Proceedings of the 6th International Conference of Control, Dynamic Systems, and Robotics (CDSR'19)
Ottawa, Canada – June 6-7, 2019
Paper No. CDSR 113
DOI: 10.1159/cdsr19.113

A Model Free Control Algorithm Based On The Sliding Mode Control Method With Applications to Unmanned Aircraft Systems

Adarsh Raj Kadungoth Sreeraj, Agamemnon Crassidis, Daniel Kaputa
Rochester Institute of Technology
1 Lomb Memorial Drive, Rochester, New York, 14623, USA
ak3092@rit.edu; alceme@rit.edu; dskiee@rit.edu

Abstract **–** Control methods require the use of a system model for the design and tuning of the controllers in meeting and/or exceeding the control system performance objectives. However, system models contain errors and uncertainties that also may be complex to develop and to generalize for a large class of systems such as those for unmanned aircraft systems. In particular, the sliding control method is a superior robust nonlinear control approach due to the direct handling of nonlinearities and uncertainties that can be used in tracking problems for unmanned aircraft system. However, the derivation of the sliding mode control law is tedious since a unique and distinct control law needs to be derived for every individual system and cannot be applied to general systems that may encompass all classifications of unmanned aircraft systems. In this paper, a model-free control algorithm based on the sliding mode control method is developed and generalized for all classes of unmanned aircraft systems used in robust tracking control applications. The model-free control algorithm is derived with knowledge of the system's order, state measurements, and control input gain matrix shape and bounds and is not dependent on a mathematical system model. The derived control law is tested using a high-fidelity simulation of a quadrotor-type unmanned aircraft system and the results are compared to a traditional PID controller for tracking performance and power consumption. Realistic type hardware inputs from joysticks and IMUs was simulated for the analysis. Finally, the model-free control algorithm was implemented on a quadrotor-type unmanned aircraft system testbed used in real flight experimental testing. The experimental tracking performance and power consumption was analysed and compared to a traditional PID-type controller.

Keywords**:** Model-Free Control, Sliding Mode Control, Robust Control, Flight Control, Unmanned Aircraft Systems.

1. Introduction

There has been a tremendous rise in the field of control systems and with an increase in system complexity, control systems need to be more stable and versatile. Nonlinear systems are more complex and exhibit unpredictable behaviour requiring complex control algorithms. Due to its robustness the Sliding Mode Control (SMC) scheme is a popular approach used for both linear and nonlinear systems with modelling uncertainties [1]. However, SMC typically requires knowledge of the mathematical model of the system to be controlled. Additionally, the control law derivation is tedious and time consuming and needs to be derived for every system [2], [3]. Therefore, a significant advantage is realized if a model-free control algorithm that is both robust and system independent.

Mizov and Crassidis in [4] developed a model-free control algorithm using SMC and successfully implemented it to control a linear and nonlinear spring mass damper system. The control algorithm was independent of the system model and was only dependent on the system states, order of the system and previous control inputs. Reis and Crassidis extended this work to Single-Input-Single-Output (SISO) systems with unitary and non-unitary gains [5] assuming the bounds of control input gain matrix was known. Reis also suggested the tuning method of the control algorithm in the presence of noise and successfully implemented the algorithm on nonlinear and linear SISO systems. El Tin [6] further extended this work on Multi-Input-Multi-Output (MIMO) systems. He successfully used the control algorithm for quadrotor positon control and investigated the issue of actuator dynamics. Schulken [7] investigated an alternative form of Model-Free Sliding Mode Control (MFSMC) and its application on an real-world quadcopter. The simulation results shown in [7] were acceptable but hardware testing results were lacking leading to improper tracking and vibrations in the model.

The work developed in this paper is to successfully implement the MFSMC derived in [4], [5], [6], and [7] on a real-time quadcopter type Unmanned Aircraft System (UAS) and compare the tracking performance and power consumption to a traditional PID-type controller. Simulations characterising the quadcopter hardware is performed prior to targeting the real-time model.

2. Model-Free Sliding Mode Controller

A similar approach performed in [4], [5], [6], and [7] is used to derive the MFSMC for a quadcopter. The quadcopter is a 6 Degree-of-Freedom (DOF) 2nd-order system. The MFSMC for a 2nd-order system will be presented in this section.

2.1. Sliding Surface

Slotine [1] defines a time varying sliding surface as:

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

where λ is a positive constant, n is the order of the system and $\tilde{x}_p = x_p - x_{d_p}$ defines the tracking error. The sliding surface for a 2nd-order MIMO system where $n = 2$ is defined as:

$$\vec{s} = \dot{\tilde{\vec{x}}} + \lambda\tilde{\vec{x}} \tag{2}$$

Lyapunov's direct method is used to ensure asymptotic stability of the system's states during the reaching phase while the control law drives the states towards the sliding surface. The energy of the system is given by Lyapunov's function:

$$\vec{V}(\vec{x}) = \frac{1}{2}\vec{s}^2 \tag{3}$$

The rate of energy of given by:

$$\dot{\vec{V}}(\vec{x}) = \dot{\vec{s}}\vec{s} = -\eta|\vec{s}| \leq 0 \tag{4}$$

where η is a small positive constant. Since Eq. (4) is negative definite the closed-loop system is asymptotically stable.

2.2. System Description

A general nth-order autonomous system is defined as:

$$x_p^n = f_p(x) + [B]_{p\times m}u_m \tag{5}$$

and can be rewritten as:

$$x_p^n = x_p^n + [B]u_m - [B]u_{m_{k-1}} - [B]u_m + [B]u_{m_{k-1}} \tag{6}$$

where the subscripts "p" and "m" are the number of inputs and outputs respectively, x is the system states, [B] is the square m x m matric of control input gains and $u_{m_{k-1}}$ is the previous control input. The error between the control input and the previous control input is defined as:

$$\varepsilon_m = u_{m_{k-1}} - u_m \tag{7}$$

A control input error estimation is required to avoid an algebraic loop within the controller and to compute the control law. Eq. (7) is redefined as:

$$\hat{\varepsilon}_m = u_{m_{k-1}} - u_{m_{k-2}} \tag{8}$$

where $u_{m_{k-2}}$is the previous control input of the previous control input. The control input error is not exactly known but is assumed to be within the following bounds:

$$(1 - \sigma_l)\hat{\varepsilon}_m \leq \hat{\varepsilon}_m \leq (1 + \sigma_l)\hat{\varepsilon}_m \tag{9}$$

where σ_u is the upper bounds and σ_l is the lower bound of the control input estimation error.

2.3. Model-Free Control Algorithm

Using Eqs. (2), (4), (6) and (8), the control is defined and updated according to the following:

$$\vec{u} = -[\hat{B}]^{-1}\left[\ddot{\vec{x}} - \ddot{\vec{x}}_d + \lambda\dot{\tilde{\vec{x}}} + \vec{K}sgn(\vec{s})\right] + 2\vec{u}_{k-1} - \vec{u}_{k-2} \tag{10}$$

where, $\vec{K}$ is the switching gain that ensures asymptotic stability of the system trajectories during the reaching phase and $[\hat{B}]$ is the estimate of the control input gain matrix. The switching gain is defined as follows:

$$\vec{K} = \left|\ddot{\vec{x}} - \ddot{\vec{x}}_d\right||[\beta] - 1| + \lambda\left|\dot{\tilde{\vec{x}}}\right||[\beta] - 1| + \left|\left[\hat{B}\right]\sigma_u(\vec{u}_{k-2} - \vec{u}_{k-1})\right| + [\beta]\eta \tag{11}$$

where the upper bounds of the error estimate is used to be conservative and an auxiliary variable β is used. The estimated control input gain and the auxiliary variable is defined as follows:

$$\left[\hat{\vec{B}}\right] = \sqrt{b_{upper}b_{lower}} \tag{12}$$

$$\beta = \hat{b}\hat{b}^{-1} = \sqrt{\frac{b_{upper}}{b_{lower}}} \tag{13}$$

The discontinuity in the control law causes high frequency chattering of the controller resulting in damage to the actuators and motors. Hence, a smoothing boundary layer is added into the control law using the following dynamics:

$$\vec{u} = [B]^{-1}\left[-\ddot{\vec{x}} + \ddot{\vec{x}}_d - \lambda\dot{\tilde{\vec{x}}} - (\vec{K} - \dot{\vec{\Phi}})sat\left(\frac{\vec{s}}{\vec{\Phi}}\right)\right] + 2\vec{u}_{k-1} - \vec{u}_{k-2} \tag{14}$$

$$\dot{\vec{\Phi}} + \lambda\vec{\Phi} = \vec{K}(\vec{x}_d) \tag{15}$$

where $\vec{\Phi}(0) = \eta/\lambda$ resulting in, essentially, applying a 1st-order filter dynamic to control effort.

3. Control Approach for a Quadcopter

A quadcopter is a 6 DOF system traveling linearly in the *X*, *Y* and *Z* direction and rotating about *X* (Roll, Φ), *Y* (Pitch, Θ) and *Z* (Yaw, ψ). The dynamic model of the quadcopter used in this work is the same used in [6-9]:

$$\ddot{X} = \frac{(\sin\psi\ \sin\Phi - \cos\psi\sin\theta\cos\Phi)U1 - A_x\dot{X}}{m} \tag{16}$$

$$\ddot{Y} = \frac{(\cos\psi\ \sin\Phi - \sin\psi\sin\theta\cos\Phi)U1 - A_y\dot{Y}}{m} \tag{17}$$

$$\ddot{X} = \frac{(\sin\psi\ \sin\Phi - \cos\psi\sin\theta\cos\Phi)U1 - A_x\dot{X}}{m} \tag{18}$$

$$\ddot{\Phi} = \frac{\dot{\theta}\psi(I_{yy} - I_{zz}) + J_r\dot{\theta}\Omega_r + lU2}{I_{xx}} \tag{19}$$

$$\ddot{\theta} = \frac{\dot{\Phi}\dot{\psi}(I_{zz} - I_{xx}) + J_r\dot{\Phi}\Omega_r + lU3}{I_{yy}} \tag{20}$$

$$\ddot{\psi} = \frac{\dot{\theta}\dot{\Phi}\left(I_{xx} - I_{yy}\right) + U4}{I_{zz}} \tag{21}$$

where A_x, A_y, A_z are the air resistance in the respective axes (kg/s); Ω_r is total velocity of all the propellers; I_{xx}, I_{yy} and I_{zz} are the moment of inertias (Kg-m^2); J_r is the rotor inertia of the motors (Kg-m^2); l is the distance between the rotor axis and quadcopter center-of-gravity; m is the mass of the quadcopter (kg); *U1* is the general thrust and *U2*, *U3* and *U4* are roll, pitch, and yaw thrust, respectively. These six 2nd-order-nonlinear dynamical equations are referenced to the earth-fixed coordinate frame. The four control inputs (*U1*-*U4*) are defined in the body-reference frame of the quadcopter.

The quadcopter model is an underactuated system since there are only four control inputs for six system states. El Tin used a transformation matrix to handle the underactuation of the quadcopter for simulation study [6]. Another approach to handle underactuation is to use the super twisting method suggested by Derafa [9] as was used in [2]. The method introduces two pseudo-control inputs U_x and U_y to define the required roll and pitch angles. These control approaches are required if the states to be controlled are Z (altitude), X, Y, and yaw (ψ). Since in this work the states to be controlled are Z (altitude), Φ (roll angle), Θ (pitch angle) and ψ (yaw angle), the states can directly controlled by the four control inputs (*U1* - *U4*). The control law in Eq. (14) was updated to handle actuator dynamics as suggested by El Tin [6]:

$$\vec{u} = [B]^{-1}\left[-\ddot{\vec{x}} + \ddot{\vec{x}}_d - \lambda\dot{\tilde{\vec{x}}} - (\vec{K} - \dot{\vec{\Phi}})sat\left(\frac{\vec{s}}{\vec{\Phi}}\right)\right] + 2\vec{u}_{pa_{k-1}} - \vec{u}_{pa_{k-2}} \tag{22}$$

where u_{pa} is the control input post actuation. Two simulation studies 1: Full Flight Simulation and 2: Gimbal Flight Simulation were conducted to study the controller behaviour before real time gimbal flight tests were conducted with the MFSMC algorithm.

4. Quadcopter Description

The Fusion 1 drone from Craft Drones is used for hardware testing. It uses the Snickerdoodle SOM which uses the Zynq 7020 System-on-Chip (SoC) board. It comprises a Dual–core ARM Cortex – AP MPCore™ with Coresight™ with a clock speed of 667 MHz. It also houses an Artix-7 FPGA with 85k programmable logic cells. The quadcopter has 4 EMAX 15 AMP Electronic Speed Controllers (ESCs) with a burst current of 25A. 4 EMAX 4500 KV 1106 brushless motors with a max speed of 36000 RPM is used. The MPU9250 IMU sensor is used for state estimation. It is comprised of a 3-axis gyroscope, a 3-axis accelerometer and an onboard Digital Motion Processor™ capable of processing complex MotionFusion algorithms. An ADS1015 Analog to digital converter is also housed in the quadcopter which provides a 12-bit precision bus voltage at 1000 samples/second. The Turnigy 1000 mAh 2S battery is used to power the quadcopter. Finally the Fusion 1 drone has a JTAG debugging header.

The control inputs are converted into the motor inputs using the motor mixing functions for an 'X' configuration quadcopter. Four MFSMC controllers are used (one for each independently controllable DOF) and the performance is compared to four PID controller required for full flight simulation. For the gimbal flight simulation, only roll, pitch and yaw are controlled so that general thrust (*U1*) is explicitly running at a rate of 1 KHz. Finally, the gimbal flight test is conducted on the Fusion 1 drone.

4.1. State Estimation

Unlike the PID controller, the MFSMC algorithm requires full state feedback. The IMU includes a 3 axis-accelerometer and 3-axis gyroscope. These sensors directly measure the $\ddot{X}, \ddot{Y}, \ddot{Z}, \dot{\Phi}, \dot{\theta}$, and $\dot{\psi}$ states. The $\dot{Z}, Z$ states are estimated by discrete integration of the accelerometer signal $\ddot{Z}$. The roll (Φ) and pitch (Θ) angle are calculated by using a complimentary filter scheme [10], [11] and the three angular accelerations are estimated using sliding mode differentiators. The sliding mode

differentiators eliminate phase lag and noise unlike traditional differentiators [12]. Finally the yaw angle is directly calculated by integrating the gyroscope signal.

4.2. Noise Estimation

All accelerometer and gyroscope sensor have an inherent noise on their outputs. This noise needs to be added into the simulation model to study the exact behaviour of the PID and MFSMC controller. The raw data (shown in Figure 1) from the IMU was collected from the Fusion 1 drone at 1 kHz sampling rate and a noise estimation study is conducted to evaluate the noise parameters. Mean, variance (σ) and the peak to peak (V_{pp}) value of the noise was determined as shown in Table 1.

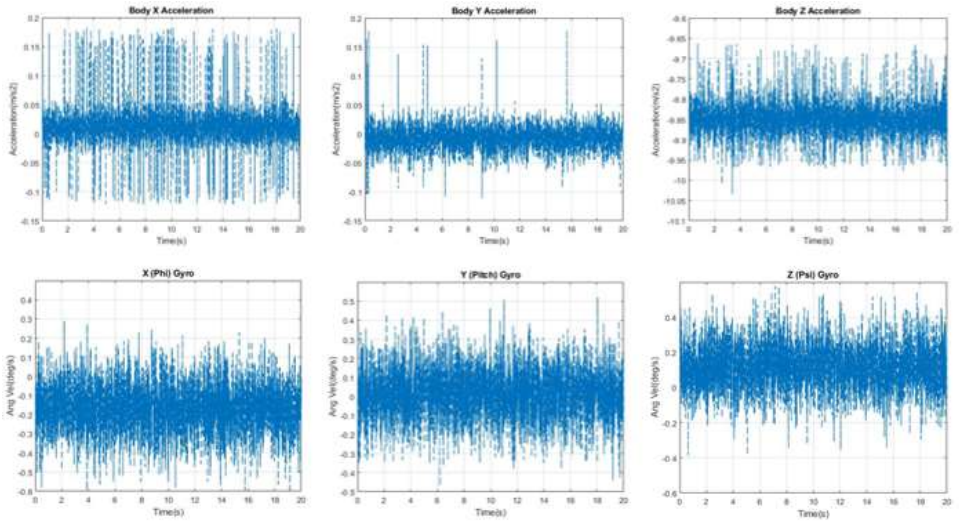

Fig. 1: Raw Noise Data for 3-axis Accelerometer and Gyroscope Sensors.

Table 1: Noise Parameter Estimation Characteristics for the 3-axis Accelerometer and Gyroscope Sensors.

Parameters	X_Accel ($m/s^{2)}$	Y_Accel (m/s^2)	Z_Accel (m/s^2)	X_Gyro (rad)	Y_Gyro (rad)	Z_Gyro (rad)
Mean	0.02	~0	-9.81	-0.004	~0	0.0017
V_{pp}	0.35	0.3	0.4	0.0157	0.0139	0.017
σ	0.0583	0.05	0.06	0.0026	0.0023	0.0029

5. Simulation Studies

Various hardware performance characteristics including sensor delays, actuator dynamics and noise were used in the simulations. The actuator dynamics were modelled as a 1^{st}-order dynamic with a time constant of 0.001 s. Full flight and gimballed flight simulation were conducted with both the PID and MFSMC running at 1 kHz. The MFSMC parameters λ, η and σ_u were set to 1, 0.1587 and 0.5 respectively and a constant thrust of 0.4 N was used for the gimbal flight simulation for both controllers.

Figures 2 and 3 present the tracking performance of the PID and MFSMC controller for both gimbal flight and full flight simulation results. The results indicate good and similar tracking precision for both the controller types. However, the RMS values of the tracking error shown in Table 2 offer insights between the performances of both the controllers. Both the controllers exhibit similar tracking performance during the gimbal flight simulation. In the full flight simulation the tracking precision for both roll and pitch angles were superior for the MFSMC law as compared to the PID controller. The tracking

precision of MFSMC was higher compared to the PID control law by a factor of 10 as observed from the RMS values shown in Table 2.

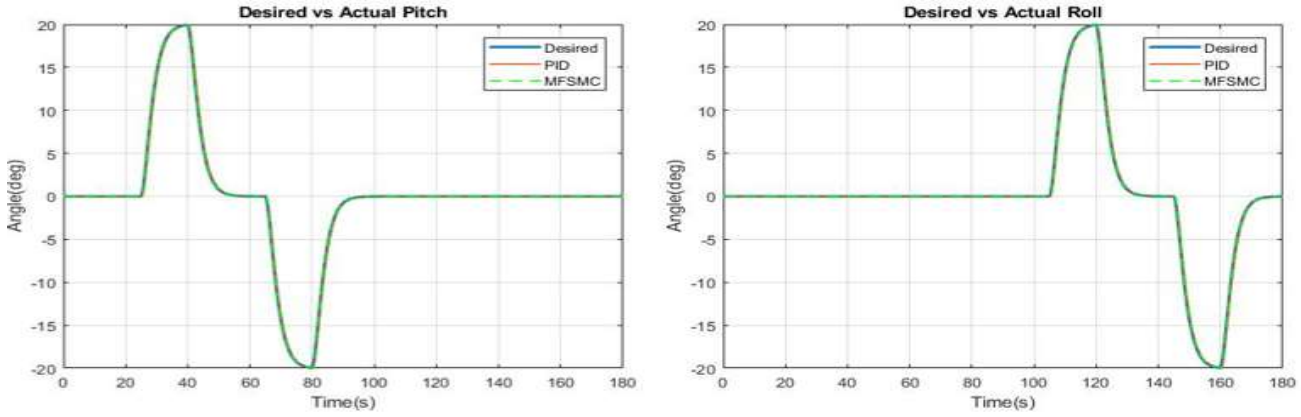

Fig. 2: Gimbal Flight Simulation of Roll and Pitch Angle Tracking for the PID and MFSMC Controllers.

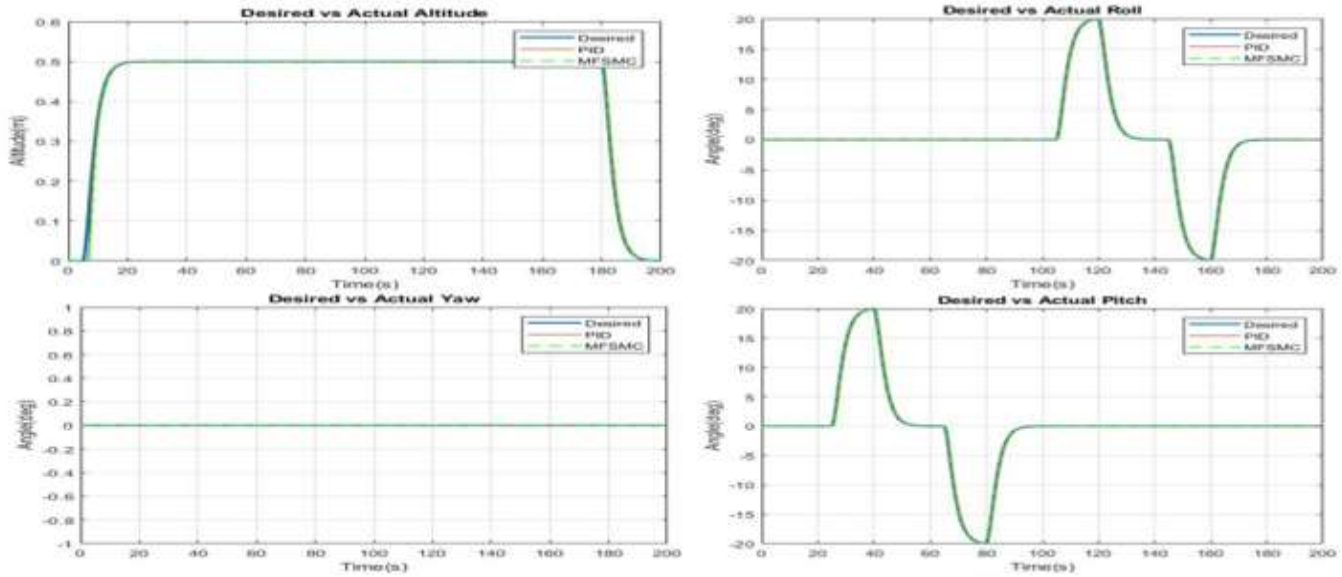

Fig. 3: Full Flight Simulation Responses for Roll and Pitch Angle Tracking for PID and MFSMC Controllers.

Table 2: RMS Tracking Errors Values for the PID and MFSMC Controllers for Simulation Study.

Controller	**RMS**					
	Gimbal Sim		**Full Flight Sim**			
	Roll (deg)	Theta (deg)	Altitude (Z)(m)	Roll(deg)	Pitch(deg)	Yaw(deg)
PID	0.24	0.05	0.0097	0.23	0.23	0.00
MFSMC	0.24	0.05	0.0114	0.03	0.03	0.00

6. Hardware Flight Testing

The Fusion 1 quadcopter from Craft Drones (http://www.craftdrones.com/) was mounted on gimbal for flight testing. Both the PID and MFSMC algorithm was used for roll and pitch control. The tracking performance and power

consumption for the controllers were studied. Figure 4 and Table 3 confirms that the MFSMC algorithm exhibits higher tracking precision compared to the optimally tuned PID controller.

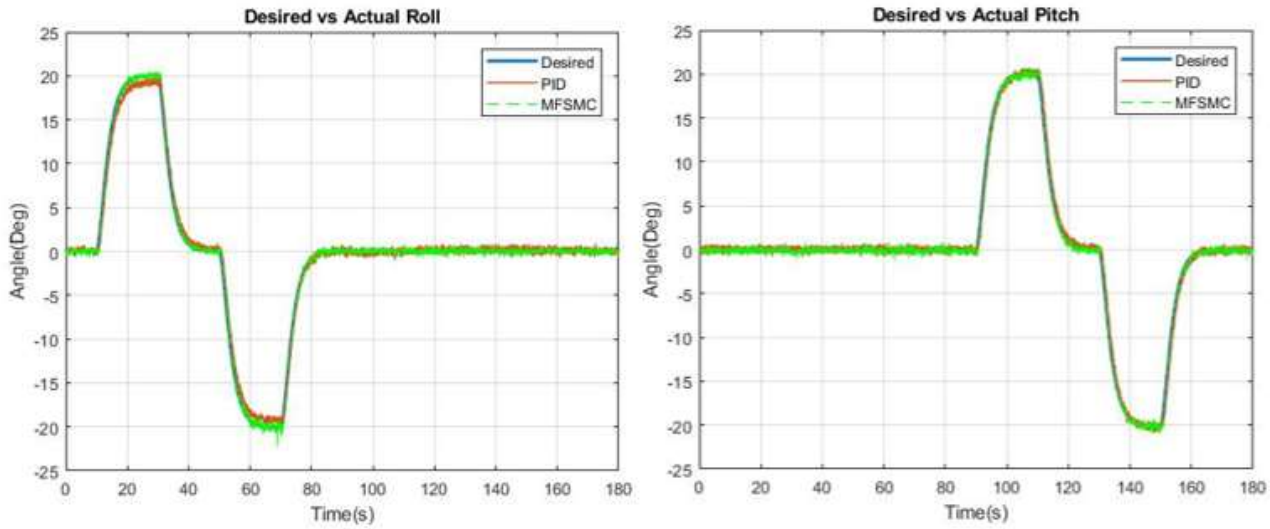

Fig. 4: Gimbal Flight Test of Roll and Pitch Angles Closed-loop Responses for PID and MFSMC Algorithms.

Table 3: RMS Tracking Errors Values for the PID and MFSMC Controllers for Flight Test Study.

Controller	**RMS**	
	Roll (deg)	Theta (deg)
PID	0.56	0.45
MFSMC	0.25	0.26

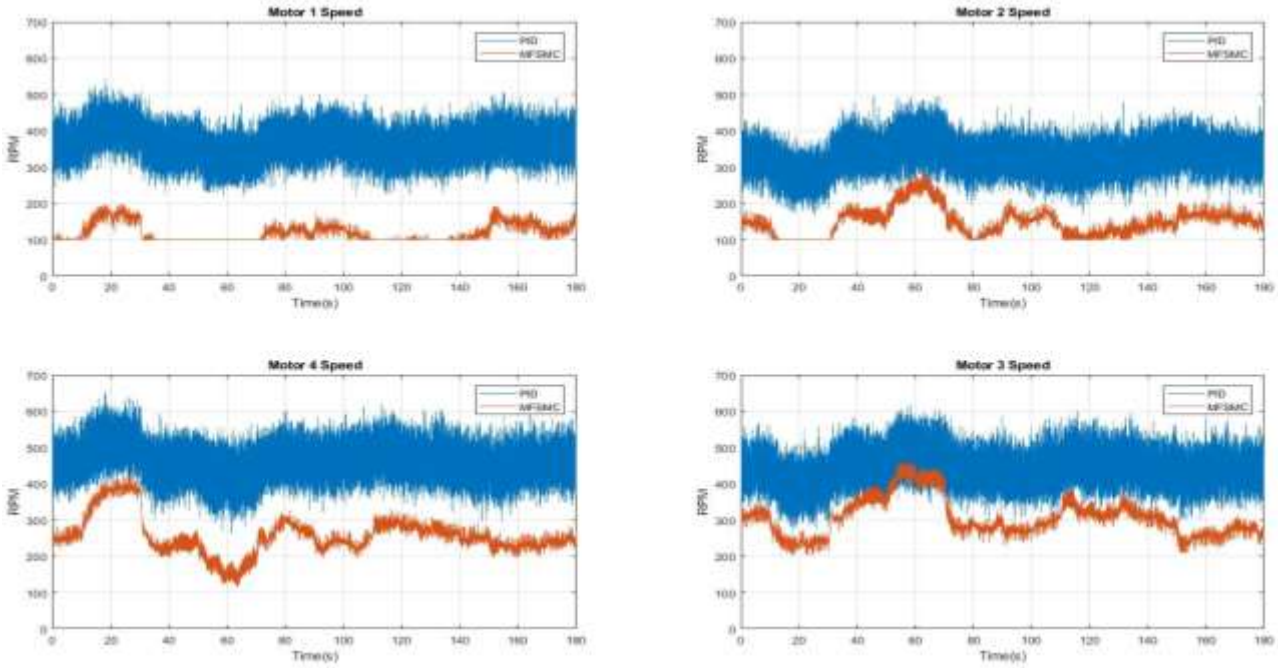

Fig. 5: Motor Speeds (RPM) Flight Test Responses for the PID and MFSMC Controllers.

Figure 5 indicates that the MFSMC algorithm operates the motors at a much lower speed as compared to the PID controller to track the desired roll and pitch angles. Additionally a 0.1 general thrust (*U1*) was given to the MFSMC while a 0.2 general thrust (*U1*) was required by the PID controller. Reducing the general thrust for the PID controllers depreciated the tracking performance. Hence this suggested that the MFSMC algorithm consumes less power than the PID controller indicating the MFSMC controller is highly efficient and robust.

7. Conclusion

A model-free control strategy based on the sliding mode control method was successfully integrated on a real-world quadcopter hardware platform to track roll and pitch angles with minimal tuning. The MFSMC algorithm achieved higher tracking precision as compared to a traditional tuned PID controller. Initial studies also suggested that the MFSMC consumed less power compared to the PID controller. Further investigations need to be conducted to confirm this result. The input power, voltage and current measurements and actual power consumed by the motors need to be investigated.

References

[1] J.-J. E. Slotine and W. Li, "Sliding Control," in *Applied Nonlinear Control*, 1st ed., Prentice-Hall, 1991.

[2] A. Villanueva, B. Castillo-Toledo, E. Bayro-Corrochano, L. F. Luque-Vega, and L. E. González-Jiménez, "Multi-mode flight sliding mode control system for a quadrotor," in *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2015, pp. 861–870.

[3] E. Abdulhamitbilal, "Robust flight sliding modes control system design for nonlinear aircraft with parameter uncertainties," in *2014 13th International Workshop on Variable Structure Systems (VSS)*, 2014, pp. 1–6.

[4] 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, 2015.

[5] A. Crassidis and R. M. Reis, "A Model-Free Sliding Mode Control Method," in *Proceedings of the 3rd International Conference of Control, Dynamic Systems, and Robotics*, Ottawa, 2016.

[6] A. Crassidis and F. E. Tin, "A Model-Free Control System Based on the Sliding Mode Control Method with Applications to Multi-Input-Multi-Output Systems," in *Proceedings of the 4th International Conference of Control, Dynamic Systems, and Robotics*, Ottawa, 2017.

[7] A. Crassidis and E. Schulken, "Model-Free Sliding Mode Control Algorithms including Application to a Real World Quadrotor.," in *Proceedings of the 5th International Conference of Control, Dynamic Systems, and Robotics*, Ottawa, 2018.

[8] R. Xu and Ü. Özgüner, "Sliding mode control of a class of underactuated systems," *Automatica*, vol. 44, no. 1, pp. 233–241, Jan. 2008.

[9] L. Derafa, A. Benallegue, and L. Fridman, "Super twisting control algorithm for the attitude tracking of a four rotors UAV," *Journal of the Franklin Institute*, vol. 349, no. 2, pp. 685–699, Mar. 2012.

[10] H.-- Ngo, T.- Nguyen, V.-- Huynh, T.- Le, and C.- Nguyen, "Experimental comparison of Complementary filter and Kalman filter design for low-cost sensor in quadcopter," in *2017 International Conference on System Science and Engineering (ICSSE)*, 2017, pp. 488–493.

[11] T. Islam, M. S. Islam, M. Shajid-Ul-Mahmud, and M. Hossam-E-Haider, "Comparison of complementary and Kalman filter based data fusion for attitude heading reference system," *AIP Conference Proceedings*, vol. 1919, no. 1, p. 020002, Dec. 2017.

[12] A. Levant, "Robust exact differentiation via sliding mode technique**This paper was recommended for publication in final form by Associate Editor Hassan Khalil under the direction of Editor Tamer Basar.," *Automatica*, vol. 34, no. 3, pp. 379–384, Mar. 1998.