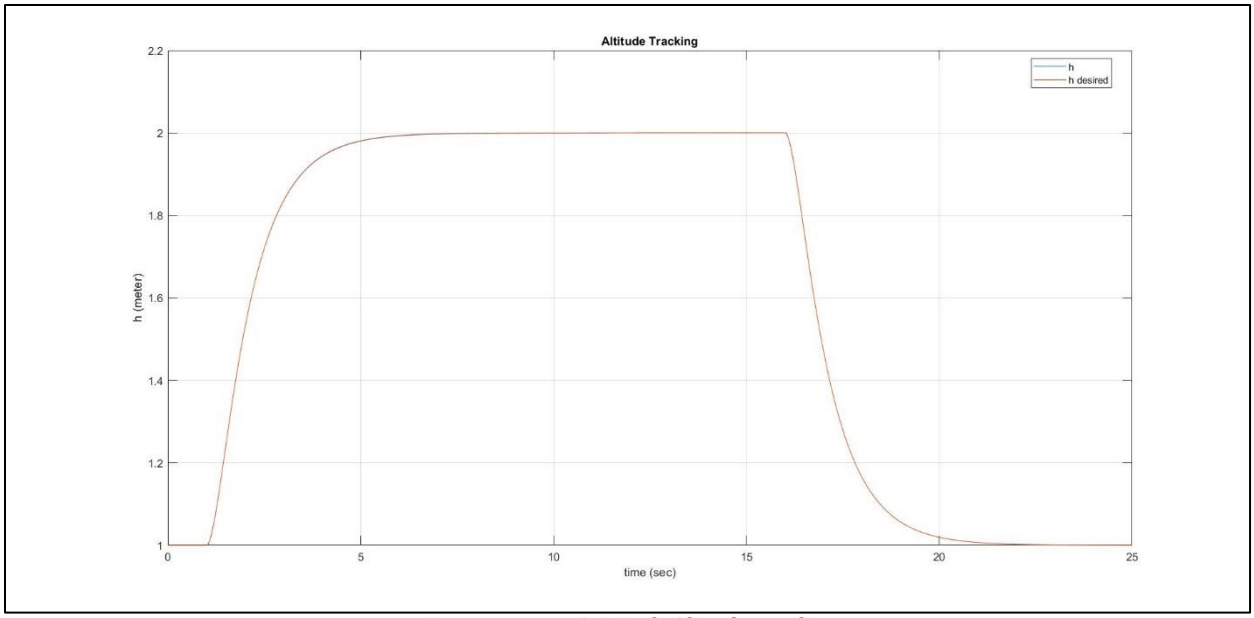


Figure 13: PID, Original; Altitude Tracking

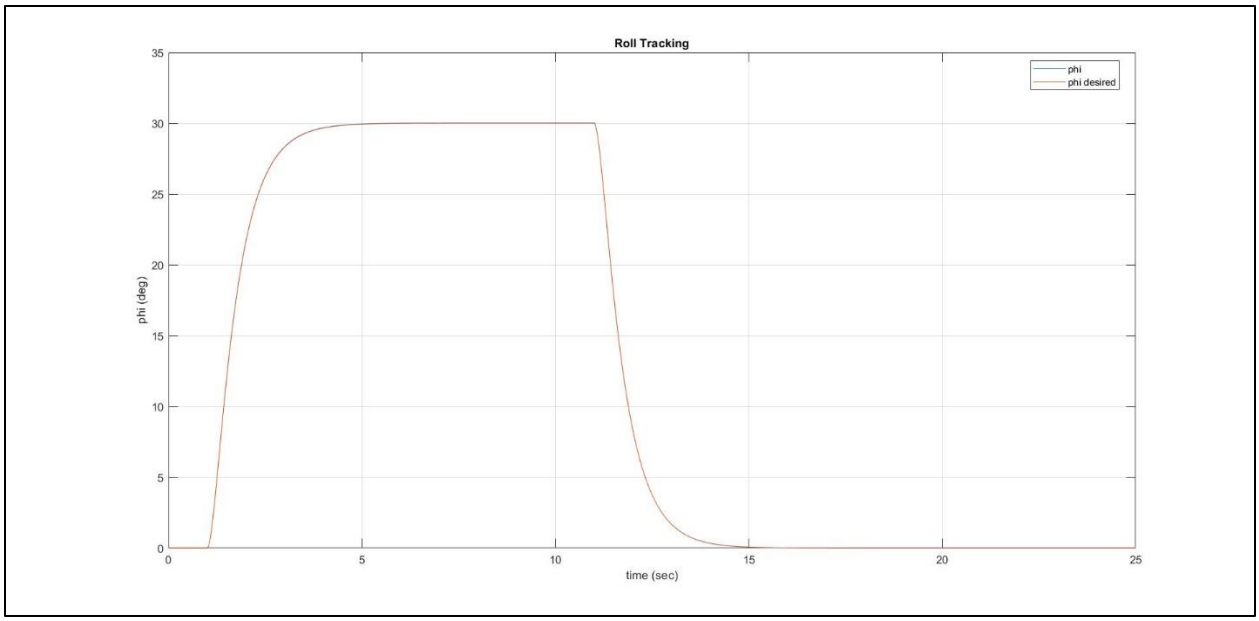


Figure 14: PID, Original; Roll Tracking

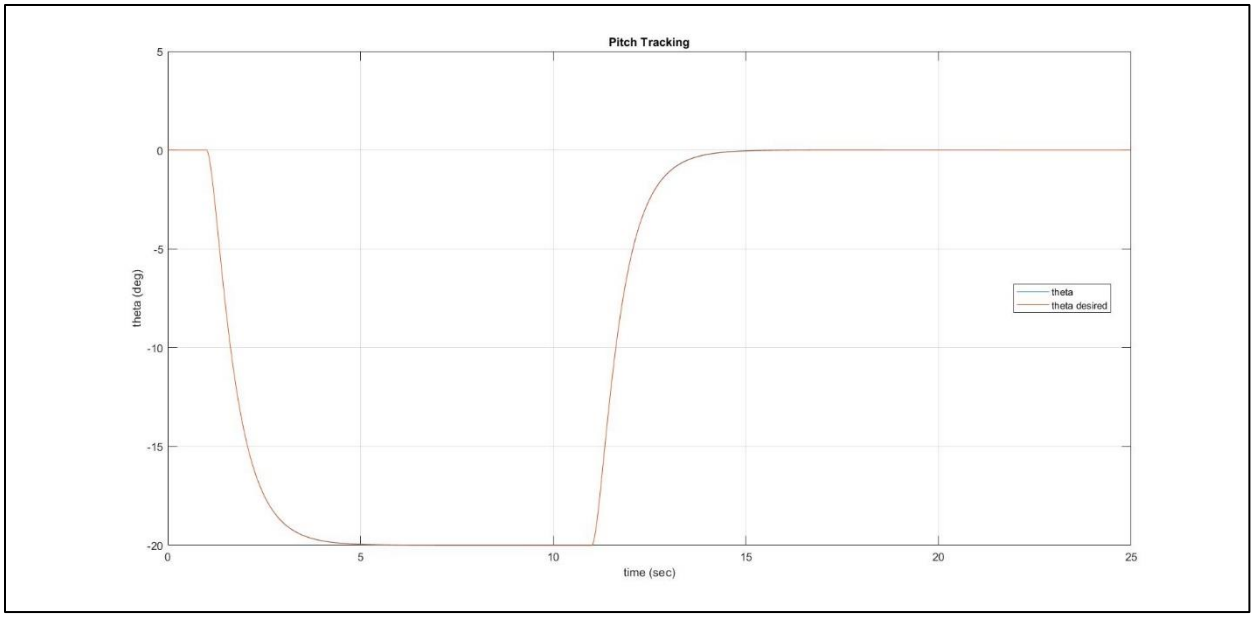


Figure 15: PID, Original; Pitch Tracking

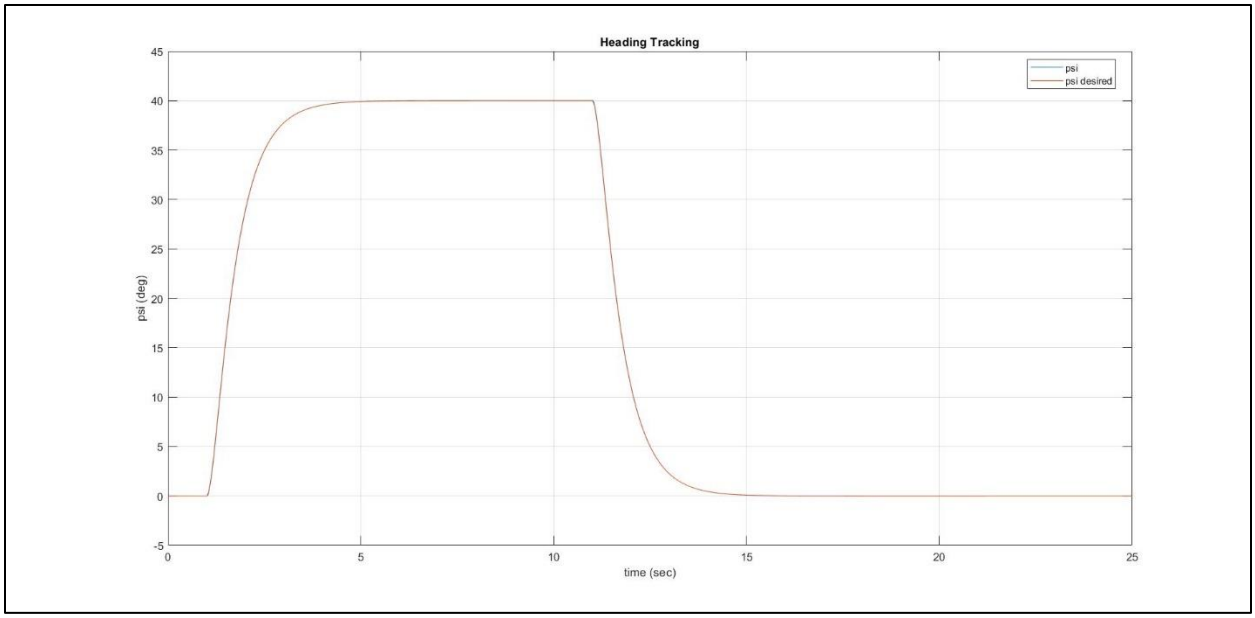


Figure 16: PID, Original; Yaw Tracking

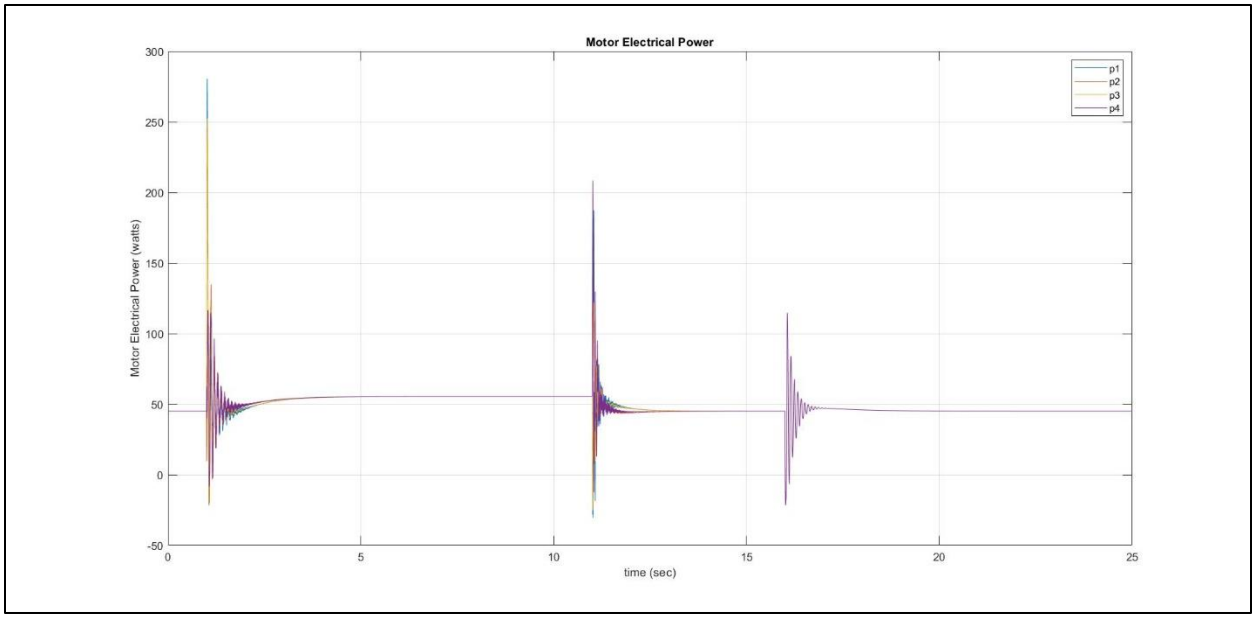


Figure 17: PID, Original; Electrical Power

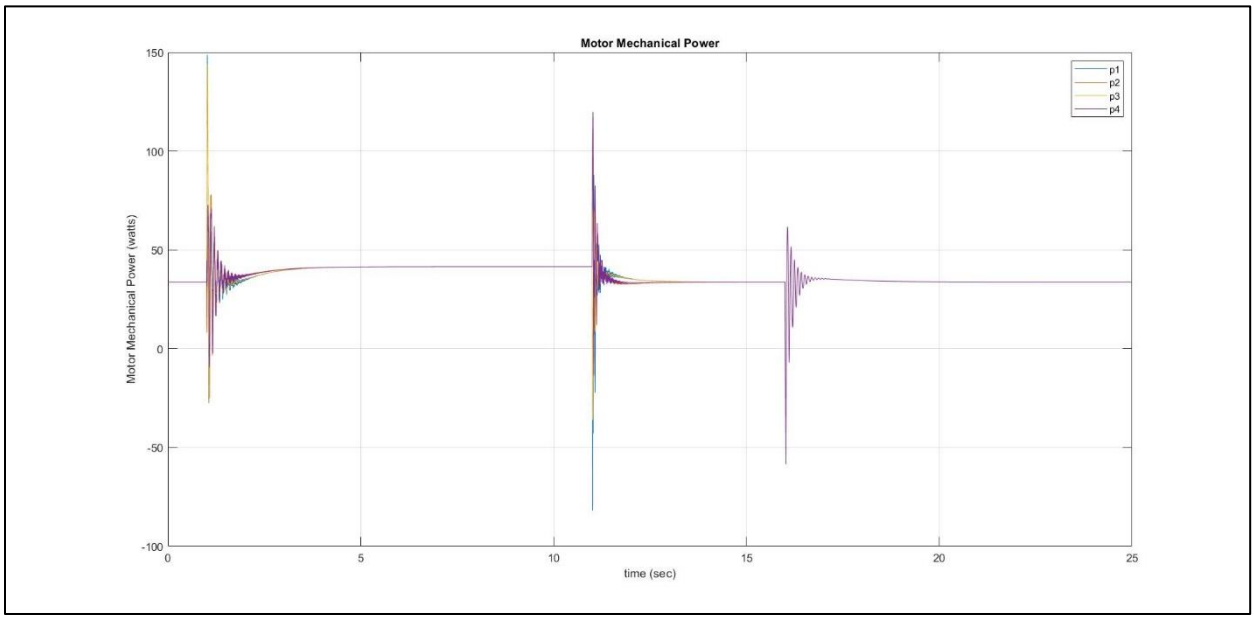


Figure 18: PID, Original; Mechanical Power

Now the performance of the MFSMC algorithm is presented:

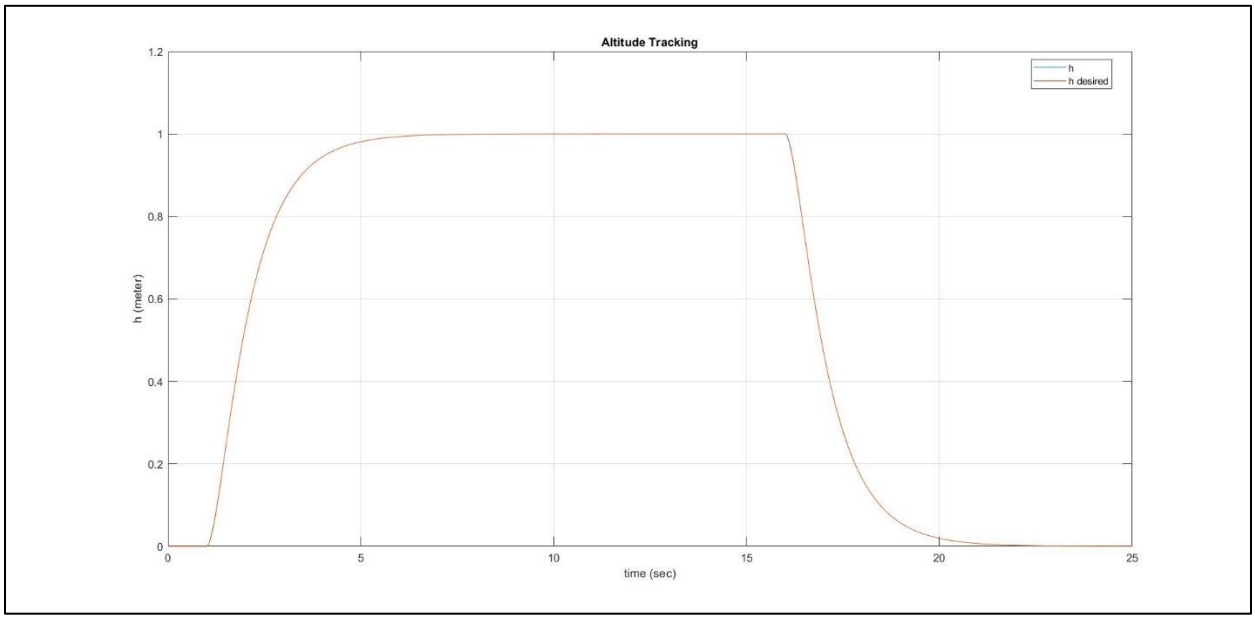


Figure 19: MFSMC, Original; Altitude Tracking

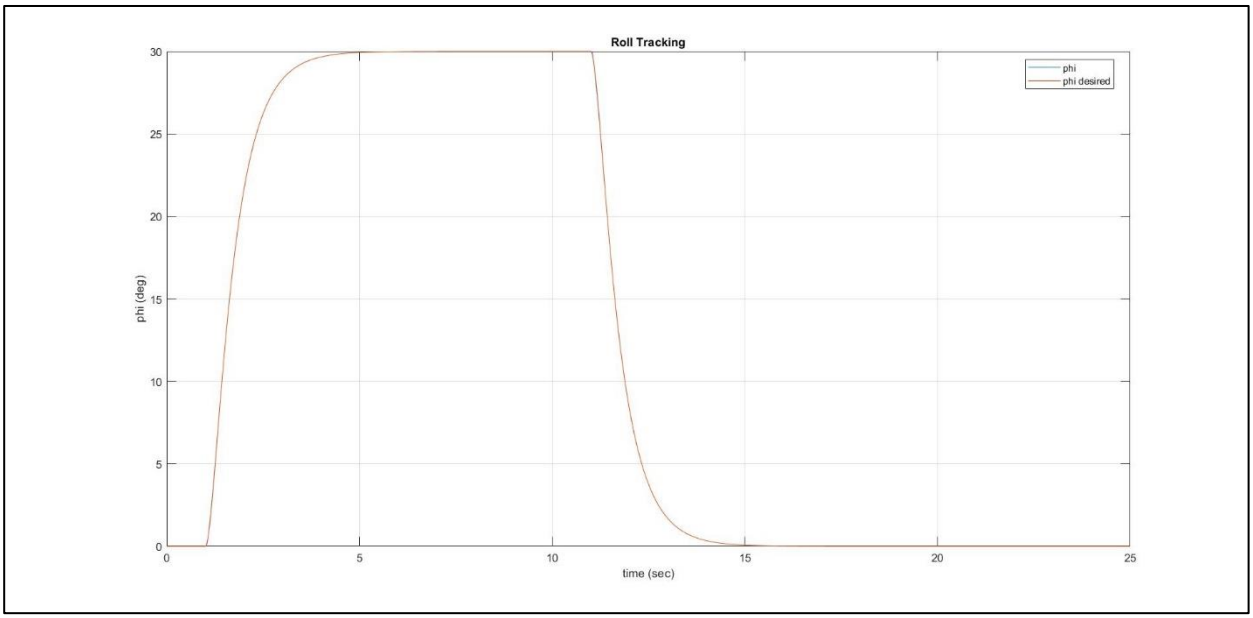


Figure 20: MFSMC, Original; Roll Tracking

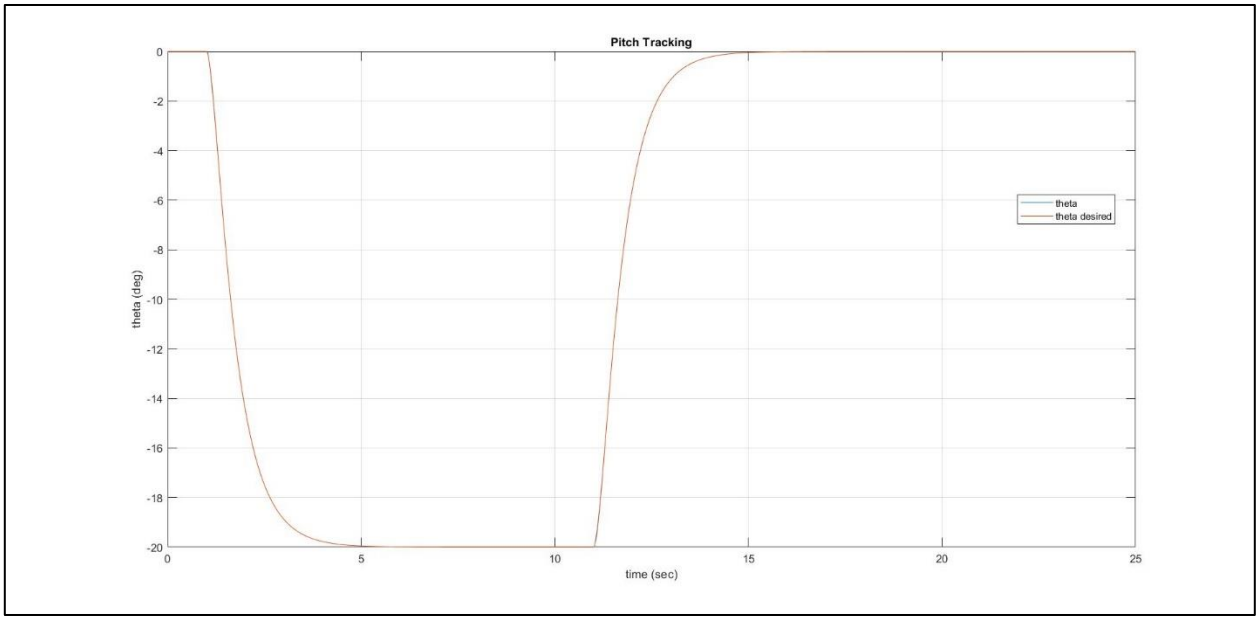


Figure 21: MFSMC, Original; Pitch Tracking

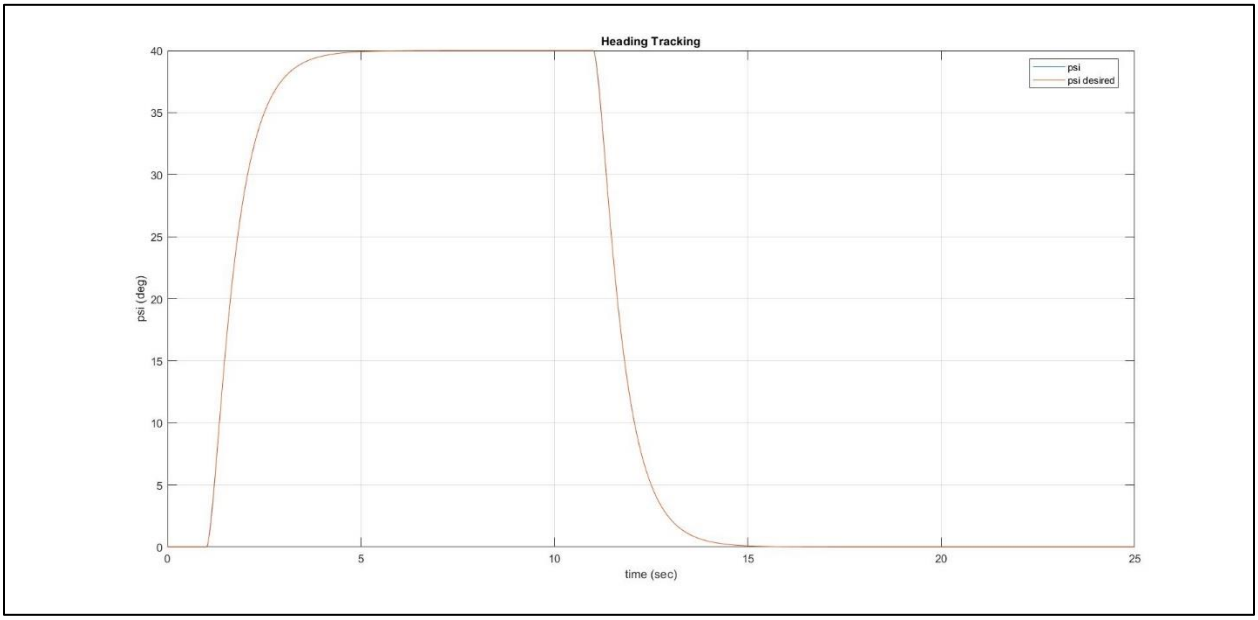


Figure 22: MFSMC, Original; Yaw Tracking

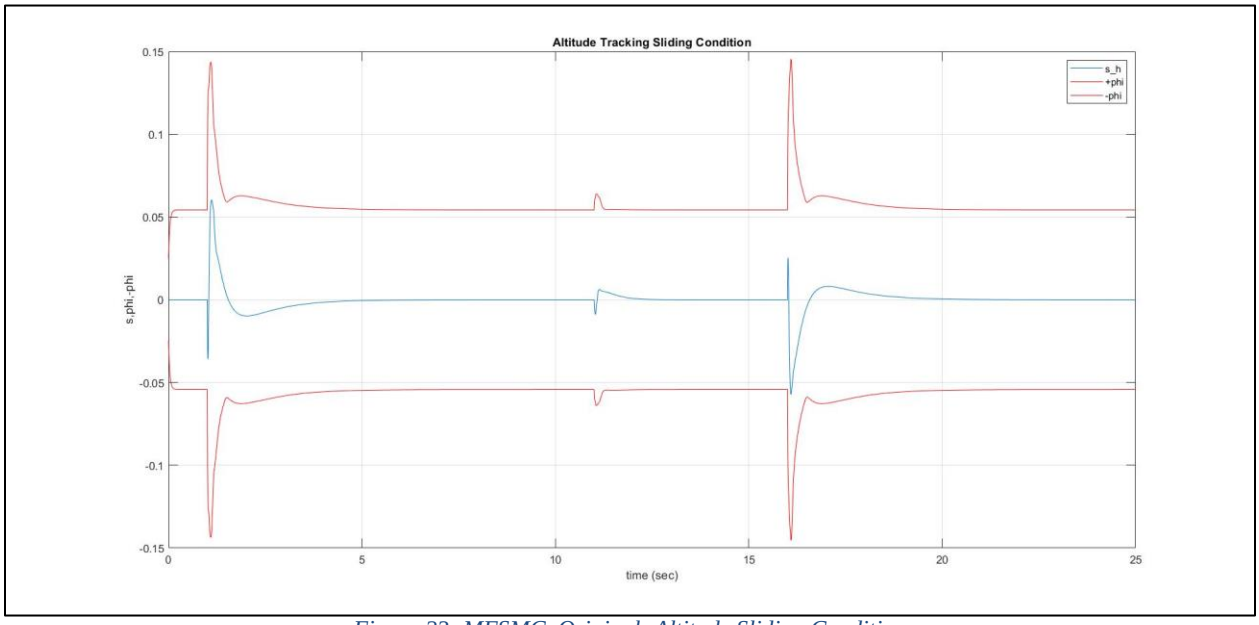


Figure 23: MFSMC, Original; Altitude Sliding Condition

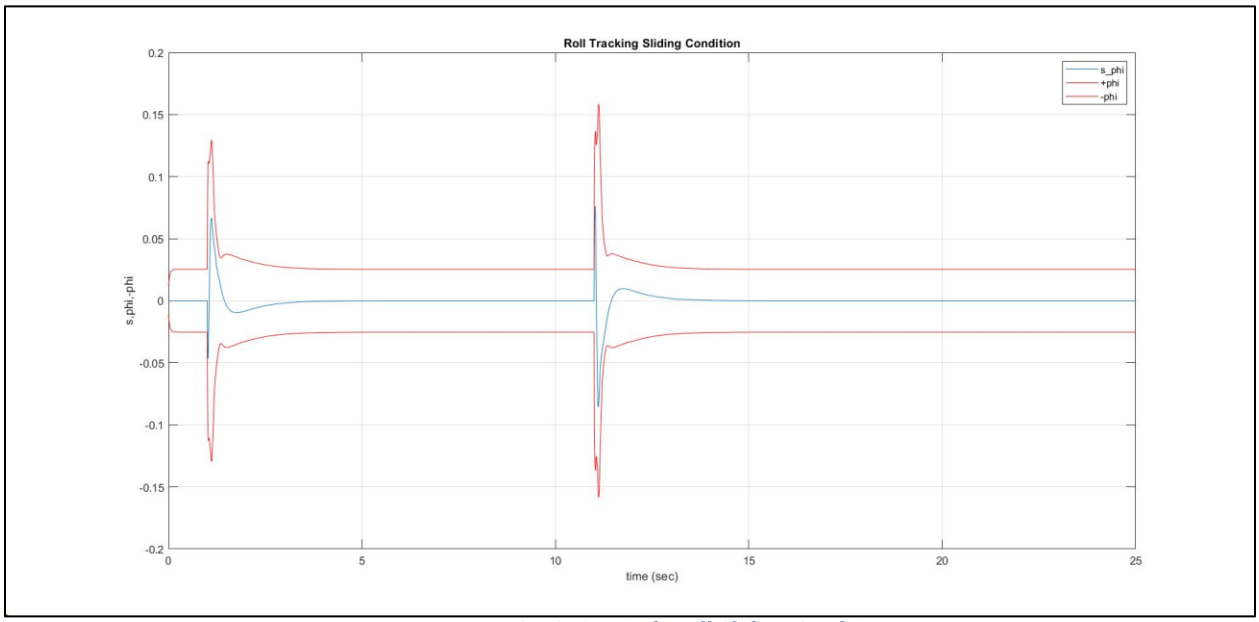


Figure 24: MFSMC, Original; Roll Sliding Condition

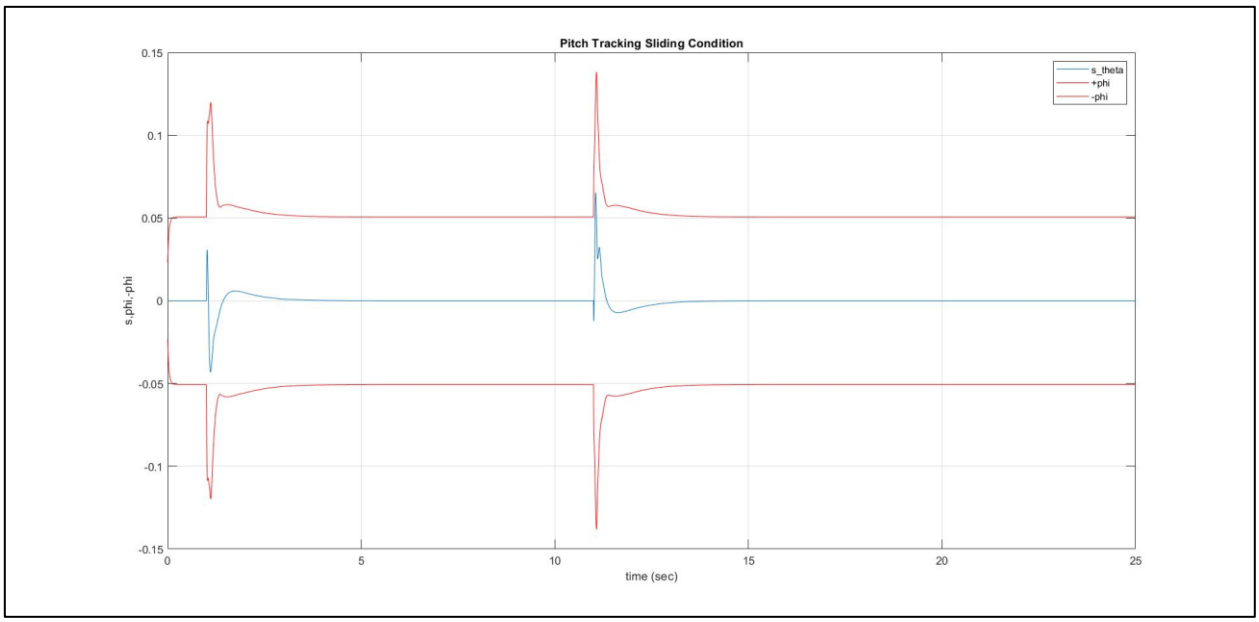


Figure 25: MFSMC, Original; Pitch Sliding Condition

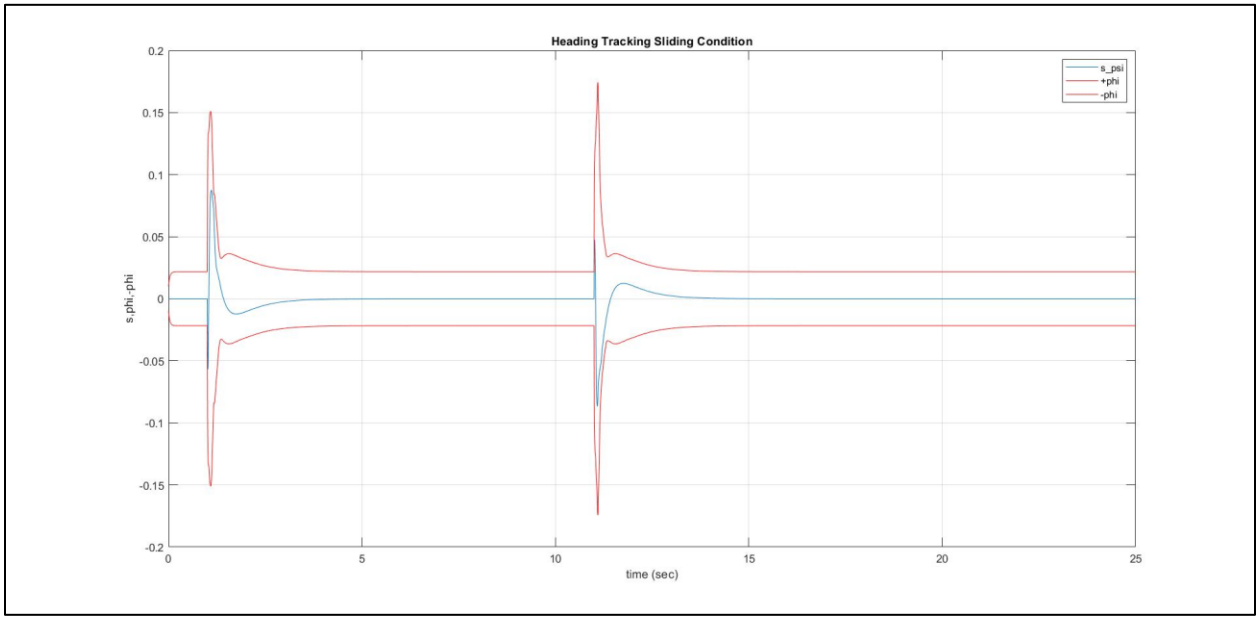


Figure 26: MFSMC, Original; Yaw Sliding Condition

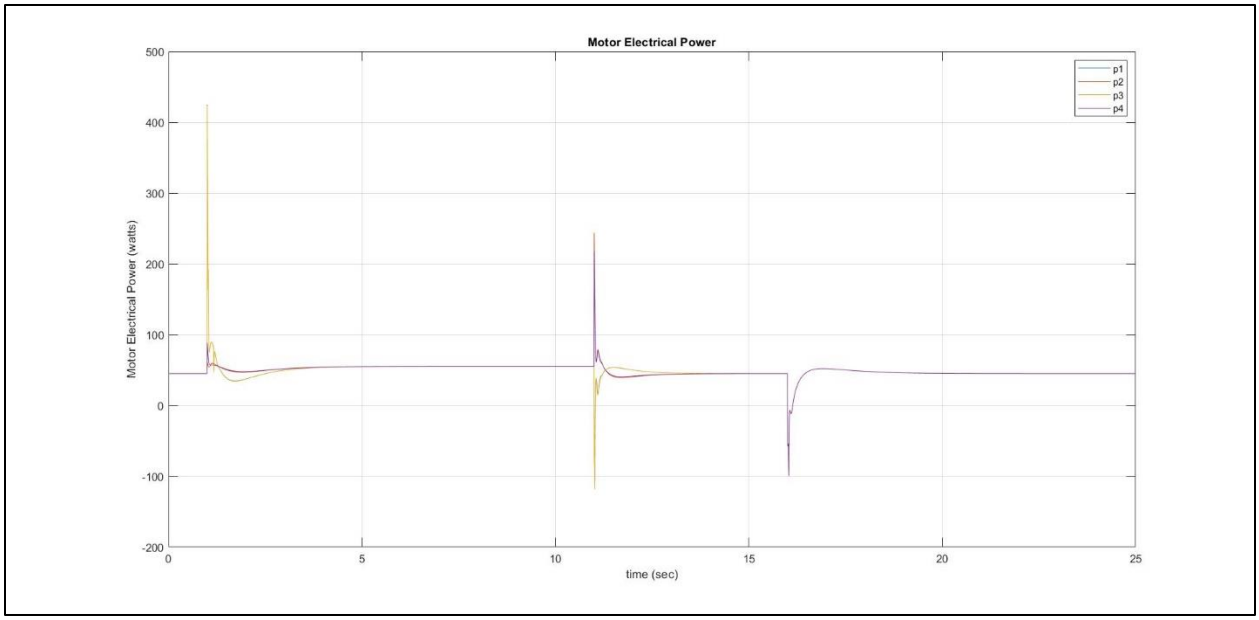


Figure 27: MFSMC, Original; Electrical Power

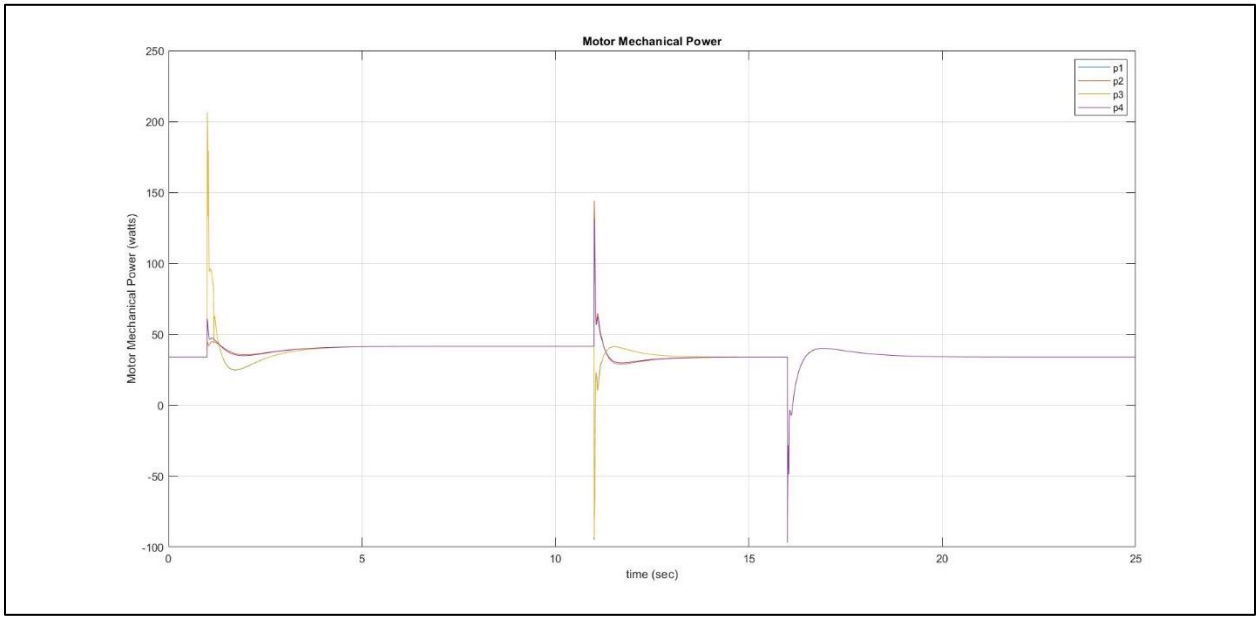


Figure 28: MFSMC, Original; Mechanical Power

	PID	MFSMC
Altitude RMS (deg)	4.13906E-04	2.12795E-04
Roll RMS (deg)	1.44825E-02	1.28304E-02
Pitch RMS (deg)	8.89239E-03	7.81558E-03
Yaw RMS (deg)	3.92335E-02	1.49953E-02
Electrical Power Average (W)	196.4	194.6
Mechanical Power Average (W)	146.6	146.4
Efficiency	74.64%	75.23%

Table 6: Results with Original Parameters

It can be concluded from Table 6 that the MFSMC has better tracking for altitude, roll, pitch, and yaw, and it used less average power than that of the PID controller, using the craft's original parameters. The MFSMC sliding condition was satisfied for altitude, roll, pitch, and yaw control.

5.3.2 Double the Craft's Mass

In this section, the craft of only the mass is doubled while using the exact same controllers in the previous simulation. The PID is presented first:

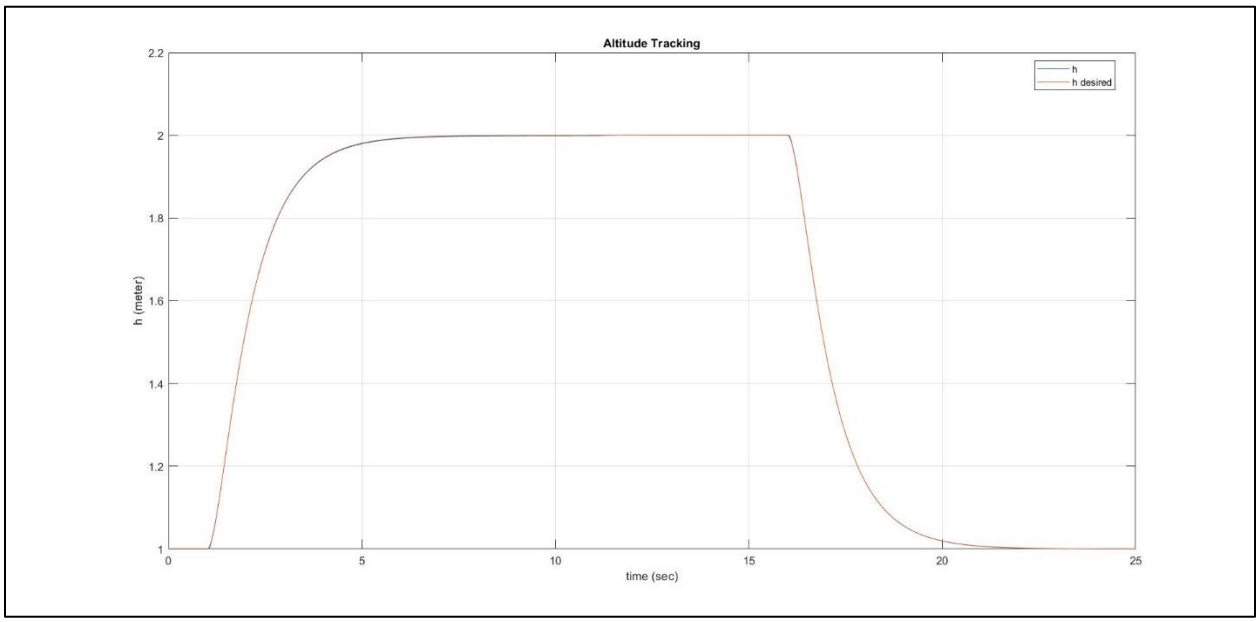


Figure 29: PID, Double Mass; Altitude Tracking

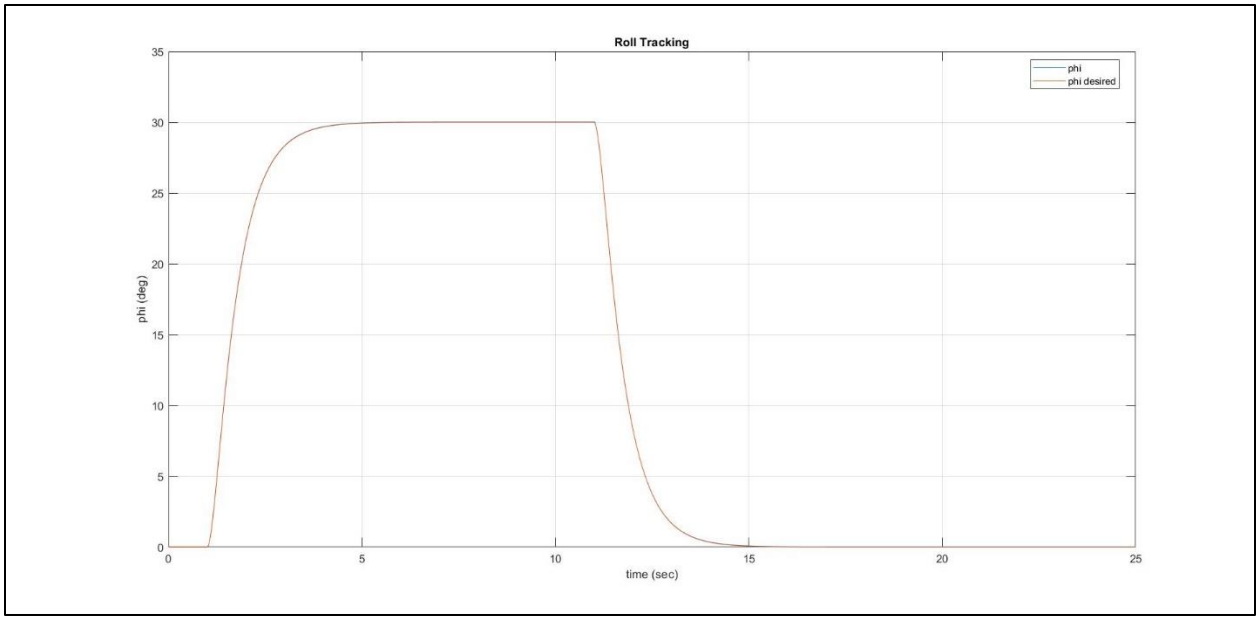


Figure 30: PID, Double Mass; Roll Tracking

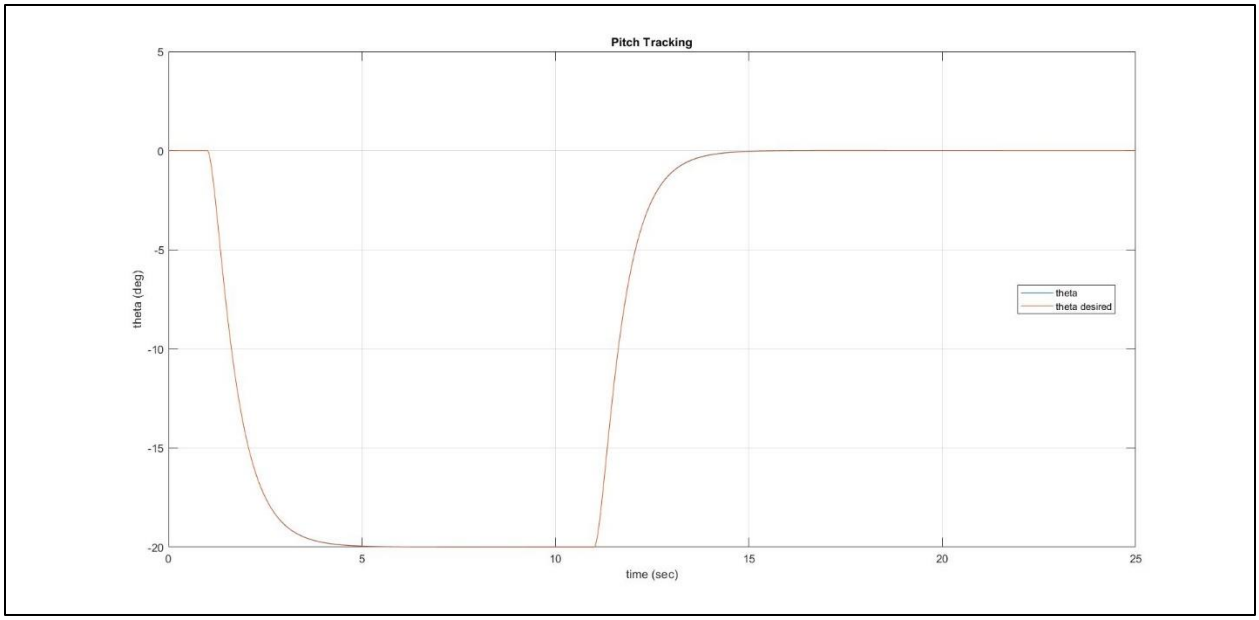


Figure 31: PID, Double Mass; Pitch Tracking

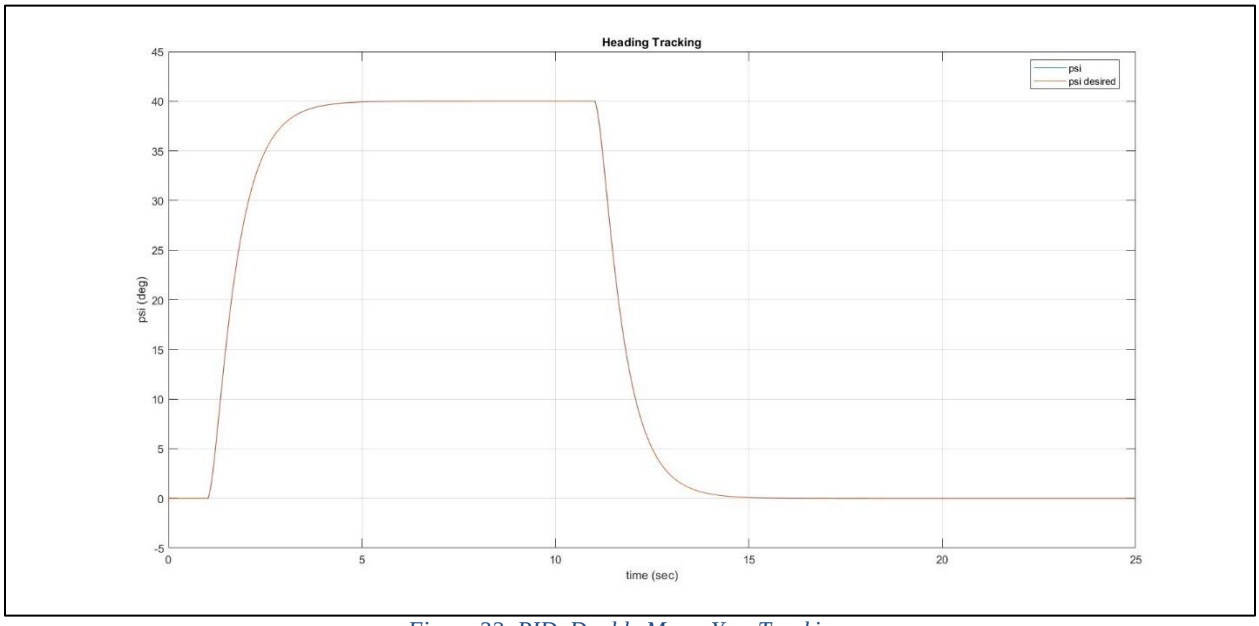


Figure 32: PID, Double Mass; Yaw Tracking

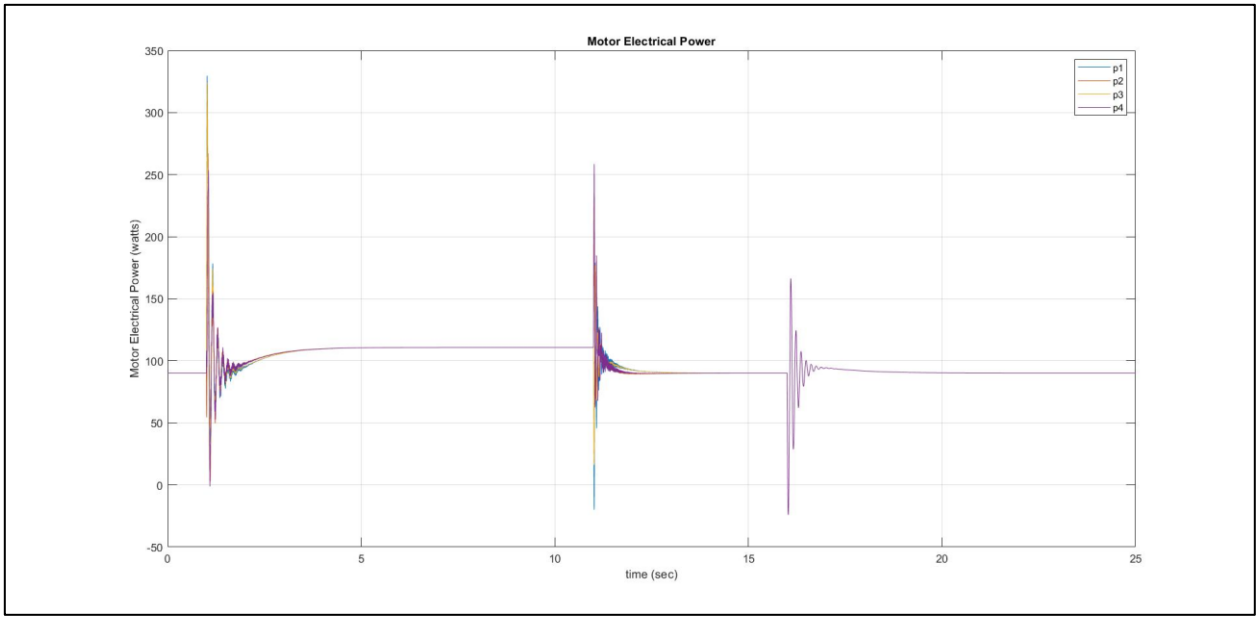


Figure 33: PID, Double Mass; Electrical Power

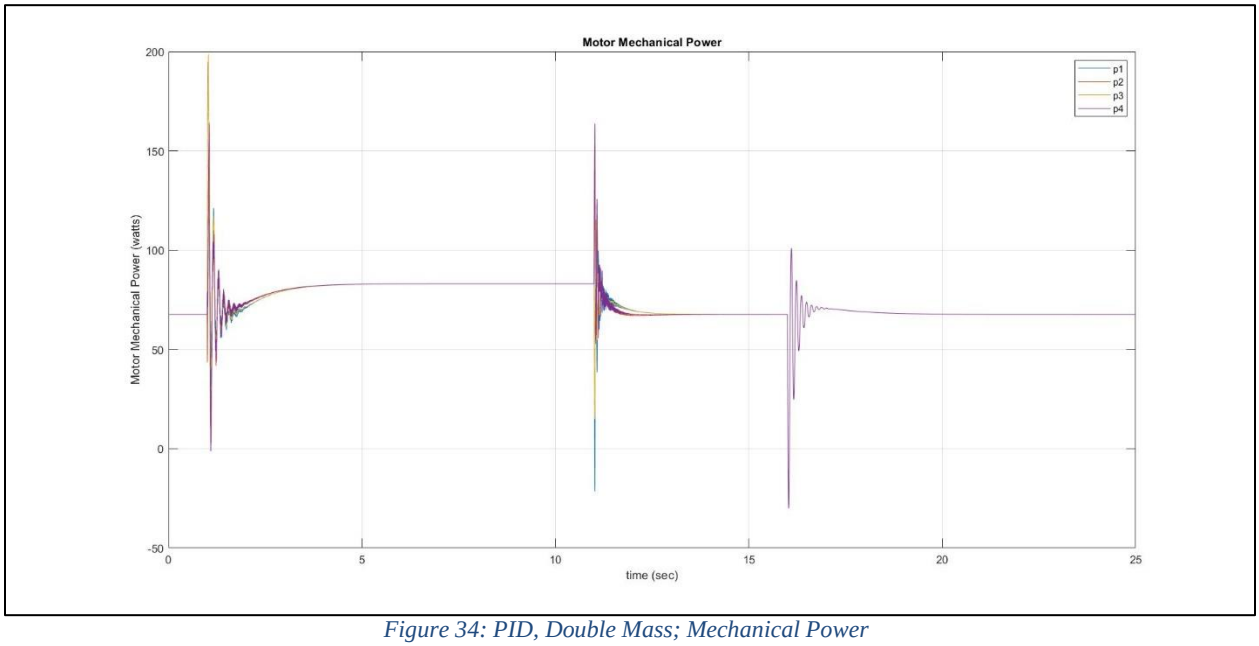


Figure 34: PID, Double Mass; Mechanical Power

Now the performance of the MFSMC algorithm is presented:

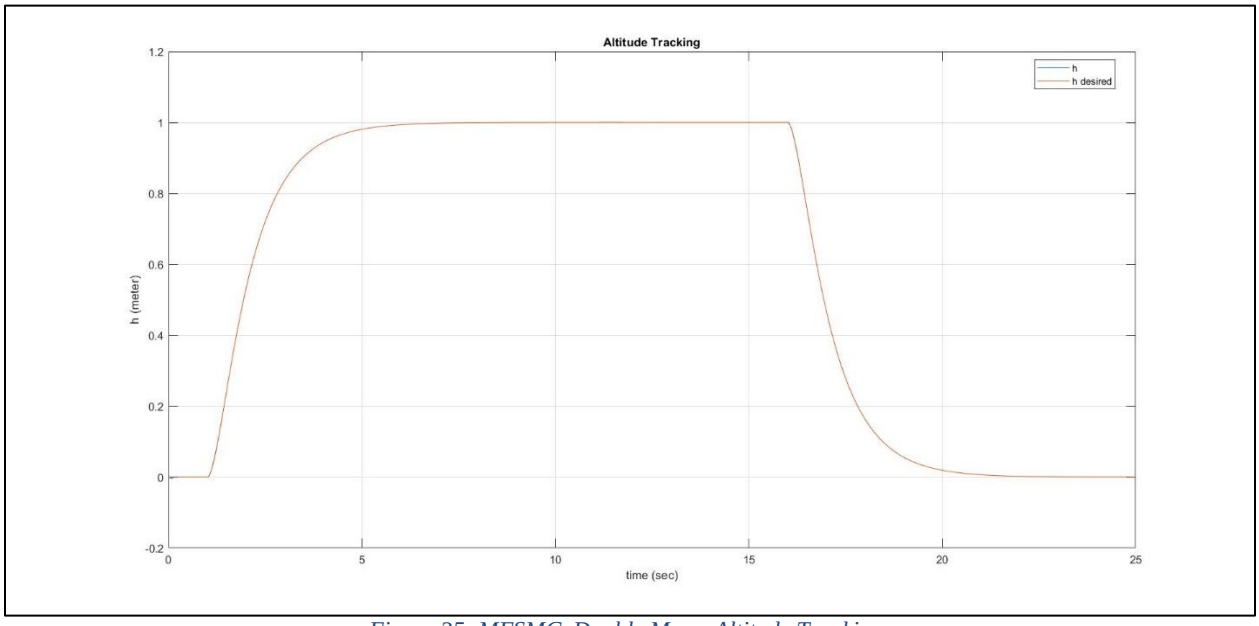


Figure 35: MFSMC, Double Mass; Altitude Tracking

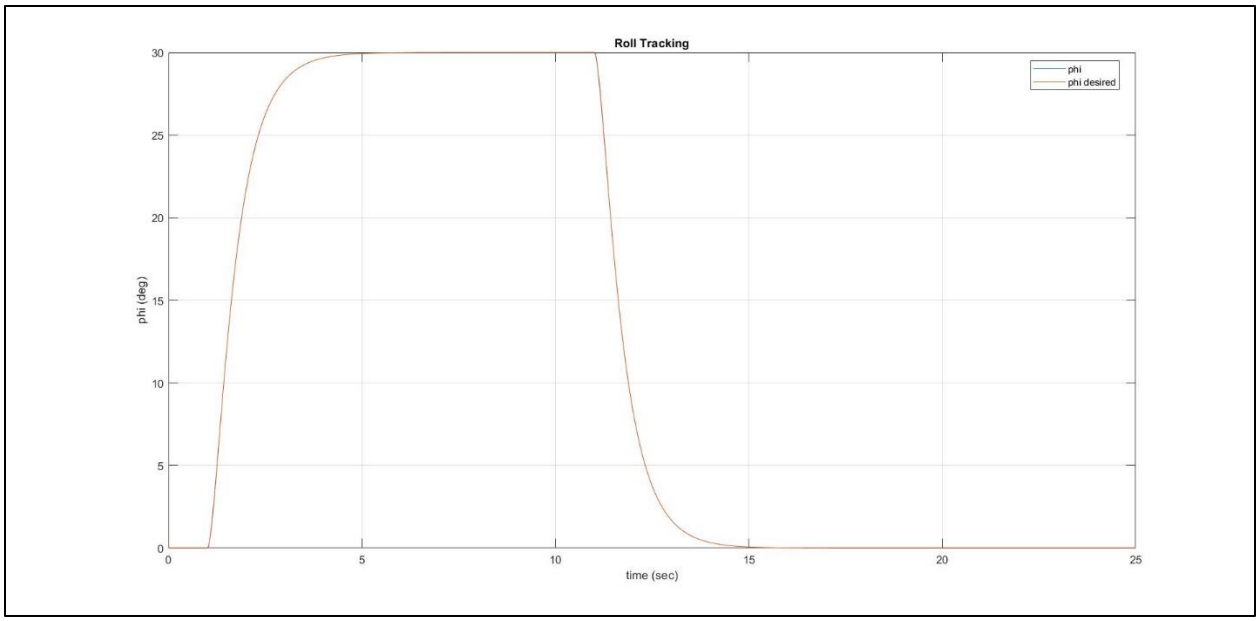


Figure 36: MFSMC, Double Mass; Roll Tracking

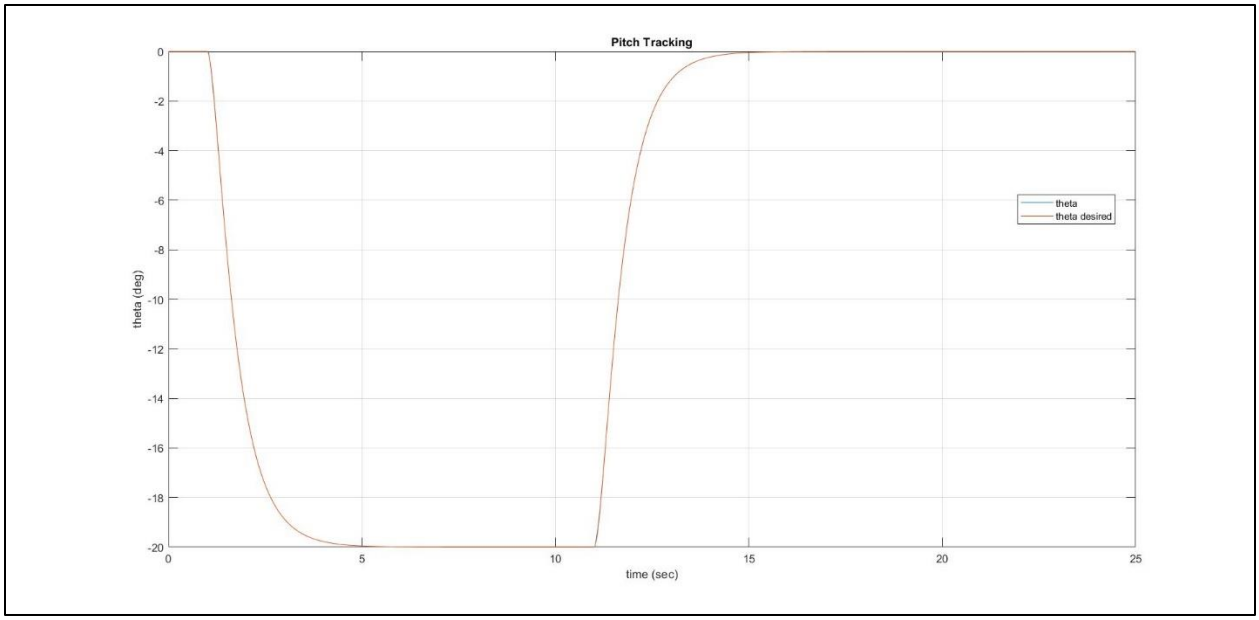


Figure 37: MFSMC, Double Mass; Pitch Tracking

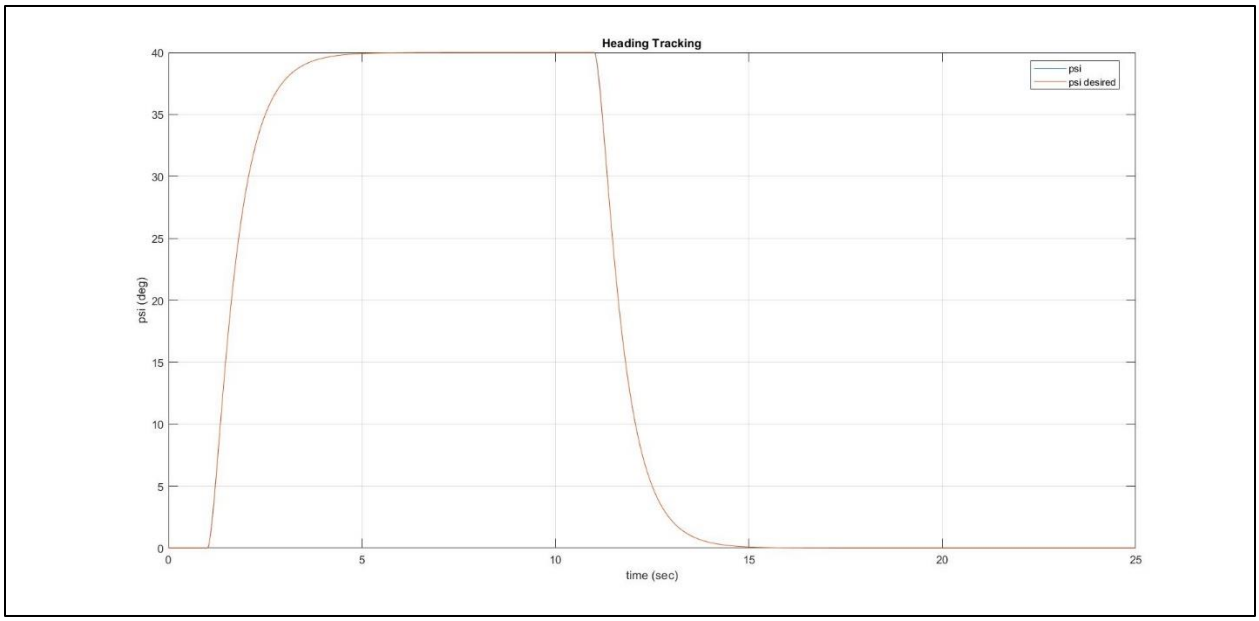


Figure 38: MFSMC, Double Mass; Yaw Tracking

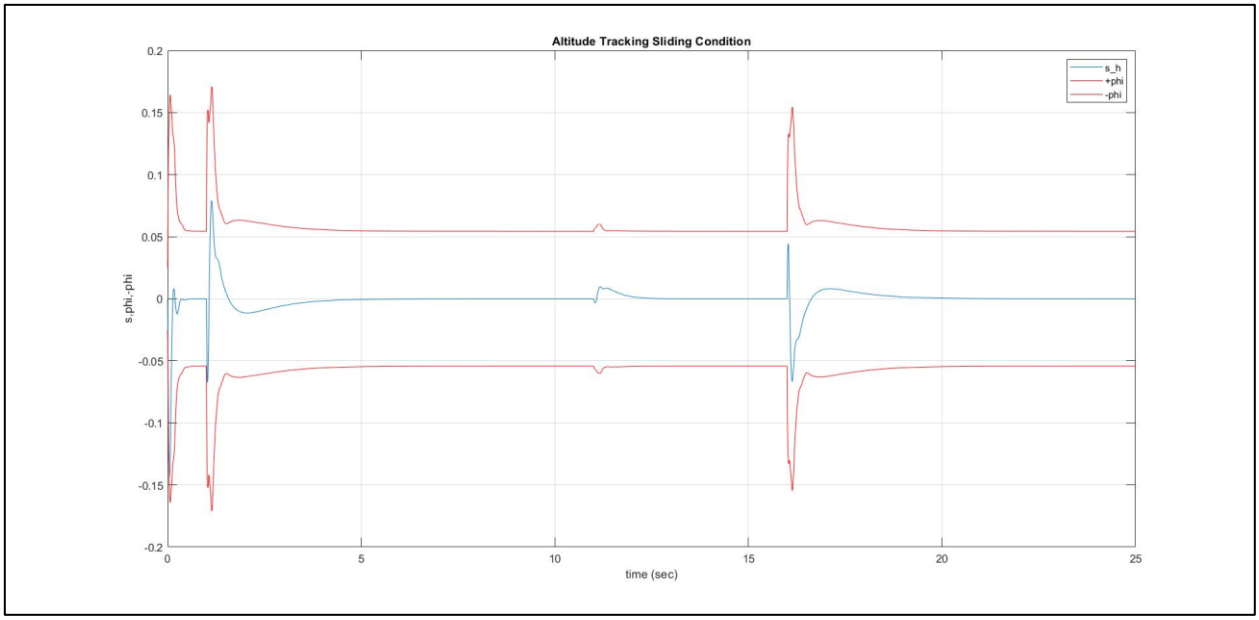


Figure 39: MFSMC, Double Mass; Altitude Sliding Condition

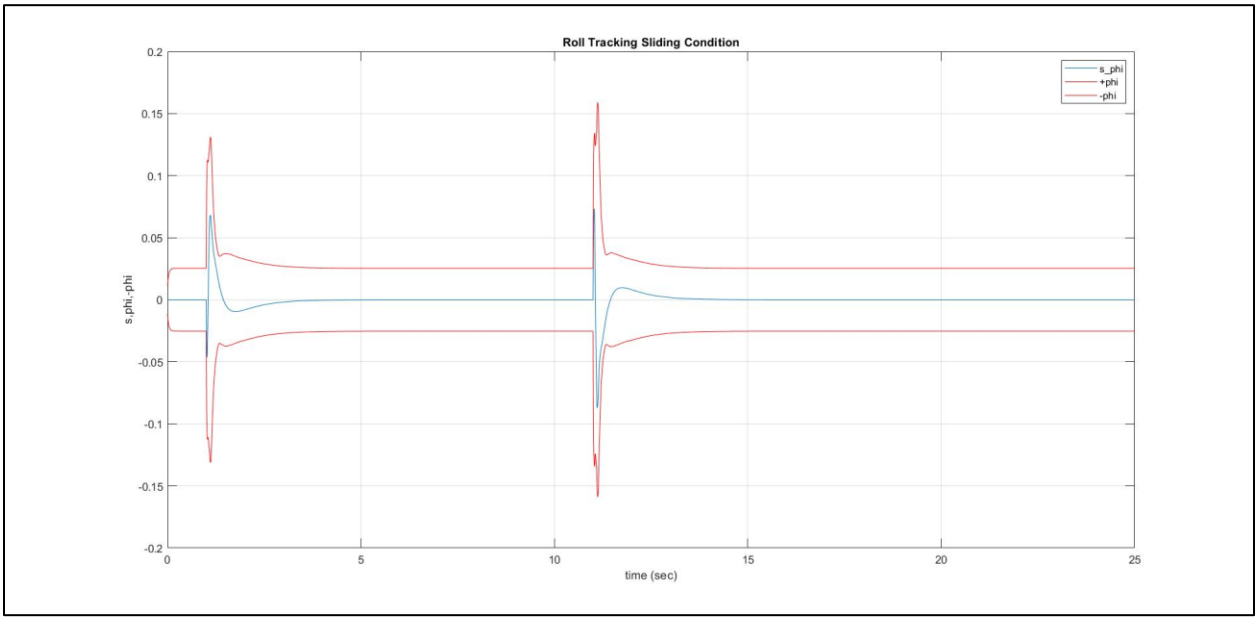


Figure 40: MFSMC, Double Mass; Roll Sliding Condition

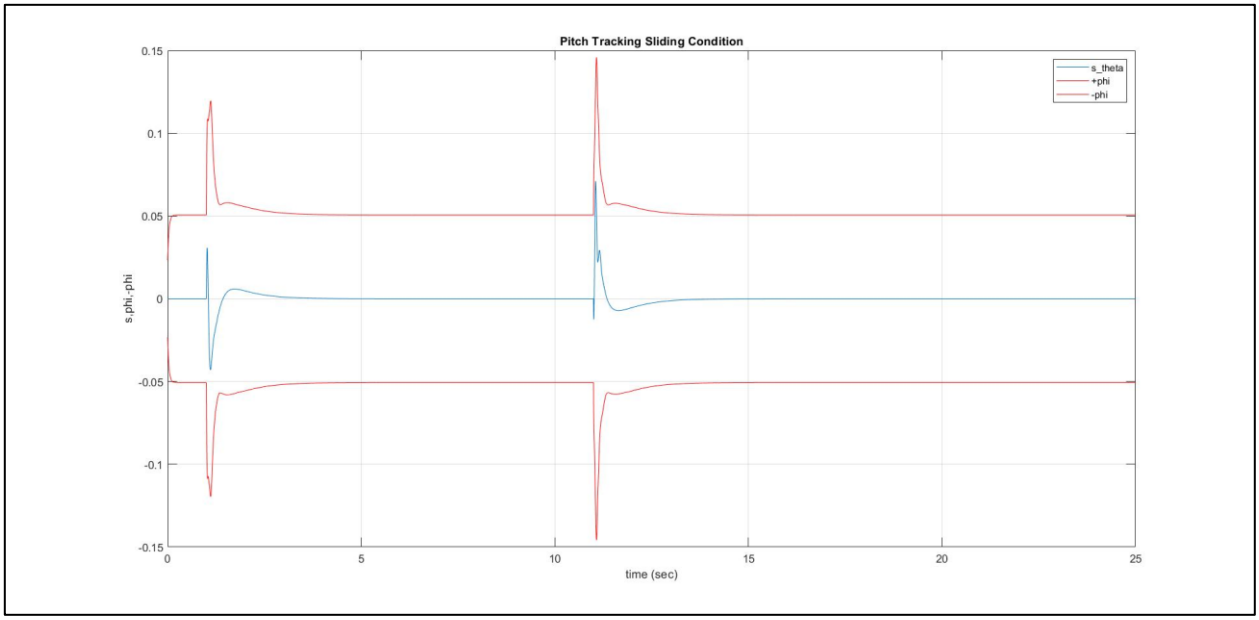


Figure 41: MFSMC, Double Mass; Pitch Sliding Condition

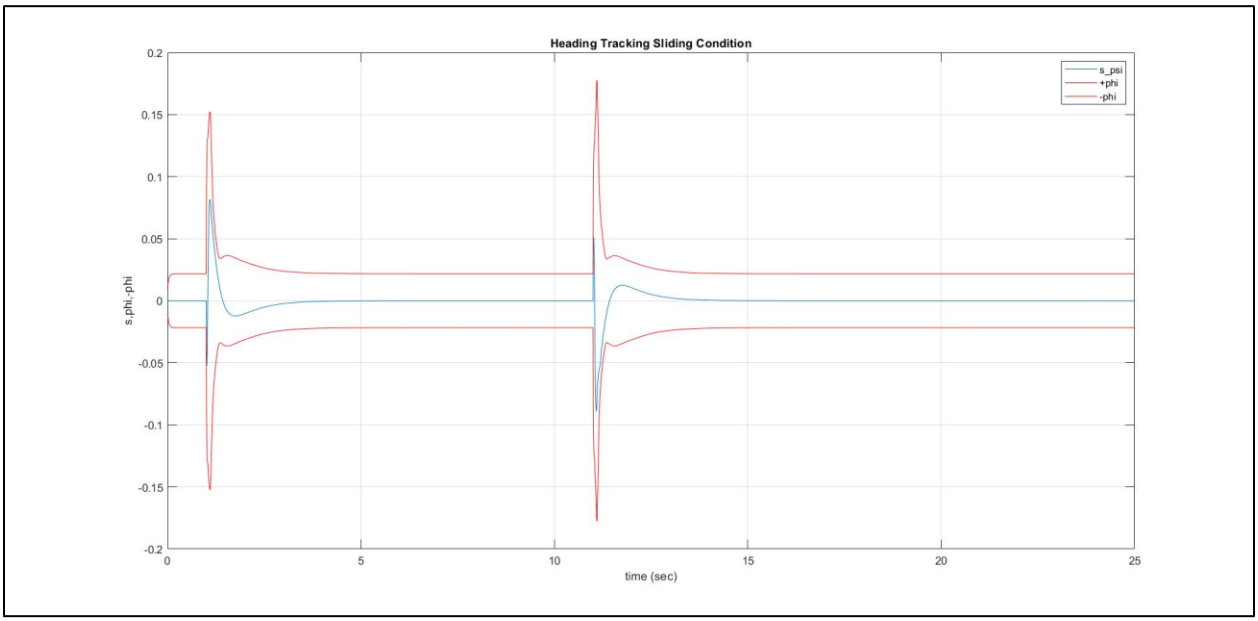


Figure 42: MFSMC, Double Mass; Yaw Sliding Condition

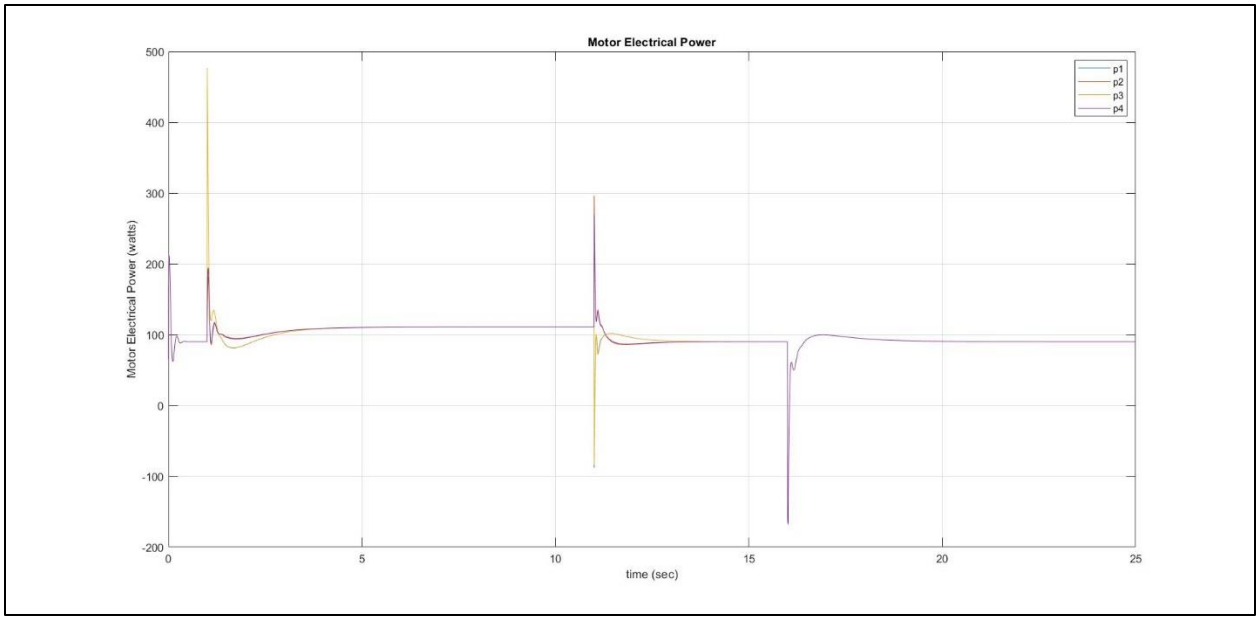


Figure 43: MFSMC, Double Mass; Electrical Power

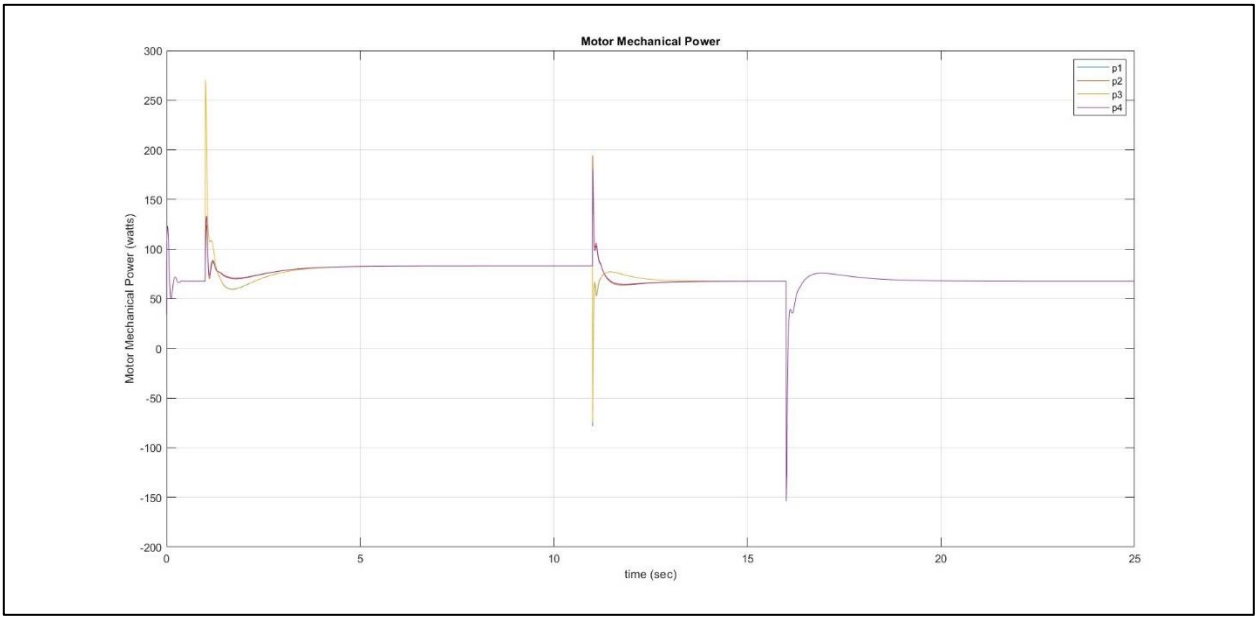


Figure 44: MFSMC, Double Mass; Mechanical Power

	PID	MFSMC
Altitude RMS (deg)	8.29678E-04	3.14351E-04
Roll RMS (deg)	1.83692E-02	1.27950E-02
Pitch RMS (deg)	8.91206E-03	7.86103E-03
Yaw RMS (deg)	3.90046E-02	1.46493E-02
Electrical Power Average (W)	391.7	391.1
Mechanical Power Average (W)	293.2	293.0
Efficiency	74.85%	74.92%

Table 7: Results using Double Mass

It can be concluded from Table 7 that the MFSMC has better tracking for altitude, roll, pitch, and yaw, and it used less average power than that of the PID controller, using double the craft's original mass. The MFSMC sliding condition was satisfied for altitude, roll, pitch, and yaw control.

5.3.3 Double the Craft's Moments of Inertia

In this section, the craft of only the moments of inertia are doubled while using the exact same controllers in the previous simulations. The performance of the PID controller is presented first:

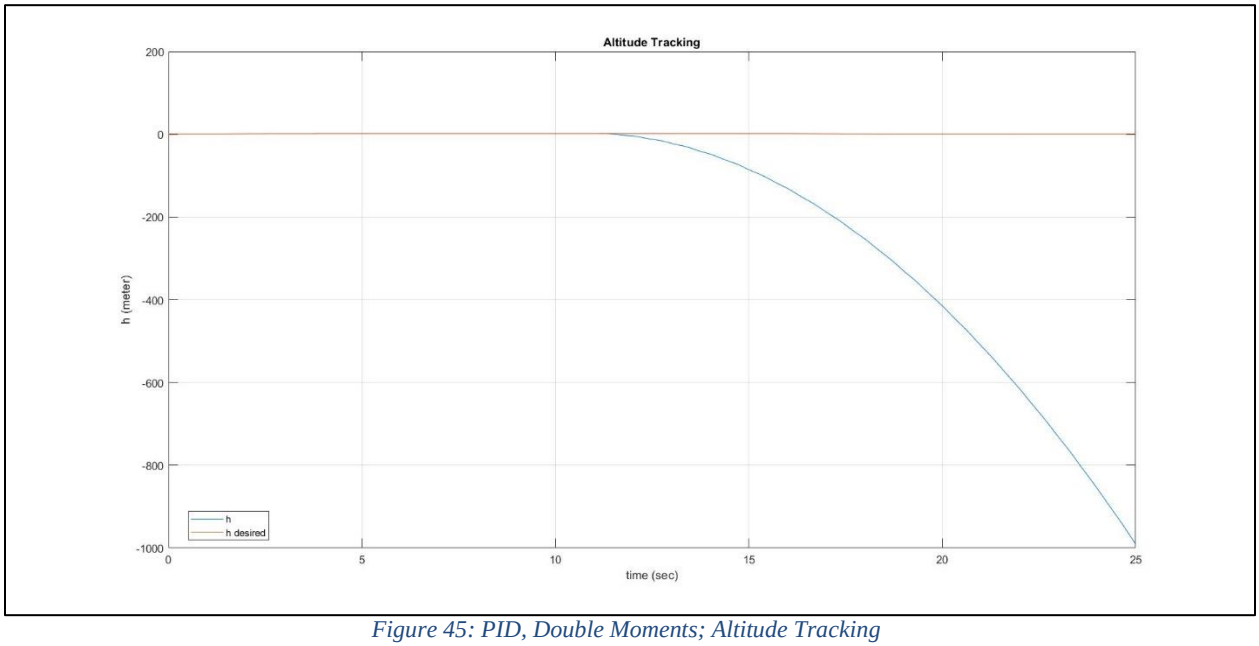


Figure 45: PID, Double Moments; Altitude Tracking

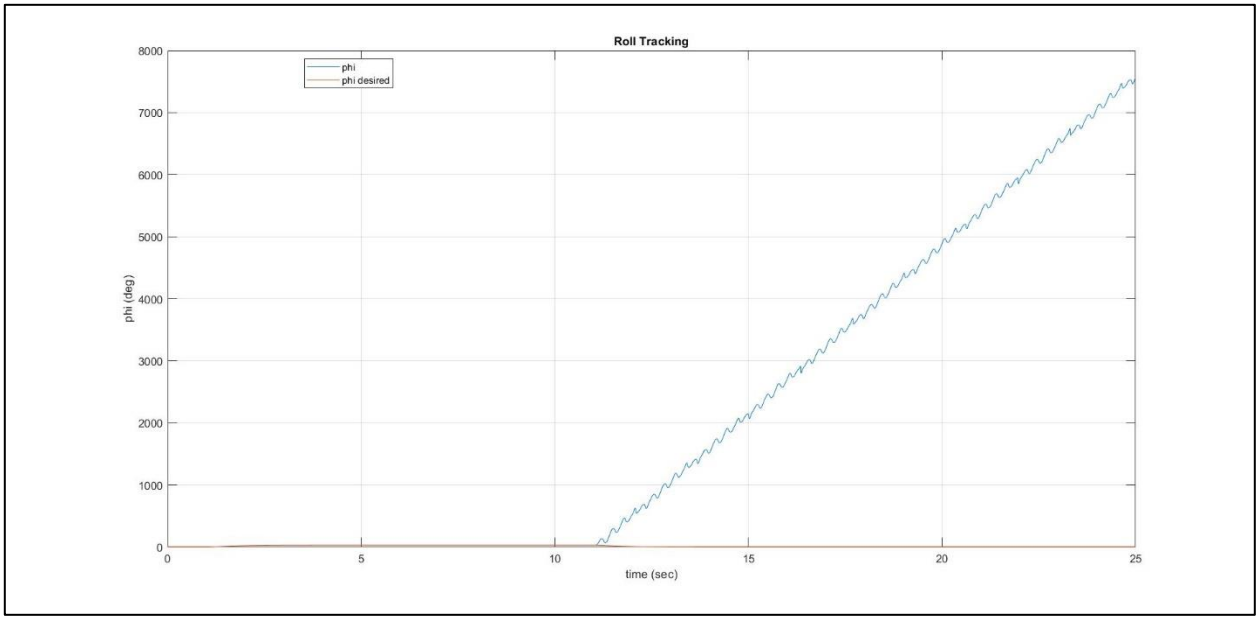


Figure 46: PID, Double Moments; Roll Tracking

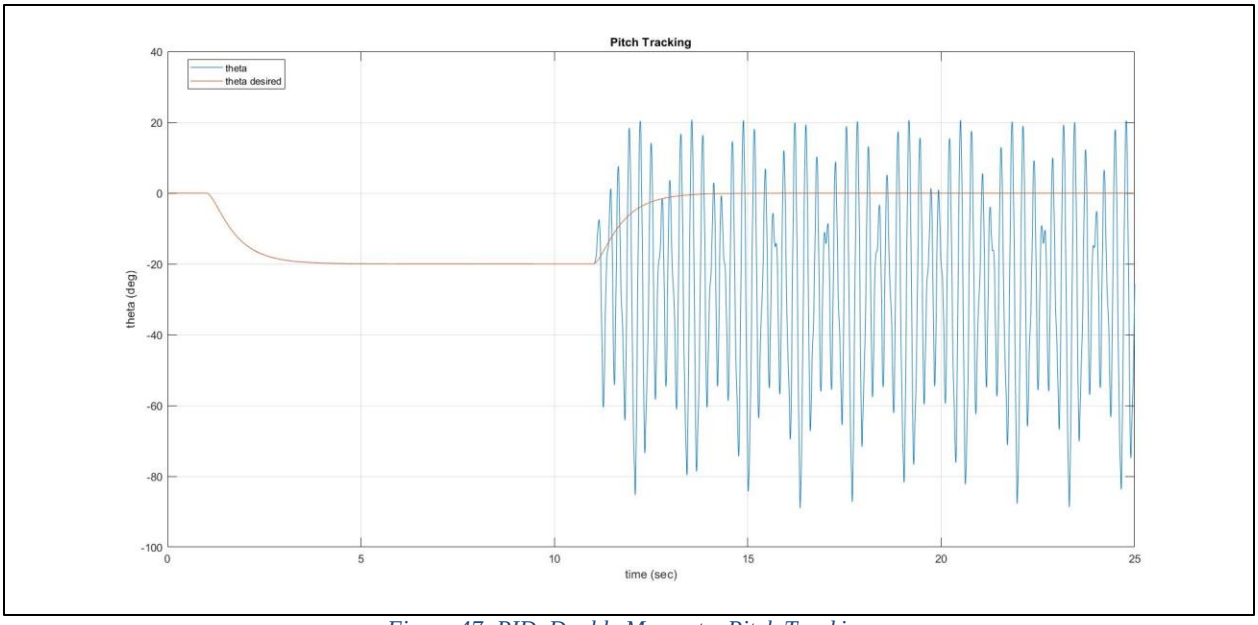


Figure 47: PID, Double Moments; Pitch Tracking

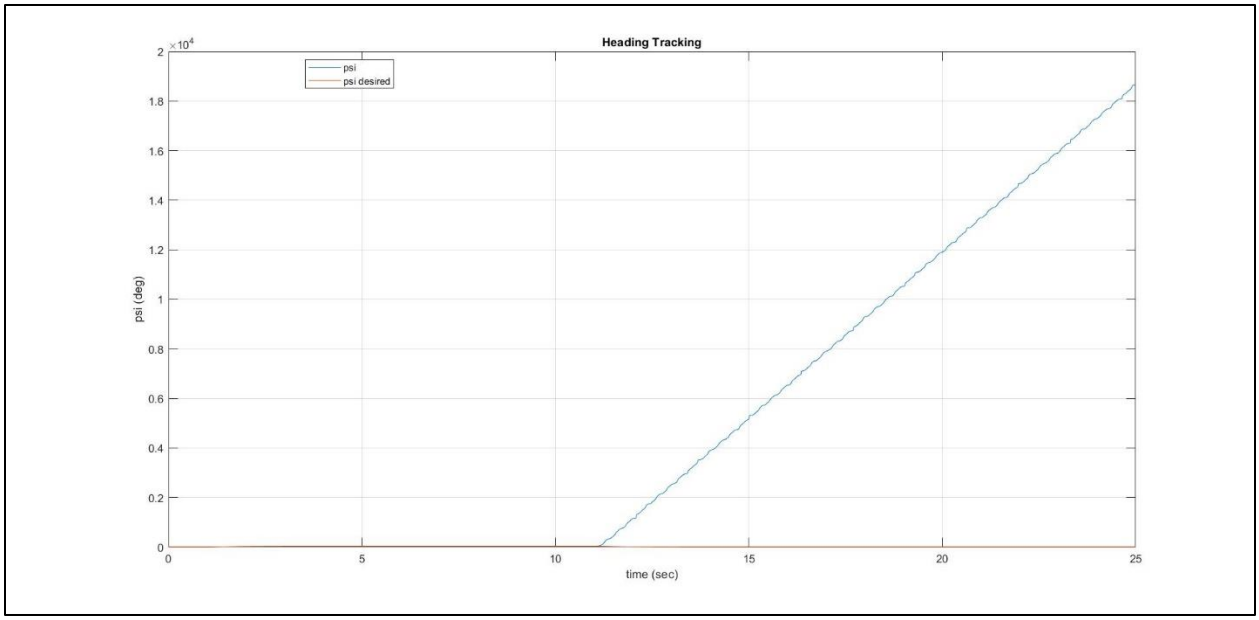


Figure 48: PID, Double Moments; Yaw Tracking

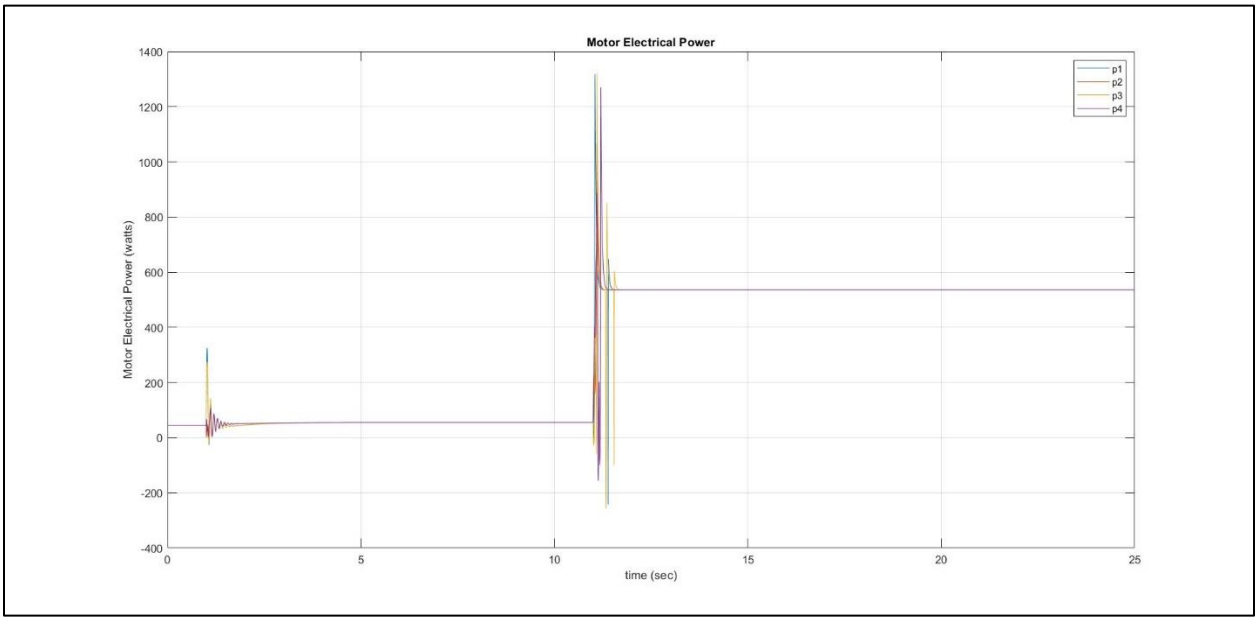


Figure 49: PID, Double Moments; Electrical Power

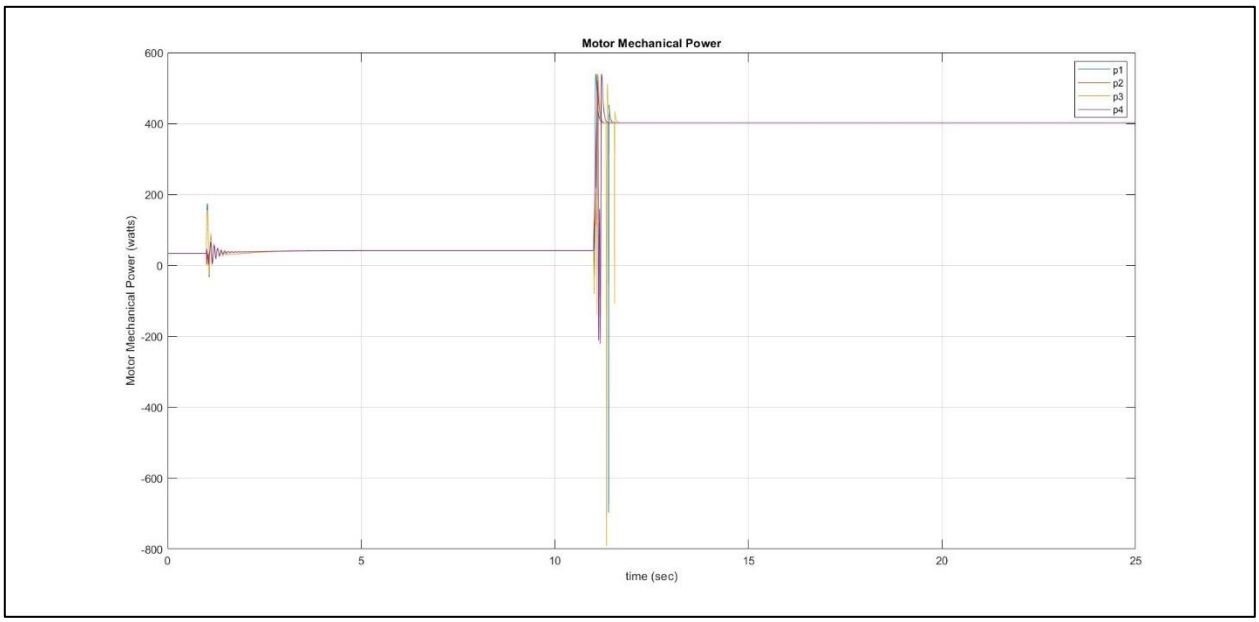


Figure 50: PID, Double Moments; Mechanical Power

Now the performance of the MFSMC algorithm is presented:

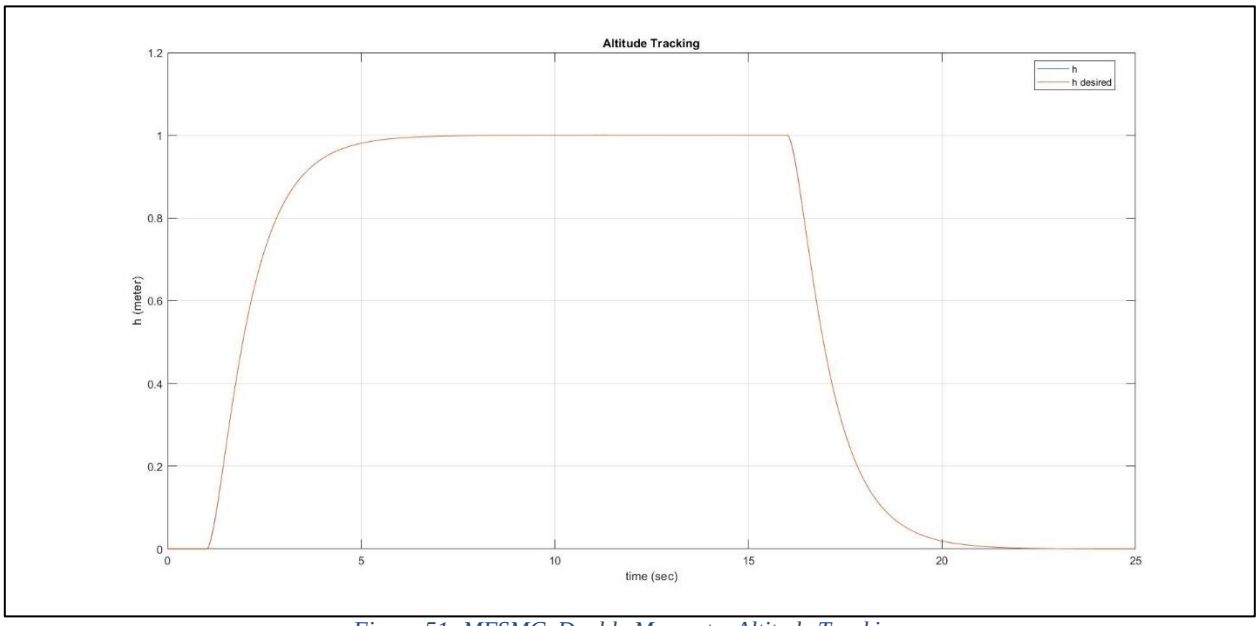


Figure 51: MFSMC, Double Moments; Altitude Tracking

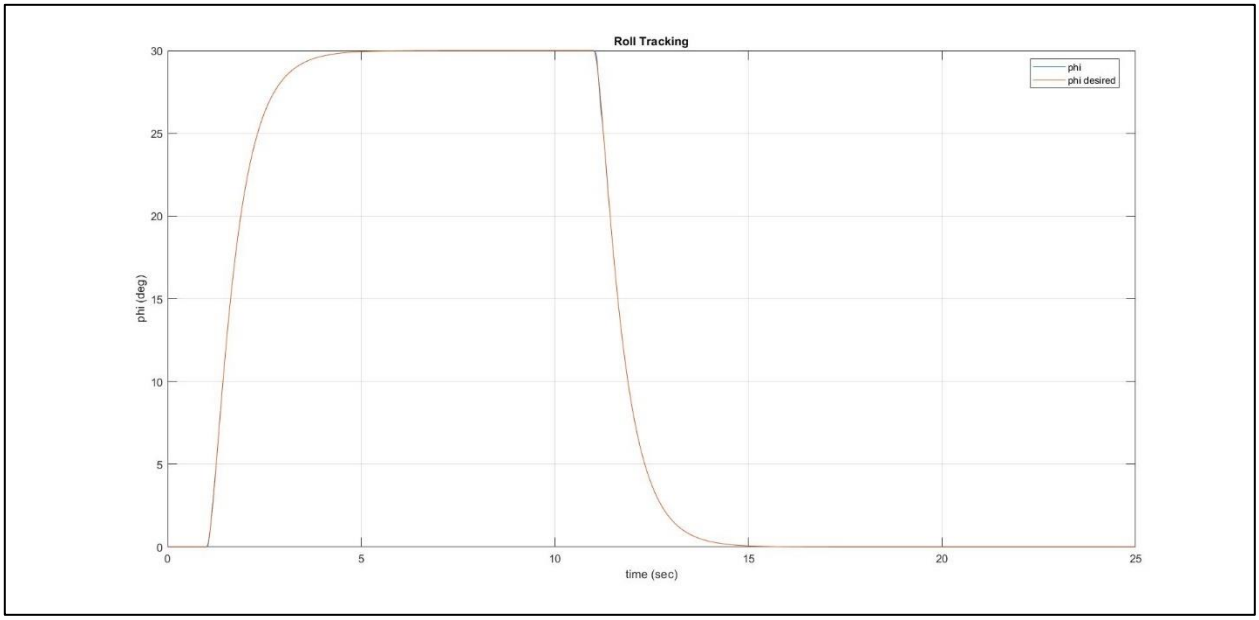


Figure 52: MFSMC, Double Moments; Roll Tracking

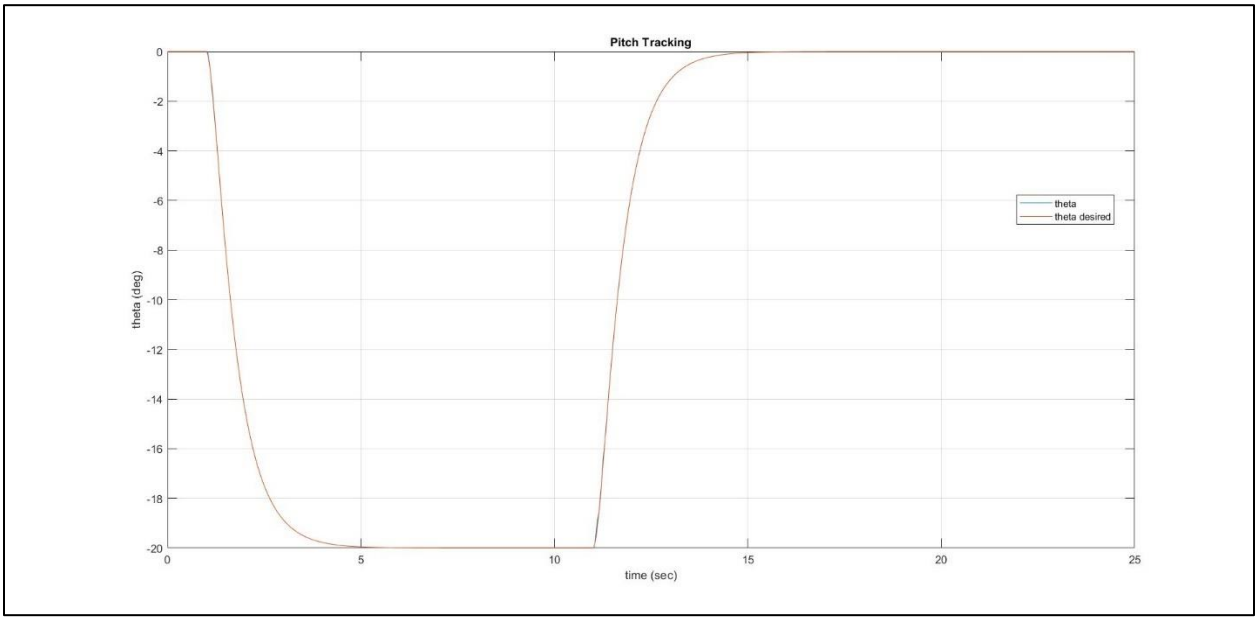


Figure 53: MFSMC, Double Moments; Pitch Tracking

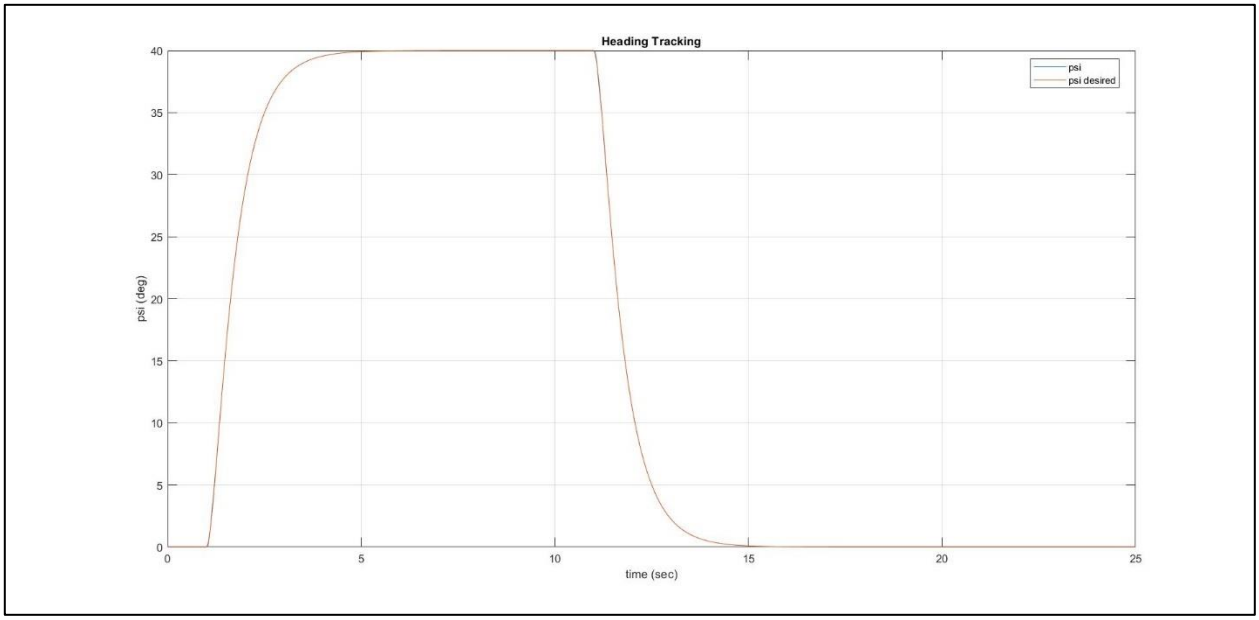


Figure 54: MFSMC, Double Moments; Yaw Tracking

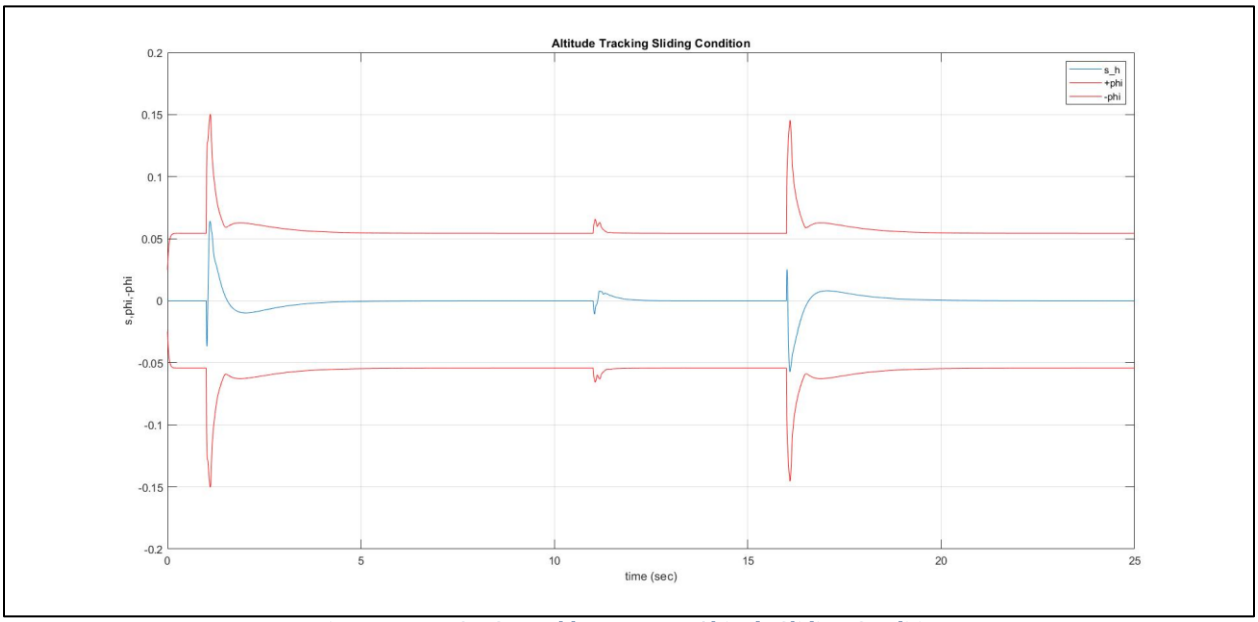


Figure 55: MFSMC, Double Moments; Altitude Sliding Condition

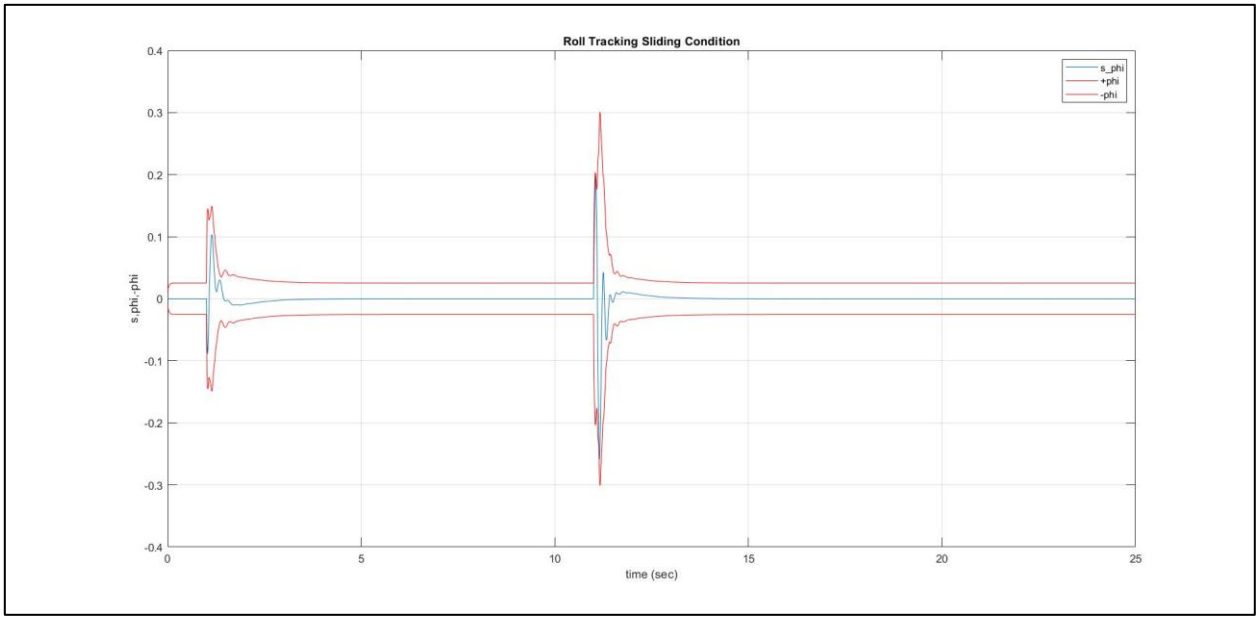


Figure 56: MFSMC, Double Moments; Roll Sliding Condition

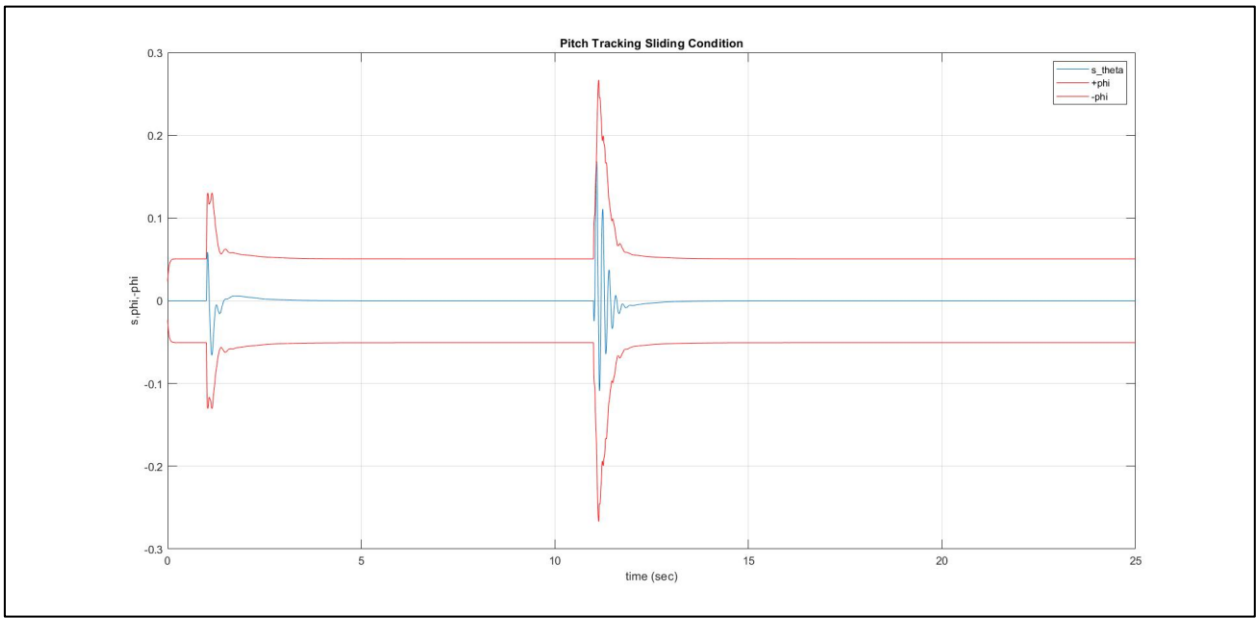


Figure 57: MFSMC, Double Moments; Pitch Sliding Condition

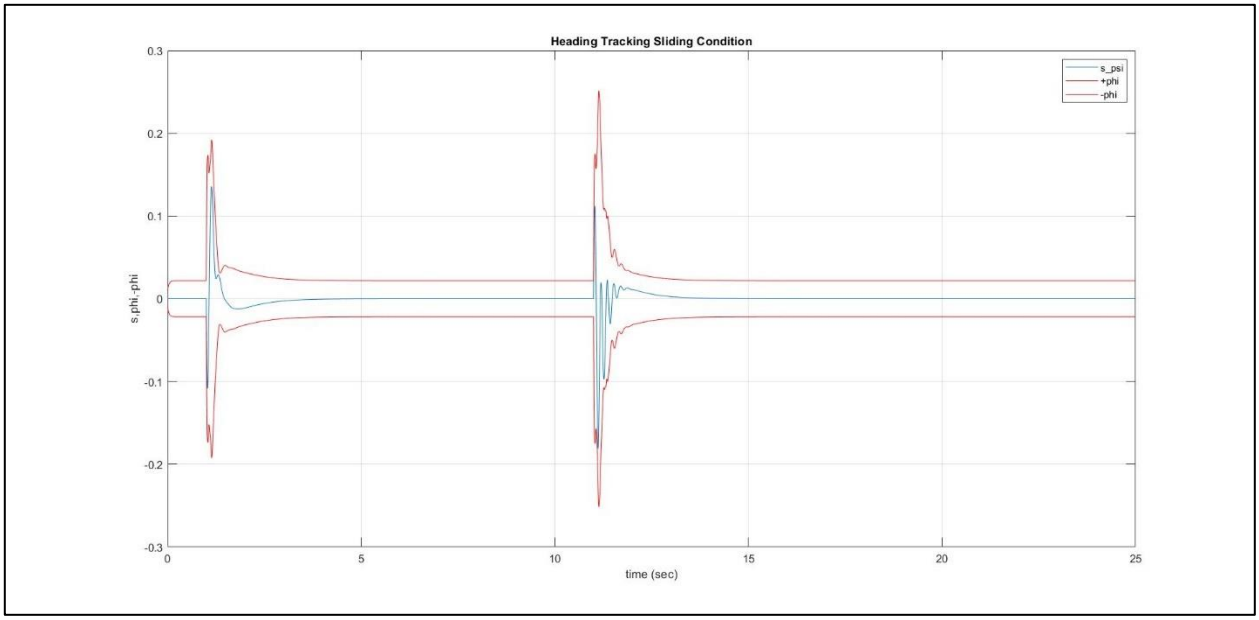


Figure 58: MFSMC, Double Moments; Yaw Sliding Condition

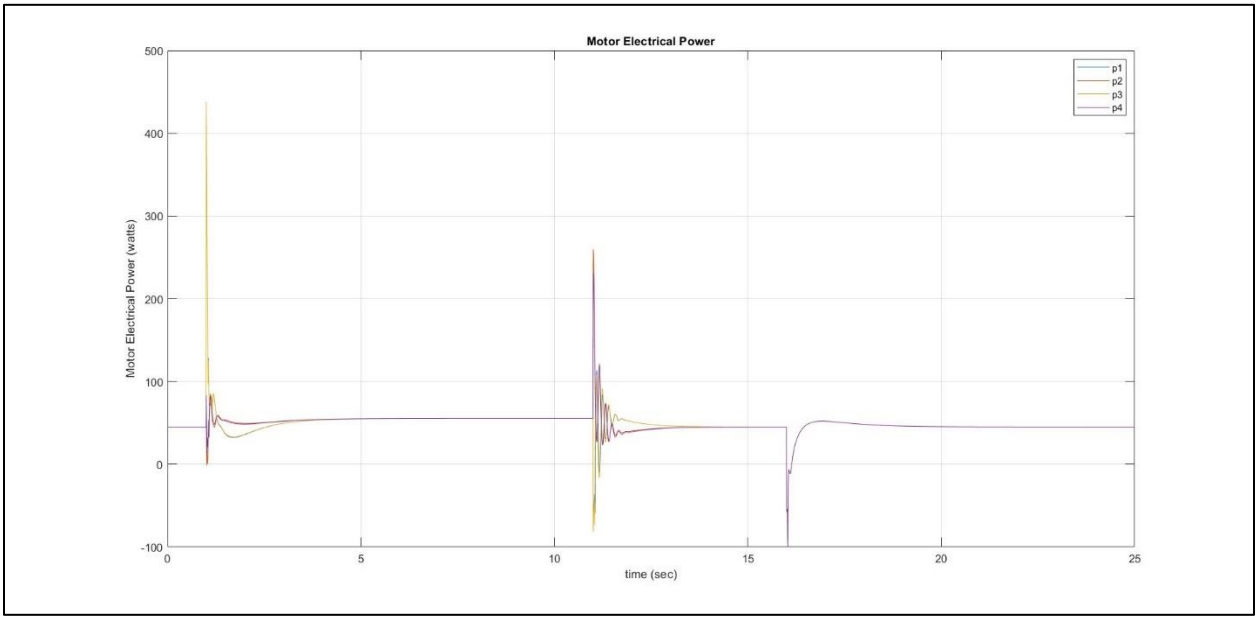


Figure 59: MFSMC, Double Moments; Electrical Power

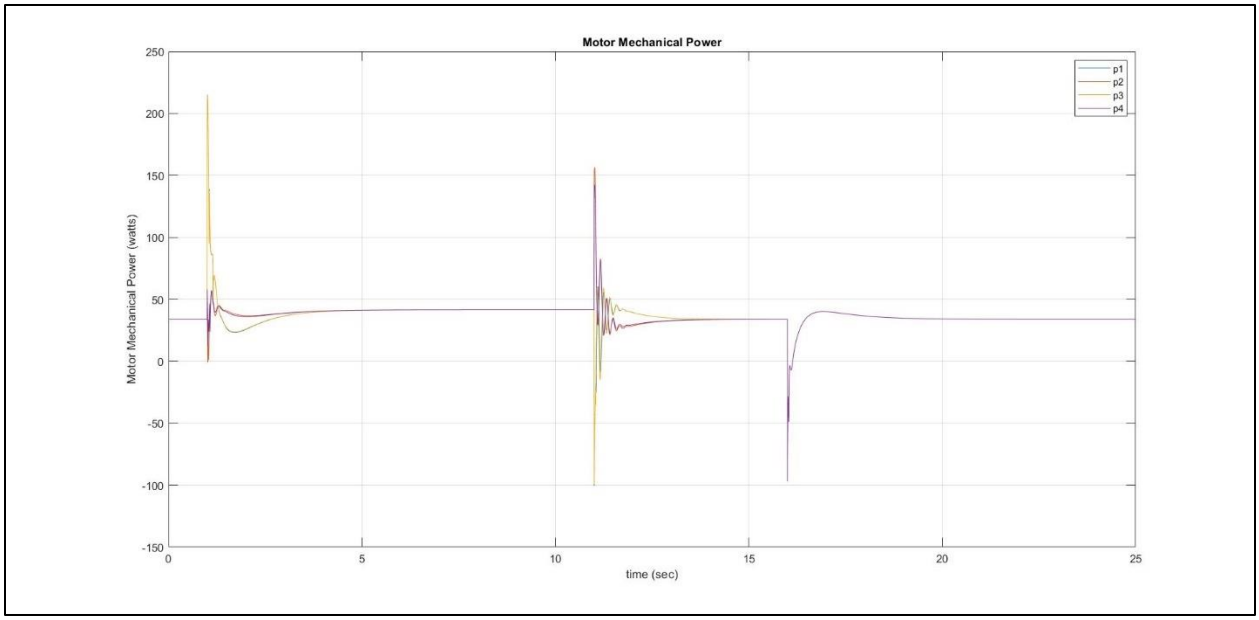


Figure 60: MFSMC, Double Moments; Mechanical Power

	PID	MFSMC
Altitude RMS (deg)	3.34188E+02	2.13310E-04
Roll RMS (deg)	3.26864E+03	2.56982E-02
Pitch RMS (deg)	2.82051E+01	1.45714E-02
Yaw RMS (deg)	8.02103E+03	2.08651E-02
Electrical Power Average (W)	1293.0	195.0
Mechanical Power Average (W)	966.6	146.4
Efficiency	74.76%	75.08%

Table 8: Results using Double Moments

It can be concluded from Table 8 that the MFSMC has better tracking for altitude, roll, pitch, and yaw, and it used less average power than that of the PID controller, using double the craft's original moments of inertia. The MFSMC sliding condition was satisfied for altitude, roll, pitch, and yaw control. It can also be noted from Figures 45 – 48 and Table 8 that the PID controller went unstable while the MFSMC algorithm performed similarly to the previous simulations.

5.3.4 Double the Craft's Mass and Moments of Inertia

In this section, the craft of both the mass and moments of inertia are doubled while using the exact same controllers in the previous simulations. The performance of the PID controller is presented first:

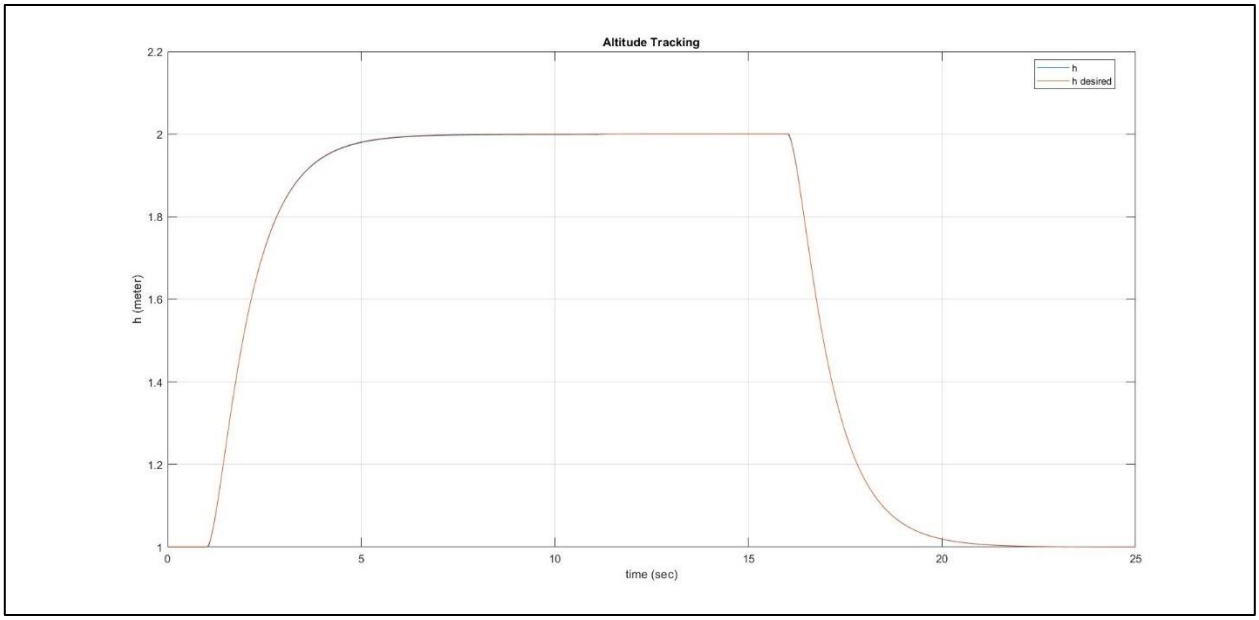


Figure 61: PID, Double Both; Altitude Tracking

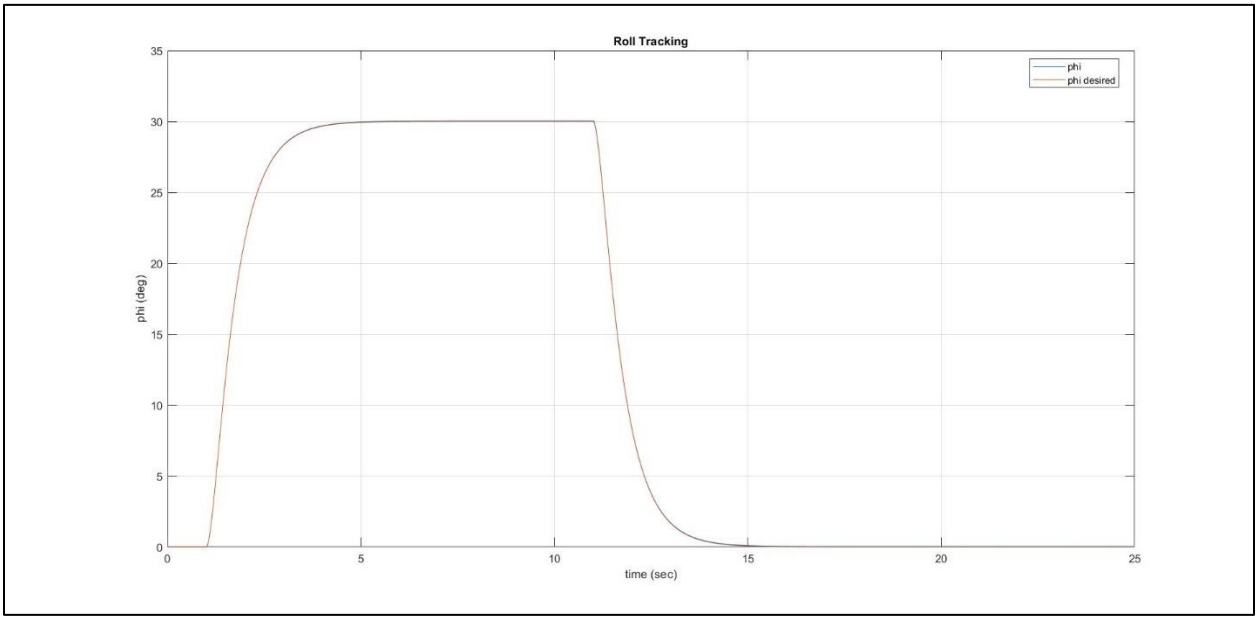


Figure 62: PID, Double Both; Roll Tracking

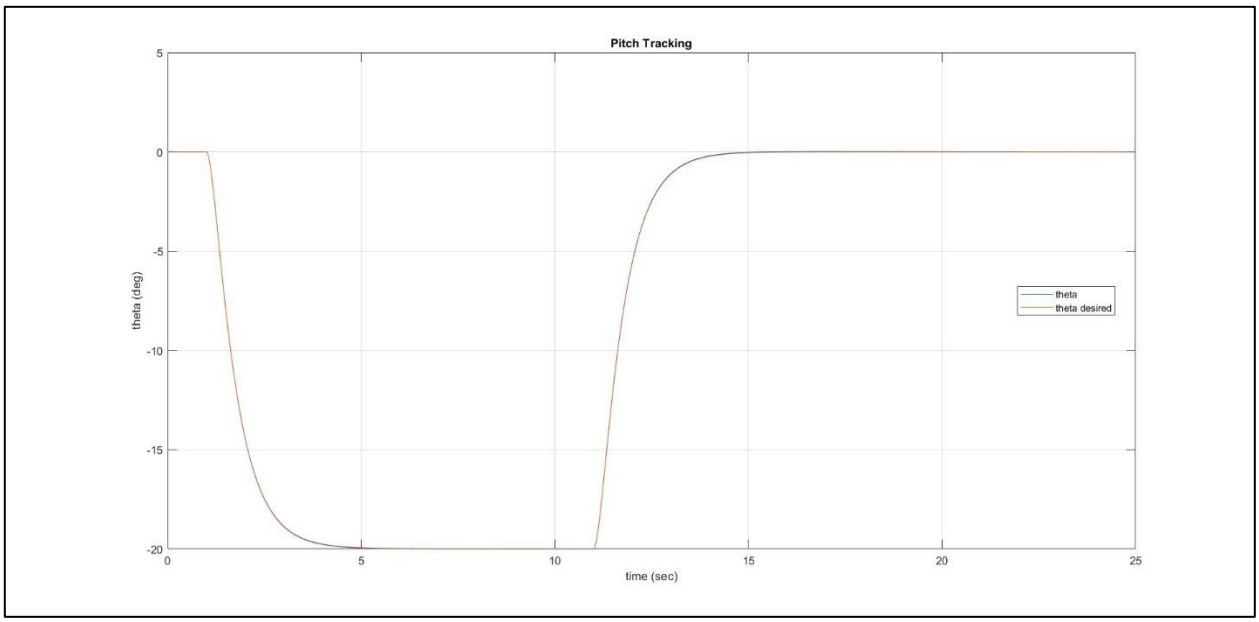


Figure 63: PID, Double Both; Pitch Tracking

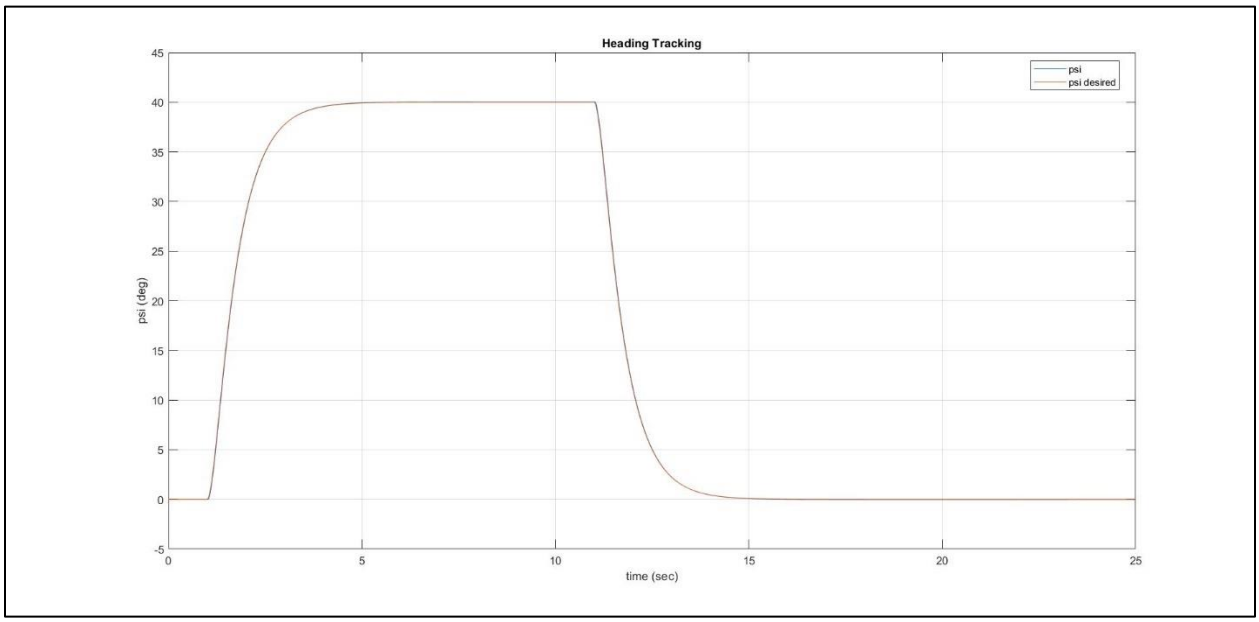


Figure 64: PID, Double Both; Yaw Tracking

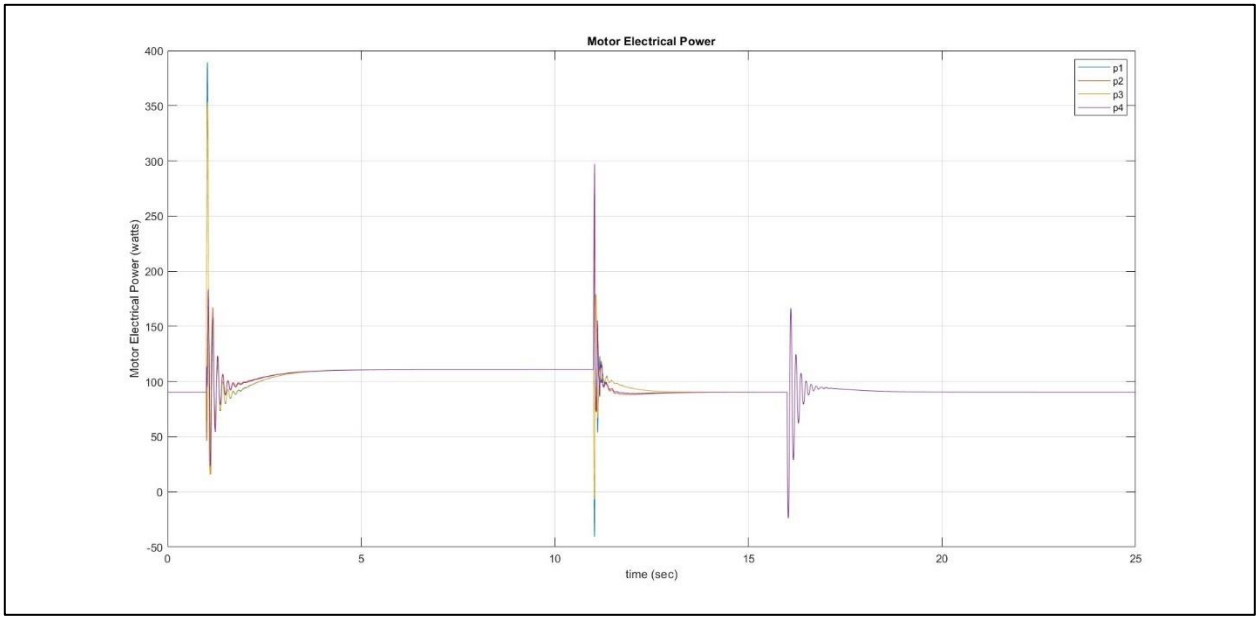


Figure 65: PID, Double Both; Electrical Power

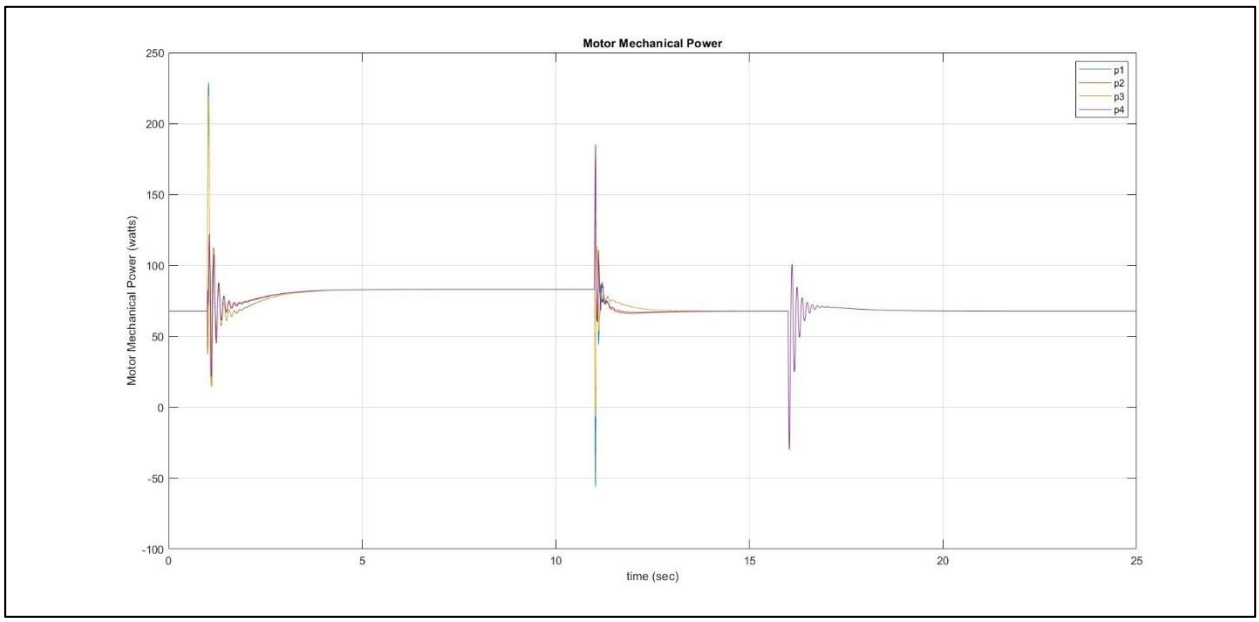


Figure 66: PID, Double Both; Mechanical Power

Now the performance of the MFSMC algorithm is presented:

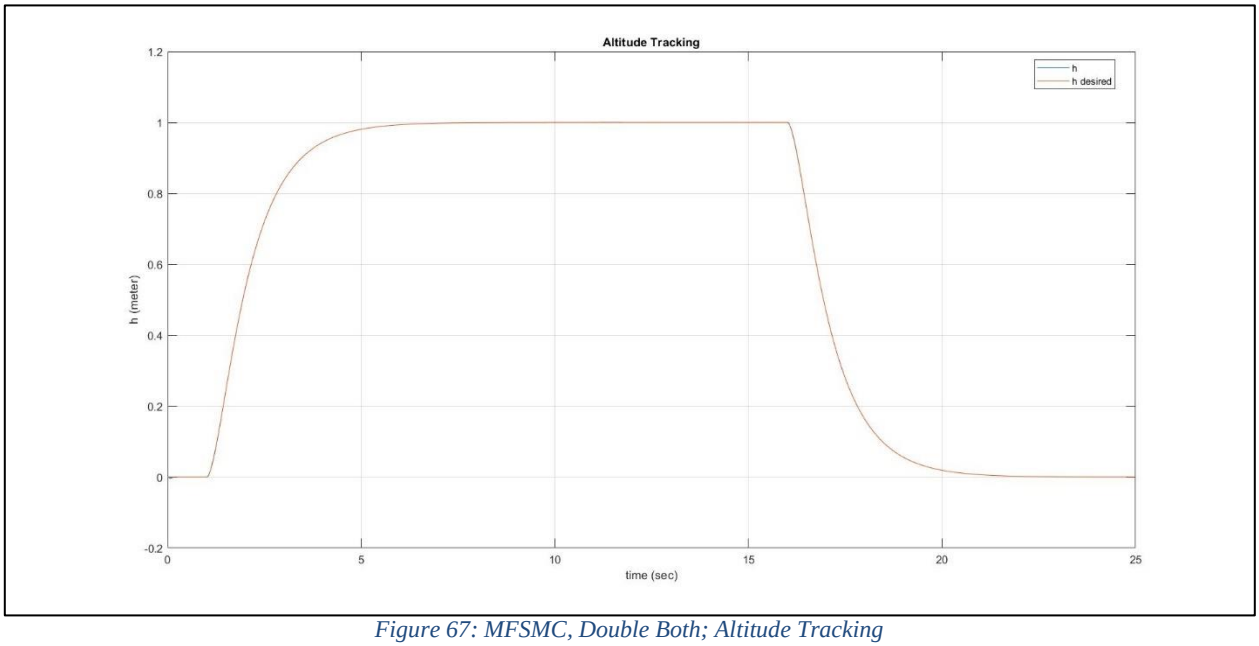


Figure 67: MFSMC, Double Both; Altitude Tracking

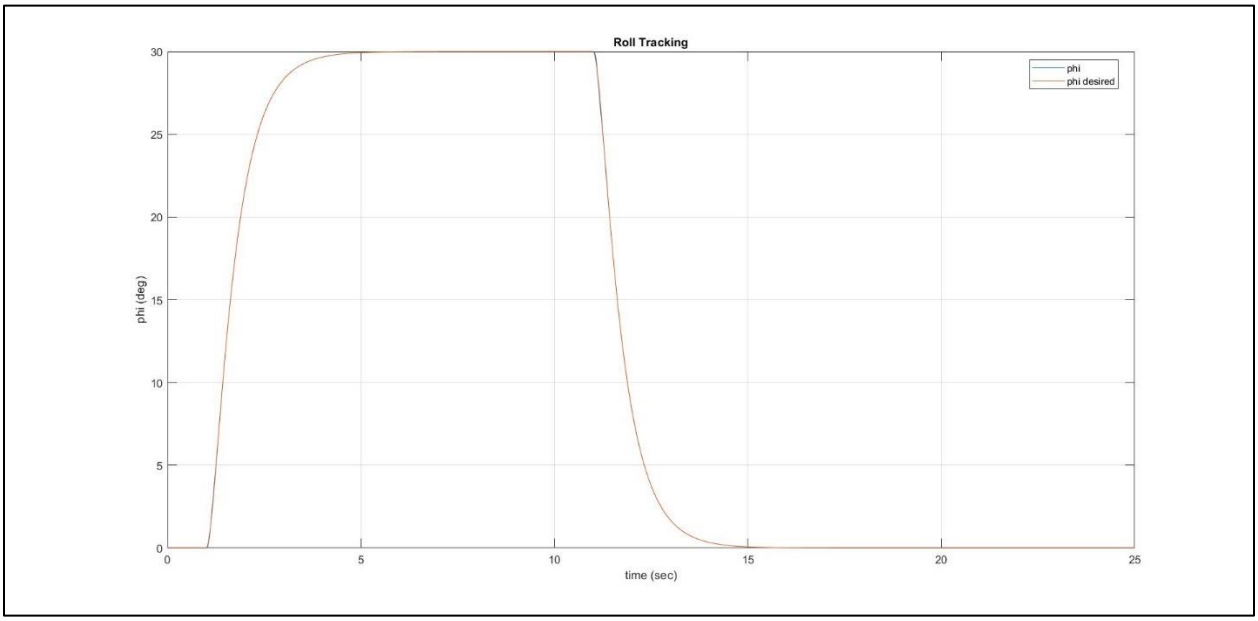


Figure 68: MFSMC, Double Both; Roll Tracking

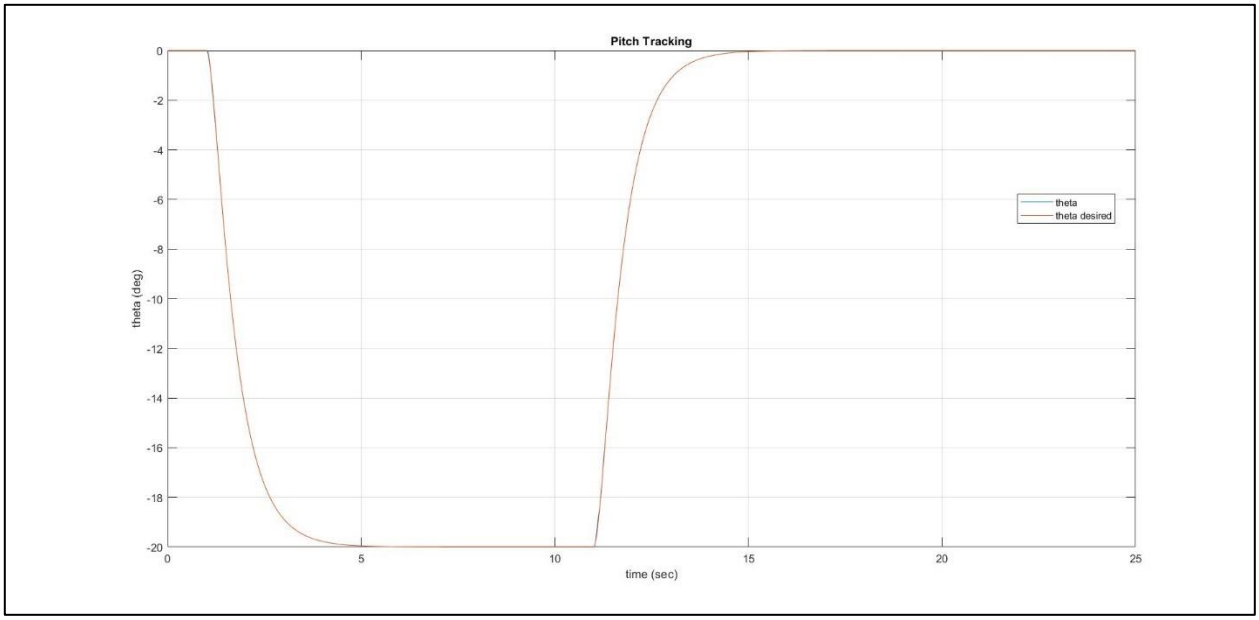


Figure 69: MFSMC, Double Both; Pitch Tracking

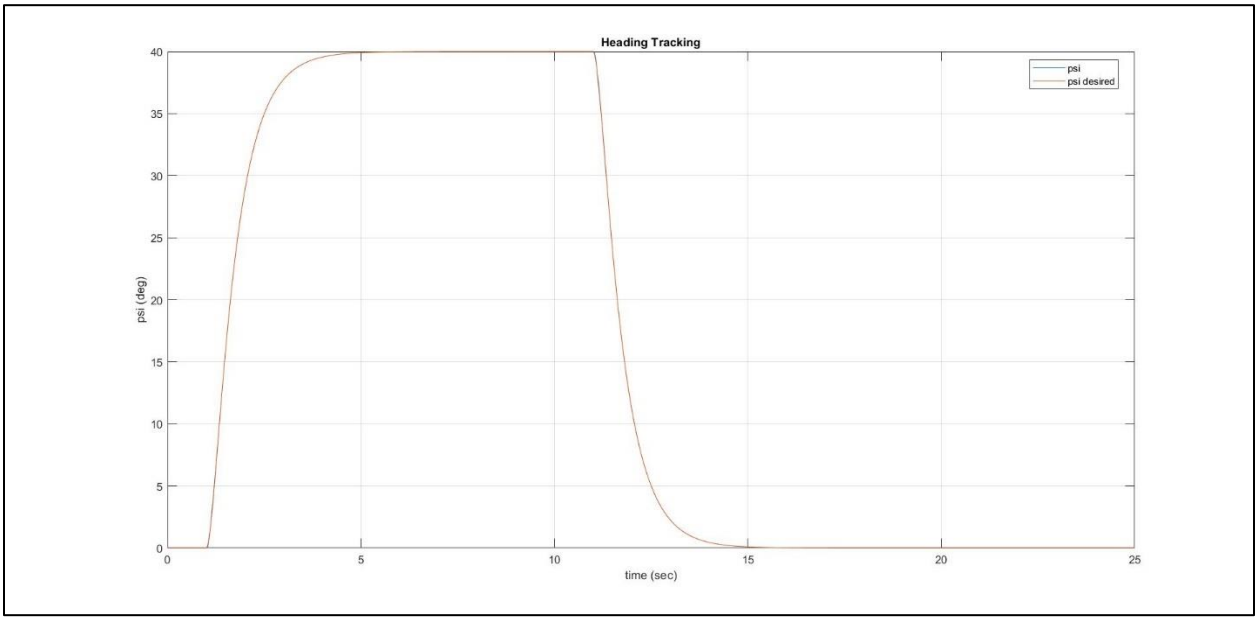


Figure 70: MFSMC, Double Both; Yaw Tracking

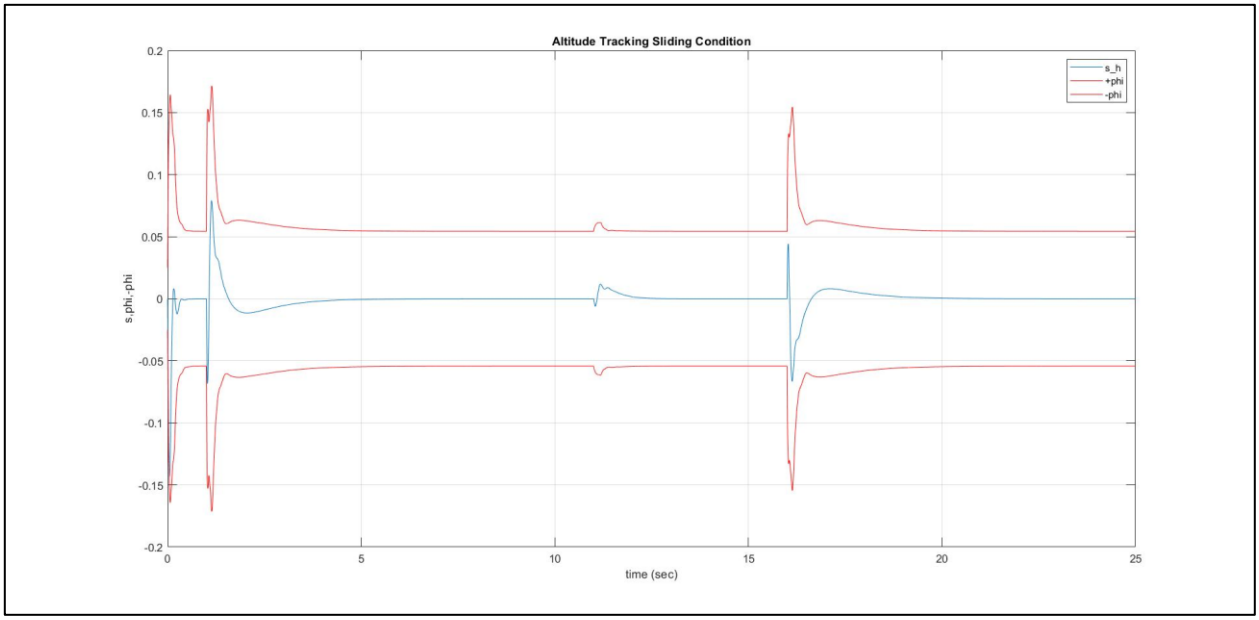


Figure 71: MFSMC, Double Both; Altitude Sliding Condition

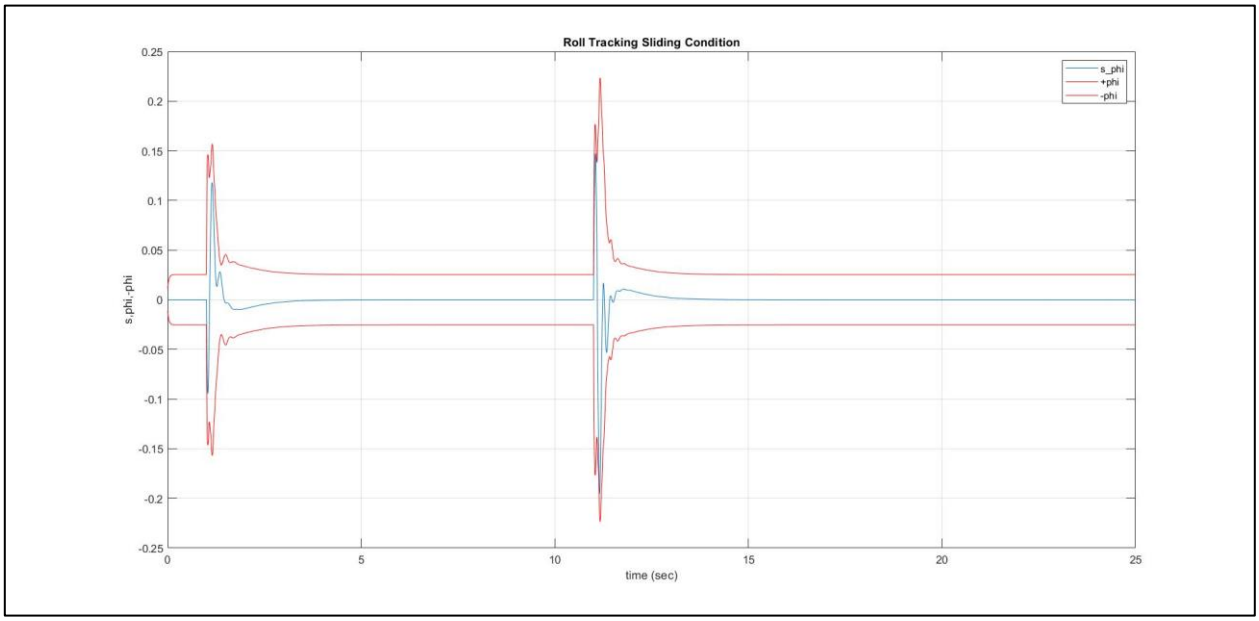


Figure 72: MFSMC, Double Both; Roll Sliding Condition

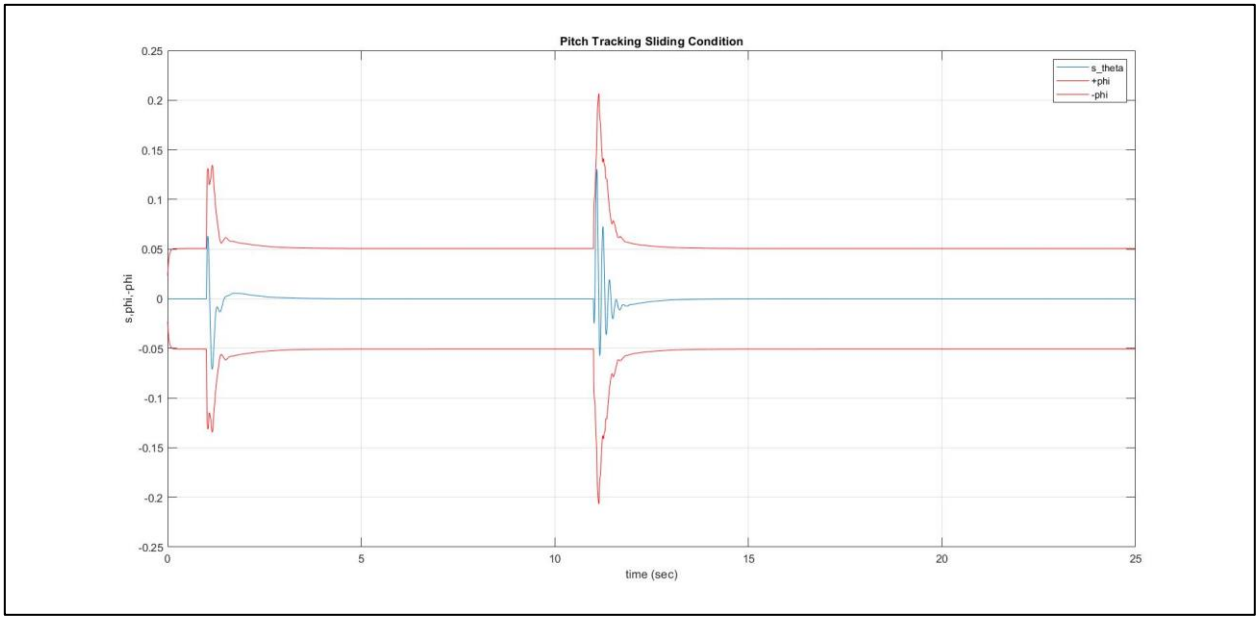


Figure 73: MFSMC, Double Both; Pitch Sliding Condition

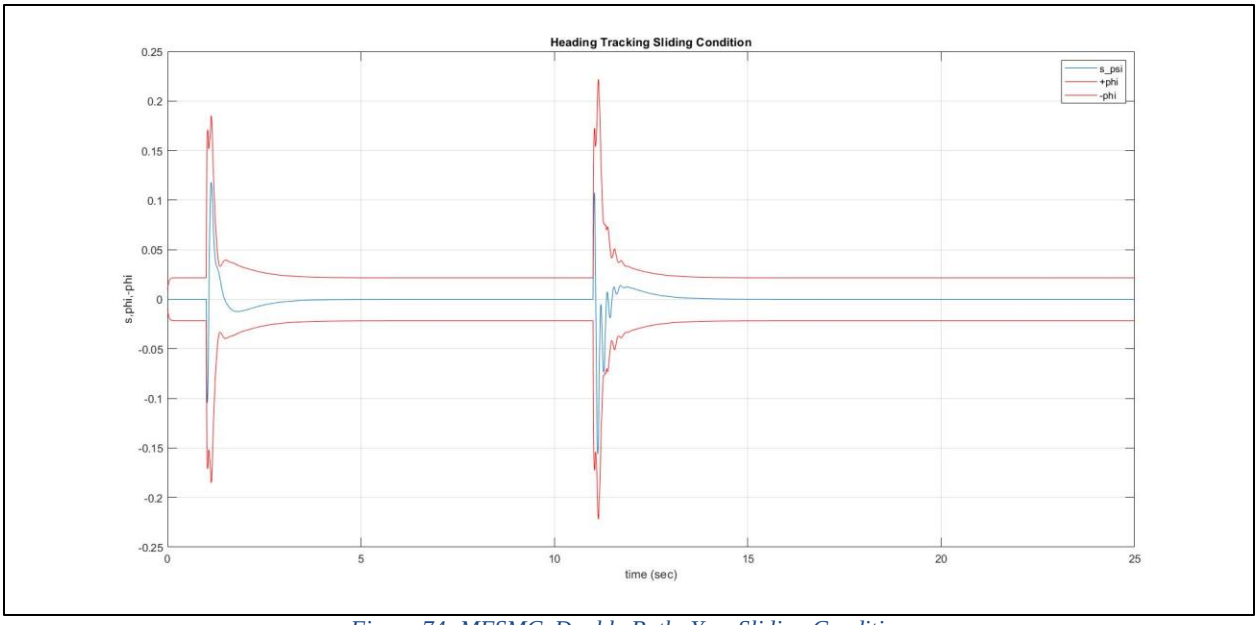


Figure 74: MFSMC, Double Both; Yaw Sliding Condition

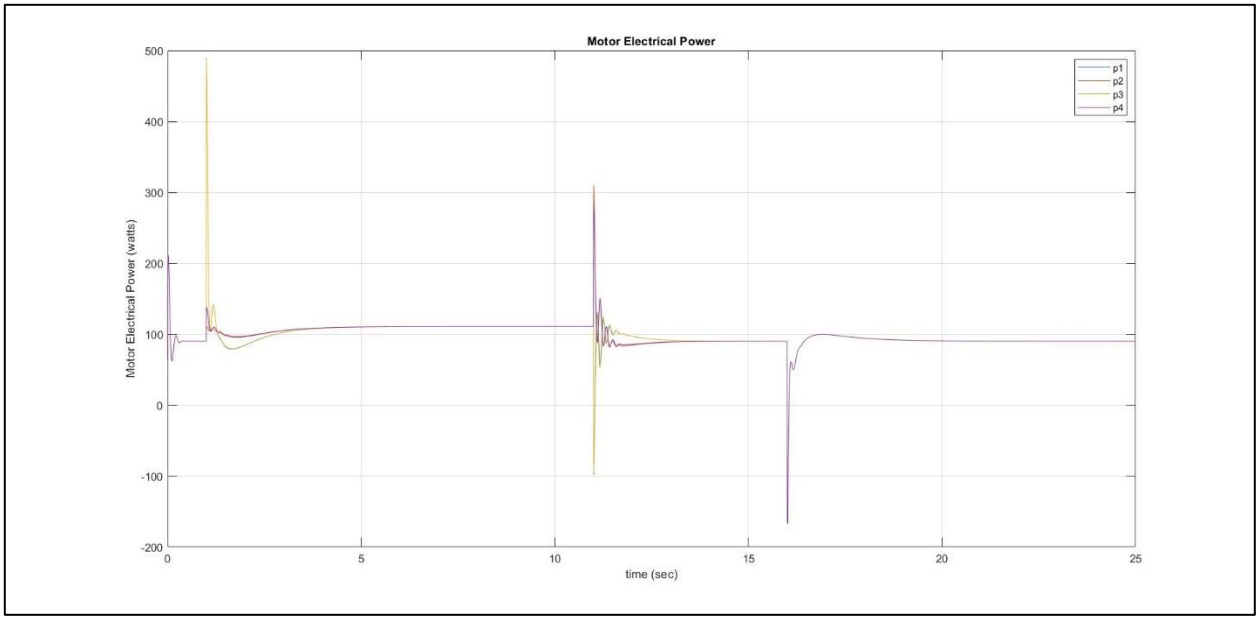


Figure 75: MFSMC, Double Both; Electrical Power

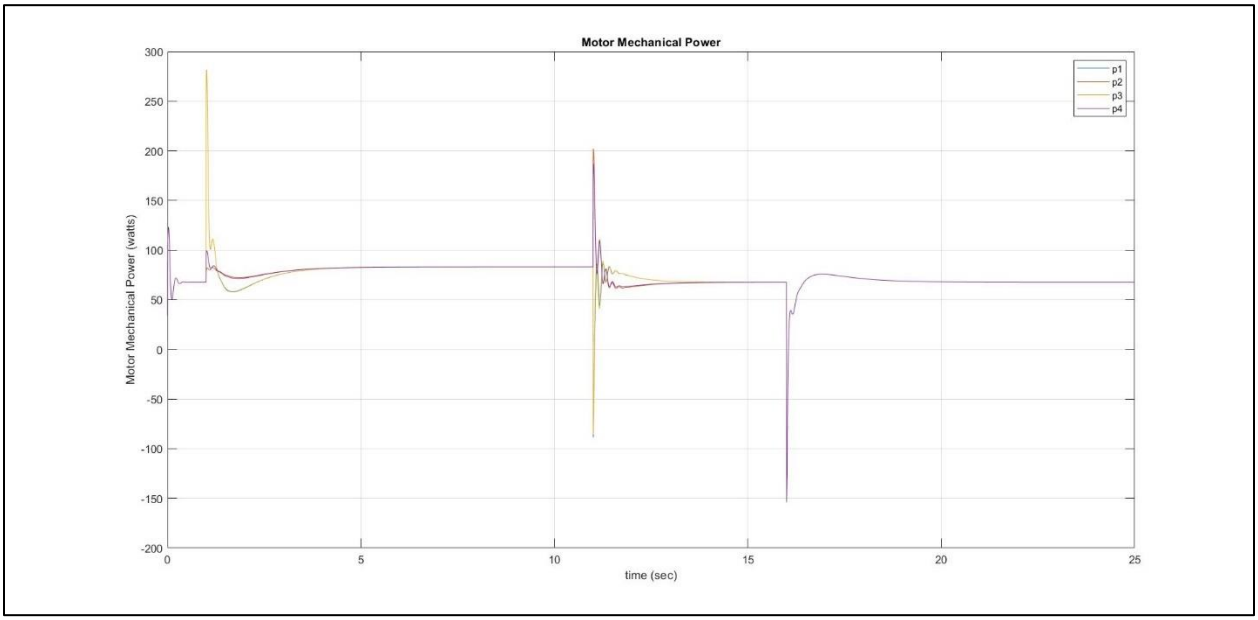


Figure 76: MFSMC, Double Both; Mechanical Power

	PID	MFSMC
Altitude RMS (deg)	8.30805E-04	3.14957E-04
Roll RMS (deg)	3.47131E-02	2.25154E-02
Pitch RMS (deg)	1.71014E-02	1.25008E-02
Yaw RMS (deg)	8.15854E-02	1.97198E-02
Electrical Power Average (W)	391.8	391.2
Mechanical Power Average (W)	293.2	293.0
Efficiency	74.83%	74.90%

Table 9: Results using Double Mass and Moments of Inertia

It can be concluded from Table 9 that the MFSMC has better tracking for altitude, roll, pitch, and yaw, and it used less average power than that of the PID controller, using double the craft's original mass and moments of inertia. The MFSMC sliding condition was satisfied for altitude, roll, pitch, and yaw control. It is suspected that the PID controller went unstable for doubling only the craft's moments of inertia and not for doubling both the mass and the moments of inertia because doubling the mass stabilized the system due to slower dynamics.

6.0 CONCLUSIONS

After deriving the $[B]$ matrix, the MFSMC algorithm and PID controllers are simulated and the performance is compared. In all four cases, the MFSMC algorithm outperformed the PID controller for altitude, roll, pitch, and yaw tracking since the RMS values for MFSMC are less than that of PID, as well as using less energy as shown in Tables 6 – 9. In the case where only the craft's moments of inertia were doubled the PID controller went unstable while the MFSMC algorithm performed adequately proving that the MFSMC algorithm can handle model uncertainties better than that of PID controllers. Based on Tables 6 – 9, it can be seen that the MFSMC algorithm not only uses less average power but also was, on average, more efficient in electrical power to mechanical power efficiency in all four cases for altitude, roll, pitch, and yaw control. It can be concluded that the MFSMC algorithm outperforms traditional PID controllers that are commonly used in applications for both tracking performance and power usage. Implementing this algorithm on a commercial aircraft would result in saving a significant portion fuel per flight while also slightly improving tracking performance. Also, due to different seating arrangements and different weight's of the passengers the MFSMC algorithm can better handle this uncertainty and further contribute to fuel savings and better tracking performance compared to that of a tradition PID controller.

7.0 FUTURE WORK

The MFSMC needs to be tested on a real-world quadcopter. Using accelerometers, gyroscopes, and a vision system the quadcopter's states can be determined. Given sample rate, noise, and uncertainty in the measurement the MFSMC algorithm needs to be monitored to assure that the performance is adequate, otherwise a study on sensors and senor fusion to maximize sample rate and/or minimize noise and uncertainty will be required.

The control algorithm needs to be updated to be able to handle X, Y desired positional coordinates rather than only altitude, roll, pitch, and yaw. This then needs to be implemented on a real-world quadcopter system and compare the performance of the MFSMC algorithm to an optimal PID controller.

The MFSMC algorithm requires researching optimal λ and η values such that deriving of the $[B]$ matrix isn't required, and the identity matrix can be assumed for the value of $[B]$. Successful completion in this will completely remove model derivation from the control algorithm and will reduce to strictly an optimization of λ and η.

8.0 REFERENCES

[1] R. Reis and A. Crassidis, "Model-free Sliding Mode Control Method," in *Proceedings of the 3rd International Conference of Control, Dynamic Systems, and Robotics*, Ottawa, Canada, no. 100, 2016.

[2] A. Crassidis and F. El Tin, " A Model-Free Control System Based on the Sliding Mode Control Method with Apllications to Multi-Input-Multi-Output Systems, " in *Proceedings of the 4th International Conference of Control, Dynamic Systems and Robotics*, Toronto, Canada, Paper No.119, 2017.

[3] A.R.K. Sreeraj, "A Model-Free Control Algorithm Based on the Sliding Mode Control Method with Applications to Unmanned Aircraft Systems", Thesis, Rochester Institute of Technology, 2019.

[4] S. Lagrouche, F. Plestan and A. Glumineau, "Higher Order Sliding Mode Control Based On Optimal Linear Quadratic Control", in *European Control Conference*, Cambridge, UK, pp. 910- 915, 2003.

[5] K. Runcharoon, and V. Srichatrapimuk, "Sliding Mode Control of Quadrotor," in *Technological Advances in Electrical, Electronics and Computer Engineering*, Konya, Turkey, 2013.

[6] L. Sen, L. Baokui and G. Qingbo, "Adaptive Sliding Mode Control for Quadrotor Helicopters", in *33rd Chinese Control Conference*, Nanjing, China, pp. 71-76, 2014.

[7] R. Xu and U. Ozguner, "Sliding Mode Control of a Class of Underactuated Systems," in *Science Direct: Automatica*, vol. 44, pp. 233-241, 2008.

[8] R. Olfati-Saber, "Normal Forms for Underactuated Mechanical Systems with Symmetry," in *IEEE Transactions on Automatic Control*, vol. 47, no. 2, pp. 305-308, 2002.

[9] H. Khalil, "Nonlinear Systems," Englewood Cliffs, NJ: Prentice-Hall, 3rd ed., ch. 14, pp. 575-579, 2002.

[10] M-C. Pai, "Robust Tracking and Model Following of Uncertain Dynamic Systems via Discrete-time Integral Sliding Mode Control," in *International Journal of Control, Automation, and Systems*, vol. 7, pp. 381-387, 2009.

[11] D. Milosavljevic, "General Conditions for the Existence of a Quasi-sliding Mode on the Switching Hyperplane in Discrete Variable Structure Systems," in *Automt. Remote Contr.*, vol. 46, pp. 307-314, 1985.

[12] S. Z. Sarpturk, Y. Istefanopulos, and O. Kaynak, "On the Stability of Discrete-time Sliding Mode Control Systems," in *IEEE Transactions on Automatic Control*, vol. 32, no. 10, pp. 930-937, 1987.

[13] S-H. Lee, "Sliding Mode Control Design Using Fast Output Sampling", in *Conference on Decision and Control*, IEEE Proceedings, Hawaii, USA, pp. 3543-3548, 2003.

[14] A. Ferrara and M. Rubagotti, "A Sub-Optimal Second-order Sliding Mode Controller for Systems with Saturating Actuators", in *IEEE Transactions on Automatic Control*, vol. 54, pp. 1082-1087, 2009.

[15] R. Martinez-Guerra, W. Yu, and E. Cisneros-Saldana, "A New Model-free Sliding Observer to Synchronization Problem," in *2006 3rd International Conference on Electrical and Electronics Engineering*, pp. 1-4, 2006.

[16] T. Salgado-Jimenez, L. G. Garcia-Valdovinos, and G. Delgado-Ramirez, "Depth Control of a 1 DOF Underwater System Using a Model-free High Order Sliding Model Control," in *Electronics, Robotics and Automotive Mechanics Conference*, pp. 481-487, 2010.

[17] R. Raygosa-Barahona, V. Parra-Vega, E. Olguin-Diaz, and L. Munoz-Ubando, "A Model-free Backstepping with Integral Sliding Mode Control for Underactuated ROVs," in *International Conference on Electrical Engineering, Computing Science and Automatic Control*, Merida City, Mexico, pp. 1-7, 2011.

[18] A.J. Munoz-Vazquez, V. Parra-Vega, and A. Sanchez, "A Passive Velocity Field Control for Naviagation of Quadrotors with Model-free Integral Sliding Mode Control," in *Journal of Intelligent & Robotic Systems*, vol. 73, pp. 373-385, 2014.

[19] A. Crassidis and A. Mizov, "A Model-Free Control Algorithm Derived Using the Sliding Mode Control Method" in *Proceedings of the 2nd International Conference of Control, Dynamic Systems, and Robotics*, Ottawa, ON, Paper No. 166, 2015.

[20] A. Mizov, "A Model-Free Control Algorithm Derived Using the Sliding Model Control Method", Thesis, Rochester Institute of Technology, Accessed from http://scholarworks.rit.edu/cgi/viewcontent.cgi?article=9826&context=theses., 2015.

[21] A. Levant, "Universal Single-Input-Single-Output (SISO) Sliding Mode Controller with Finite-Time Convergence," in *IEEE Transactions on Automatic Control*, Vol, 46, No. 9, pp.1447-1451, 2001.

[22] V. I. Utkin., "Discussion Aspects of High-Order Sliding Mode Control," in *IEEE Transactions on Automatic Control*, Vol. 61, No. 3, pp.829-833, 2016.

[23] R. E. Precup, M. B. Radac, R. C. Roman, and E. M. Petriu, "Model-Free Sliding Mode Control of Nonlinear Systems: Algorithms and Experiments, " in *Information Sciences*, Vol. 381, pp. 176-192, doi: 10.1016/j.ins.2016.11.026, 2017.

[24] M. A. Dehghani, M. B. Menhaj, "Integral sliding mode formation control of fixed-wing unmanned aircraft using seeker as a relative measurement system, "in *Aerospace Science and Technology*, Vol. 58, pp. 318-327. http://dx.doi.org/10.1016/j.ast.2016.08.011, 2016.

[25] D. Qian, J. Yi, and D. Zhan, "Multiple Layers Sliding Mode Control for a Class of Underactuated Systems," in *IMACS Multiconference on CESA*, Beijing, China, pp. 530-535, 2006.

[26] R. Schkoda, "Dynamic Inversion of Underactuated Systems via Squaring Transformation Matrix," *Thesis*, Rochester Institute of Technology, 2007.

[27] M. Norton, S. Khoo, A. Kouzani, and A. Stojcevski, "Adaptive Fuzzy Multi- Surface Sliding Control of Multiple-Input and Multiple-Output Autonomous Flight Systems," in *IET Control Theory and Applications*, Vol. 9, Iss. 4, pp. 587-597, doi:10.1049/iet-cta.2014.0209, 2014.

[28] E. Abdulhamitbilal, "Robust Flight Sliding Modes Control System Design for Nonlinear Aircraft with Parameter Uncertainties," in *13th IEEE Workshop on Variable Structure Systems*, Nantes, France, 2014.

[29] L. Duan, W. Lu, F. Mora-Camino, and T. Miquel, "Flight-Path Tracking Control of a Transportation Aircraft: Comparison of Two Nonlinear Design Approaches," in *25th Digital Avionics Systems Conference*, No. 4A5, pp. 1-9, 2006.

[30] A. Brezoescu, R. Lozano, and P. Castillo, "Lyapunov-Based Trajectory Tracking Controller for a Fixed-Wing Unmanned Aerial Vehicle in the Presence of Wind," in *International Journal of Adaptive Control and Signal Processing*, Vol. 29, pp. 372-384, doi:10.1002/acs.2480, 2014.

[31] A. Villanueva, B. Castillo-Toledo, E. Bayro-Corrochano, L.F. Luque-Vega, and L.E. Gonzalez-Jimenez, "Multi-Mode Flight Sliding Mode Control System for a Quadrotor," in *International Conference on Unmanned Aircraft Systems*, pp 861-870, 2015.

[32] L. Derafa, A. Benallengue, and L. Fridman, "Super Twisting Control Algorithm for the Attitude Tracking of a Four Rotors UAV," in *Journal of the Franklin Institute*, Vol. 349, pp. 685-699, doi:10.1016/j.jfranklin.2011.10.011, 2012.