Journal of Engineering Research and Reports

23(7): 42-58, 2022; Article no.JERR.91186
ISSN: 2582-2926

Implementation and Stabilization of a Quadcopter Using Arduino and the Combination of LQR and SMC Methods

Rahmi Elagib [a*] and Ahmet Karaasrlan [a]

[a] Department of Electrical and Electronics Engineering, Ankara Yildirim Beyazit University, Turkey.

Authors' contributions

This work was carried out in collaboration between both authors. Both authors read and approved the final manuscript.

Article Information

DOI: 10.9734/JERR/2022/v23i7735

Open Peer Review History:
This journal follows the Advanced Open Peer Review policy. Identity of the Reviewers, Editor(s) and additional Reviewers, peer review comments, different versions of the manuscript, comments of the editors, etc are available here: https://www.sdiarticle5.com/review-history/91186

Original Research Article

Received 20 June 2022
Accepted 30 August 2022
Published 12 October 2022

ABSTRACT

This paper presents a combination of Linear Quadratic Regulator (LQR) and Sliding Mode Control (SMC) methods to control a four-rotor unmanned aerial vehicle (UAV) that takes off and lands vertically (VTOL). Although controlling UAVs is difficult due to their highly nonlinear characteristics, the controller successfully controlled and stabilized the quadcopter in altitude and attitude by combining the advantages of the nonlinear and linear controllers. The Newton-Euler method is employed to build the dynamic model of the quadcopter, which is divided into two subsystems: the under-controlled subsystem and the fully actuated subsystem. The entire controller model was demonstrated in MATLAB/Simulink, and results demonstrating the controller's performance in various scenarios were obtained.

Keywords: Quadcopter; unmanned air vehicle; quadcopter; controller; SMC; LQR.

1. INTRODUCTION

Unmanned aerial vehicles (UAVs) have received a lot of attention in recent decades. Quadcopters, unmanned aerial vehicles with four rotors, have been the focus of most UAV research in recent years due to their simple design and high performance. The quadcopter is characterized by nonlinearity, instability, and susceptibility to external disturbances. Control system

**Corresponding author: Email: rahmyelageeb@gmail.com;*

development for quadcopters is an ongoing and expanding field of study. Researchers have devised several methods and strategies to control the quadcopter motion. The Sliding Mode Control (SMC) method is a robust and efficient nonlinear control strategy. It is used to control nonlinear systems under uncertain conditions. Due to its drawbacks, such as chattering, combining SMC with a control method that can eliminate these issues, such as the Linear Quadratic Regulator (LQR) method, can ensure a strong controller for stabilizing quadcopters. Many studies and scientists have developed LQR, SMC, and LQR&SMC techniques for controlling and stabilizing a quadcopter.

As shown in [1–4], the use of SMC in quadcopters has triggered a lot of interest. A. Noordin et al. [2] developed a sliding mode control to stabilize a quadrotor's altitude and attitude in the presence of external disturbances. The study looked at a "X-configuration quadcopter" and the saturation function, but the main flaw was that it ignored external disturbances. There are several LQR controllers designed for quadcopters in the literature [5–7]. Dhewa et al. [8] designed and implemented a quadrotor control system based on the LQR method to find the best value for the feedback gain K in the hover condition. The work was based on the results of experimental tests.

Although the controller's rise time on the quadrotor's roll and pitch angles is fast enough to overcome disturbances, when testing pitch angle, the controller was influenced by a large disturbance caused by environmental factors. In this work, the controller provided very low steady-state error and good quadrotor position control. Researchers have developed several control techniques that combine LQR and SMC control to control the attitude and attitude of quadcopters. Kaan T. Oner et al. [9] developed a quadcopter with a VLOT tilt-wing mechanism. They used a LQR controller to control the flight mode of the vehicle for all possible yaw angles, and SMC to stabilize the vehicle's attitude.

Even though the sliding mode controller did an excellent job of monitoring the reference inputs, the controller reference angles are quite large, and the controller achieves the appropriate roll and pitch angles in less than a second. Chi Yuan et al. [10] used the SMC in conjunction with a LQR to construct a controller for a forest firefighting quadcopter. The designed controller has two loops: an inner loop and an outer loop. The LQR controls the quadrotor positions, while the sliding mode controller controls the inner loop, which is responsible for attitude stabilization. Khaled A. Ghamry et al. [11] present a control strategy for the takeoff, tracking, and landing of an unmanned aerial vehicle (UAV) on an unmanned ground vehicle (UGV). The local UAV controller was a hybrid of SMC and LQR, while the UGV controller was a tracking-only strategy. During the takeoff, tracking, and landing phases, an SMC-based leader-follower formation controller approach was used. Yang et al. [12] consider a combination state-feedback control system for a small quadrotor that incorporates a robust SMC and optimal LQR methods in a hierarchical multilayer structure.

In this paper, a LQR&SMC controller combination is designed to stabilize the altitude and attitude of a quadrotor using MATLAB/ Simulink and an experimental demonstration. The designed quadrotor system takes nonlinearity, parameter uncertainties, and external disturbances into account. The main contribution of this work is that the controller controls and stabilizes the attitude of a quadcopter while considering the effects of external disturbances caused by quadcopter components such as the battery and sensors. This work discusses the following topics: Section II, Quadcopter and Dynamic Model of a Quadcopter Using the Newton-Euler Formulation Section III demonstrates the evolution of the LQR and SMC methods. In section IV, the entire model, including the SMC&LQR controller, is run through MATLAB/Simulink. Section V, Hardware for Quadcopters, Section VI contains the results, discussion, conclusion, and future work.

2. QUADCOPTER WORKING PRINCIPLE AND DYNAMIC MODEL

The quadcopter is a 6 degree of freedom underactuated system (6-DOF). The vehicle is made up of four propellers that are orthogonally arranged. The speed and direction of rotation of the propellers are independently controlled to ensure the vehicle's balance and movement. One pair of rotors rotates clockwise, while the other pair rotates counterclockwise, to keep the system balanced. The quadcopter can move forward, backward, and sideways by varying the speed of the rotors. The quadcopter's motion is classified into four types based on the relative motion of the four propellers: altitude, pitch, roll,

and yaw. Two frames could be used to measure the physical properties of a quadrotor: an Earth-fixed frame (E) and a Body-fixed frame (B). The Earth-fixed frame can be used to measure roll, pitch, yaw angles, and angular velocities, whereas the body-fixed frame can be used to measure linear acceleration. Fig.1. depicts an X-configuration with B and E frames. The quadcopter system has three translational states (x, y, z) three rotational states (ϕ, θ, ψ) and their derivatives ($\dot{x}, \dot{y}, \dot{z}, \phi\dot{\phi}, \dot{\theta}, \dot{\psi}$).

Where, x, y, and z are the position in the x, y, and z axes, $\dot{x}, \dot{y},$ and $\dot{z}$ are the speed in the axes. Where ϕ, θ, and ψ are the Euler angles which represent roll,pitch, and yaw respectively, and the parameters, $\dot{\phi}$, $\dot{\theta}$, and $\dot{\psi}$ are the speed for roll, pitch, and yaw. The quadrotor dynamics were expressed using Newton-Euler translational and rotational dynamics formulation [12] as:

$$\ddot{x} = \frac{U_1}{m}(\sin\psi\cos\phi + \cos\psi\sin\theta\cos\phi) \quad (1)$$

$$\ddot{y} = \frac{U_1}{m}(-\cos\psi\sin\ \phi + \sin\psi\sin\theta\cos\phi) \quad (2)$$

$$\ddot{z} = g + \frac{U_1}{m}(\cos\psi\cos\phi) \quad (3)$$

$$\ddot{\phi} = \left(\frac{I_{yy}-I_{zz}}{I_{xx}}\right)\dot{\psi}\,\dot{\theta} - \left(\frac{J_r\Omega_d}{I_{xx}}\right)\dot{\theta} + \left(\frac{l}{I_{xx}}\right)U_2 \quad (4)$$

$$\ddot{\theta} = \left(\frac{I_{zz}-I_{xx}}{I_{yy}}\right)\dot{\psi}\,\dot{\phi} - \left(\frac{J_r\Omega_d}{I_{yy}}\right)\dot{\phi} + \left(\frac{l}{I_{yy}}\right)U_3 \quad (5)$$

$$\ddot{\psi} = \left(\frac{I_{xx}-I_{yy}}{I_{zz}}\right)\dot{\theta}\,\dot{\phi} + \left(\frac{l}{I_{zz}}\right)U_4 \quad (6)$$

Where input signal U_1 is the total thrust of the 4 rotors, U_2, U_3 and U_4 are the moments for pitch, roll, and yaw respectively. m represents the mass of the quadrotor, l is the distance between the motor and the quadrotor center. J_r is the inertia of the rotor, and I_{xx}, I_{yy}, and I_{zz} are the inertia of the quadrotor in $x, y,$ and z respectively.

$$\mathrm{U}_1 = \mathrm{b}(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \quad (7)$$

$$\mathrm{U}_2 = \mathrm{b}\sin\left(\frac{\mathrm{pi}}{4}\right)(\Omega_1^2 - \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \quad (8)$$

$$\mathrm{U}_3 = \mathrm{b}\sin\left(\frac{\mathrm{pi}}{4}\right)(\Omega_1^2 + \Omega_2^2 - \Omega_3^2 - \Omega_4^2) \quad (9)$$

$$\mathrm{U}_4 = \mathrm{d}(\Omega_1^2 + \Omega_3^2 - \Omega_2^2 - \Omega_4^2) \quad (10)$$

$$\Omega_\mathrm{r} = \Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2 \quad (11)$$

Where b is the thrust coefficient and d is the drag coefficient. The angular speed for each rotor is $\Omega_1, \Omega_2, \Omega_3, \Omega_4$. Ω_r is the general angular speed.

2.1 The Controller

Sliding Mode Controller: Sliding Mode Control is a process that is derived from Variable Structure Control (VSC). The goal of the sliding mode control is to enforce the error in the system to a certain point. SMC law $\mathrm{U_X}$ Eqn.23. is composed of two major components: A continuous part (equivalent control $\mathrm{U_{eqX}}$) and a discontinuous part (switching control $\hat{\mathrm{U}}_\mathrm{X}$) as illustrated in Fig.2. The switching control rule enforces the system to the sliding surface, $\mathrm{S_X}$, which is set by the user, and it preserves the system state trajectory on this surface. The dynamic performance of the system is closely related to the choice of switching control rule. The equivalent control is utilized to ensure that the system state moves toward the sliding surface. Sliding control law design is essential to the controller in order to enforce error to the sliding surface.

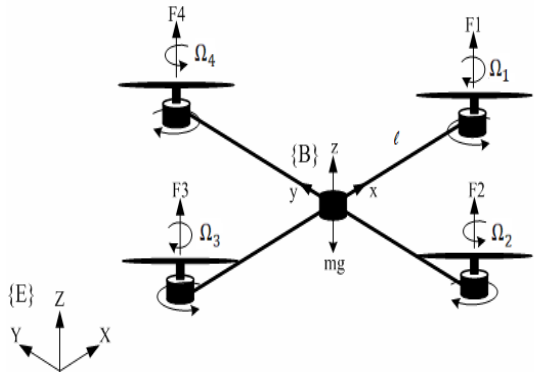

Fig. 1. The configuration of Quadcopter UAV

Sliding Mode Control is a Variable Structure Control-derived process (VSC). The sliding mode control's goal is to limit the system's error to a certain point. As shown in Fig.2., SMC law U_X Eqn.23. is made up of two major components: a continuous part (equivalent control U_{eqX}) and a discontinuous part (switching control $\hat{U}_X$). The switching control rule forces the system to the user-specified sliding surface, S_X, and maintains the system state trajectory on this surface. The choice of switching control rule has a strong influence on the system's dynamic performance. To ensure that the system state moves toward the sliding surface, the equivalent control is used. The controller requires the design of sliding control laws in order to enforce error to the sliding surface. The error should be defined in an attempt to develop the sliding surface or decision rule. As a result, the error is the difference between the actual value and the desired value, and this may be expressed mathematically as:

$$e_X = X_d - X \tag{12}$$

By calculating the error's first and second derivatives:

$$\dot{e}_X = \dot{X}_d - \dot{X} \implies \ddot{e}_X = \ddot{X}_d - \ddot{X} \tag{13}$$

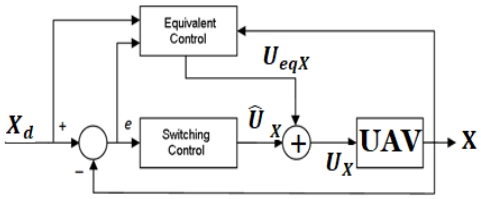

Fig. 2. Conventional sliding-mode control for a plant

The sliding surface is given according to the following equation:

$$S_X = \dot{e}_X + \lambda_X e_X \tag{14}$$

Also, the first derivative can be computing as:

$$\dot{S}_X = \ddot{e}_X + \lambda_X \dot{e}_X \tag{15}$$

Where variable e_X is the tracking error, and variable λ_X is the tuning parameter must satisfy the condition (λ_X>0), S_X is sliding surface $S_X = 0$, X is state space

A Lyapunov function is defined as:

$$(S_i) = \frac{1}{2} S_X^2 \tag{16}$$

then the necessary sliding condition is verified and Lyapunov stability is guaranteed. The chosen law for the attractive surface must satisfy:

$$S_X \dot{S}_X < 0 \tag{17}$$

then the desired sliding condition is verified and Lyapunov stability is guaranteed. The chosen law for the attractive surface must satisfy

$$S_X \dot{S}_X = -k_{X1} S_X - k_{X2} \text{sign}(S_X) < 0 \tag{18}$$

As mentioned, the control law, U_X , consists of two parts: a continuous part, U_{eqX} and a discontinuous part, $\hat{U}_X$.

$$U_X = \hat{U}_X + U_{eqX} \tag{19}$$

$$\hat{U}_X = -k_{X1} S_X - k_{X2} \text{sign}(S_X) \tag{20}$$

Where k_{X1}, k_{X2} are the tuning parameters, sign can be as:

$$\text{sign}(S_X) = \begin{cases} +1 & \text{if } S_X > 0 \\ -1 & \text{if } S_X < 0 \end{cases}$$

Linear Quadratic Regulator: LQR is an optimal control that has robust and produces a steady-state minimum error. Fig.3. shows the controller block diagrams for a quadcopter.

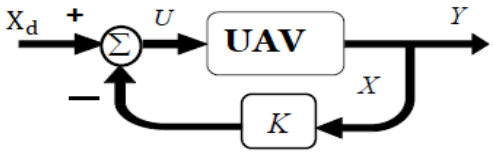

Fig. 3. Block diagram of LQR controller

In LQR, a cost function is minimized to provide the best control signal as :

$$J = \int_0^\infty ({}_xT\ Qx + {}_uT\ Ru)dt \tag{21}$$

where Q and R are weighting matrices. The gain matrix K can be obtained by solving the Ricatti equation as follow:

$$A^* P + PA - PB\ R^{-1}B * P + Q = 0 \tag{22}$$

With solved, the feedback gain matrix (K) would then be calculated using the formula below:

$$K = R^{-1}B * P \quad (23)$$

The linear dynamics in Fig.3. of x and y are given in the following state-space form to obtain the matrices A and B

$$\dot{x} = Ax + Bu \quad (24)$$

The controller aims to find the matrix of the optimal K of the optimal control vector u such that $u(t) = -Kx(t)$ and Eqn.24 will be $\dot{x} = (A - BK)x$ to minimize the quadratic cost function J.

MATLAB was used as it provides a convenient way of solving for K by just using the following command;

$$K = lqr(A, B, Q, R) \quad (25)$$

The Quadcopter Controller: The introduced UAV controller is based on a combination of sliding mode control (SMC) and linear quadratic regulator (LQR). Fig. 4. shows the control system block diagram, to simplify the design, the proposed control law is derived by dividing the system model into two subsystems, the fully actuated subsystem, and the underactuated subsystem. In the fully actuated subsystem, it is two outputs (z, ψ) for the inputs (U_1, U_4), whereas, in the under-actuated subsystem, inputs U_2 and U_3 are less than the number of outputs, which are (x, y, ϕ, θ). The dynamic model of the considered quadcopter is divided into two subgroups: a fully-actuated system and an under-actuated subsystem. Two outputs (z, ψ) were considered for the inputs (U_1, U_4) in the fully actuated subsystem. Contrary to this, in the under-actuated subsystem, inputs U_2 and U_3 are less than the total number of outputs which are (x, y, ϕ, θ).

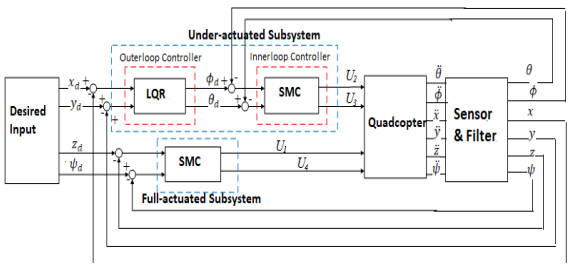

Fig. 4. Quadcopter configuration

Fully-actuated Subsystem Control: The objective of this fully-actuated subsystem controller is to minimize the error in the altitude and yaw angle e_z and e_ψ to satisfy the following conditions in Eqn.12 to Eqn.13

The control laws for the altitude and yaw angle can be derived using classical SMC theory:

$$\hat{U}_1 = \left(\frac{m}{\cos\phi\cos\theta}\right)(g + \ddot{z}_d + \lambda_z \dot{e}_z) \quad (26)$$

$$\hat{U}_4 = \frac{I_{zz}}{l}\left(\ddot{\psi}_d - \frac{I_x - I_y}{I_z}\dot{\theta}\,\dot{\phi} - \lambda_\psi \dot{e}_\psi\right) \quad (27)$$

By employing the Eqn. 19, 20 such that:

$$U_1 = \frac{m}{\cos\psi\cos\phi}(+g + \ddot{z}_d + \lambda_z \dot{e}_z) + k_{z1}S_z + k_{z1}si(S_z) \quad (28)$$

$$U_4 = I_{zz}\left(\ddot{\psi}_d - \frac{I_{zz} - I_{xx}}{I_{yy}}\dot{\theta}\,\dot{\phi} + \lambda_\psi \dot{e}_\psi\right) + k_{\psi 1}S_\psi + k_{\psi 1}si(S_\psi) \quad (29)$$

Under-Actuated Subsystem Control: The under-actuated subsystem in **Fig.4.**, is controlled via two independent loops. The outer and inner loop. The outer loop is designed to control the translational dynamics of the quadrotor to achieve trajectory tracking to the desired position in x and y axes. Its output is the reference orientation angles, roll θ_d, and pitch. According to these reference data, the inner-loop controller calculates the minimum system input U_2 and U_3. LQR is used for the outer-loop controller, while SMC is used for controlling the inner loop.

Outer loop: LQR is used in the outer loop for position control to obtain the desired altitude by converging the error and extracting the desired attitude angles θ_d and ϕ_d,

$U_x = (\cos\phi_d \sin\theta_d \cos\psi - \sin\phi_d \sin\psi)$, and $U_y = (\cos\phi_d \sin\theta_d \sin\psi - \sin\phi_d \cos\psi)$

By starting from the second-row ϕ_d and then θ_d can be calculated as follows:

$$\phi_d = \arcsin(U_x \sin\psi - U_y \cos\psi) \quad (30)$$

$$\theta_d = \arcsin((U_x \cos\psi - U_y \sin\psi)/\cos\phi_d) \quad (31)$$

The linear dynamics in **Eqn.1,2** of x and y are given in the following state-space form to obtain the matrices A and B as defined:

$$\ddot{x} = \phi g \quad (32)$$

$$\ddot{y} = -\theta g \quad (33)$$

$$A_x = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix},\ B_x = \begin{bmatrix} 1 \\ g \end{bmatrix},\ C_x = [1 \quad 0] \qquad (34)$$

$$A_y = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix},\ B_y = \begin{bmatrix} 1 \\ -g \end{bmatrix}, C_y = [1 \quad 0] \qquad (35)$$

As mentioned MATLAB was used to solve Riccatti equation and the **K** matrix was obtained **Eqn.30.**

Inner loop: SMC controller is the inner-loop controller is used to drive the attitude angles and ϕ, that are obtained from the outer-loop controller to their desired values θ_d and ϕ_d.

to achieve accurate stabilization through the following control laws:

$$U_2 = \frac{I_{xx}}{l}\left(\ddot{\phi}_d - \frac{I_{zz}-I_{xx}}{I_{yy}}\dot{\theta}\,\dot{\phi} + \lambda_\phi \dot{e}_\phi\right) + k_{\phi 1}S_\phi + k_{\phi 2} sign(S_\phi) \qquad (36)$$

$$U_3 = \frac{I_{yy}}{l}\left(\ddot{\theta}_d - \frac{I_{zz}-I_{xx}}{I_{yy}}\dot{\theta}\,\dot{\phi} + \lambda_\theta \dot{e}_\theta\right) + k_{\theta 1}S_\theta + k_{\theta 2} sign(S_\theta) \qquad (37)$$

where λ_ϕ and λ_θ are control gains with $\lambda_\phi > 0$ and $\lambda_\theta > 0$. e_θ and e_ϕ are the errors in roll and pitch angles, $e_\phi = \theta_d - \theta$, and $e_\phi = \phi_d - \phi$. θ_d and θ_d are the desired roll and pitch angles respectively.

3. SIMULATION RESULTS AND DISCUSSION

To verify the performance of the proposed controller, a MATLAB/Simulink is presented in this section. The parameters of the quadrotor used in the simulation are selected in Table 1.

The suggested SMC approaches' design parameters have been tuned manually in MATLAB/Simulink to track the trajectory smoothly. The LQR parameters, such as the state weighting matrix (Q) and control weighting matrix (R), were chosen to minimize the system's existing inaccuracy as well as its energy expenditure, as shown below, whereas the K value was obtained using MATLAB to solve the Riccati equation as in (16). Table 2 and 3 list the parameters of the recommended controllers.

Table 1. The parameters of the quadrotor

Parameter	Symbol	Value
Quadcopter mass	m	0.984 kg
Arm length	l	0.225m
Gravity	g	9.81 m/s^2
Rotor inertia	J_r	2.6e-06 $kg.m^2$
Inertia constants	$I_{yy} = I_{xx}$	9.5*10^{-3} $kg.m^2$
	I_{zz}	1.86*10^{-2} $kg.m^2$
Thrust coefficient	b	1.4865e-07$N.s^2$
Drag coefficient	d	2.925e-09 $N.m.s^2$
Aerodynamic coefficient	A_ϕ=A_θ=A_ψ	0 N/rad/s
Drag coefficient	A_x=A_y=A_z	0 N/m/s

Table 2. The design parameters tuning of SMC

Controller Tuning	SMC			
	Roll	Pitch	Yaw	Altitude
k_1	1.99	1.99	1	16
k_2	1.8	1.8	0	13.8
λ	2.68	2.68	18	2.2

Table 3. The design parameters tuning of LQR

Controller Tuning	LQR	
	X	Y
Q	[1 0;0 0]	[1 0;0 0]
R	0.001	[0.0001]
K	[31.6228 2.5391]	[-100 -4.5152]

To test roll, pitch, yaw, and altitude control, the simulation was run with the reference values shown in Fig.5. and Fig.6. As previously stated, the quadcopter is an underactuated system with six degrees of freedom, whereas the model has only four inputs that are used to control the vehicle so that x_d and y_d values are used to obtain phi and theta desired values as shown in Eqn.30. and Eqn.31. LQR is used to control these inputs and provide a fast response for the SMC. Fig.5. shows x_d , y_d , z_d , $\dot{\psi}_d$ values that utilized in this simulation. The obtained ϕ_d, and θ_d values are shown in Fig.6.

Note: The d refers to the desired value Fig.7., 8., 9., and 10. show the actual and desired values of altitude, roll, pitch, and yaw.

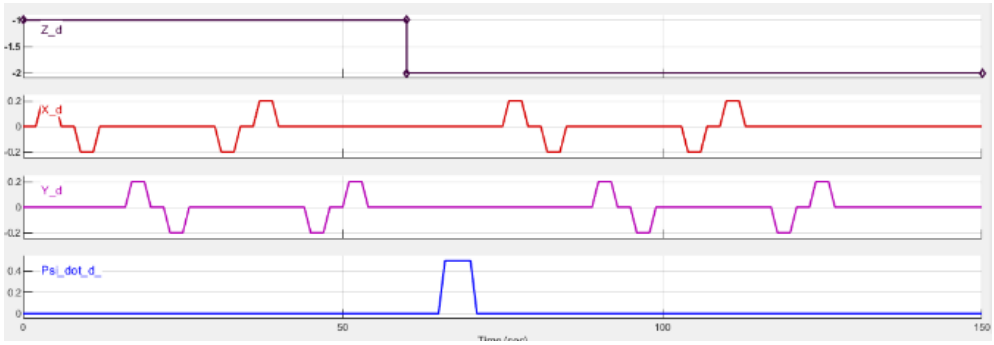

Fig. 5. x_d, y_d, z_d, $\dot{\psi}_d$ values

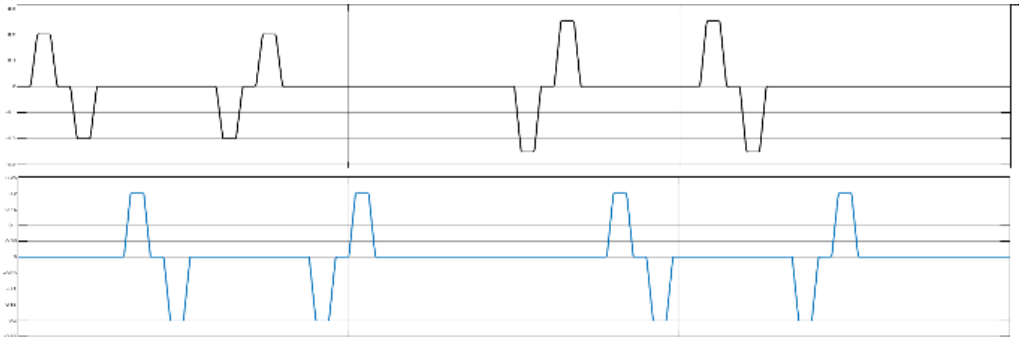

Fig. 6. ϕ_d, θ_d values obtained form LQR

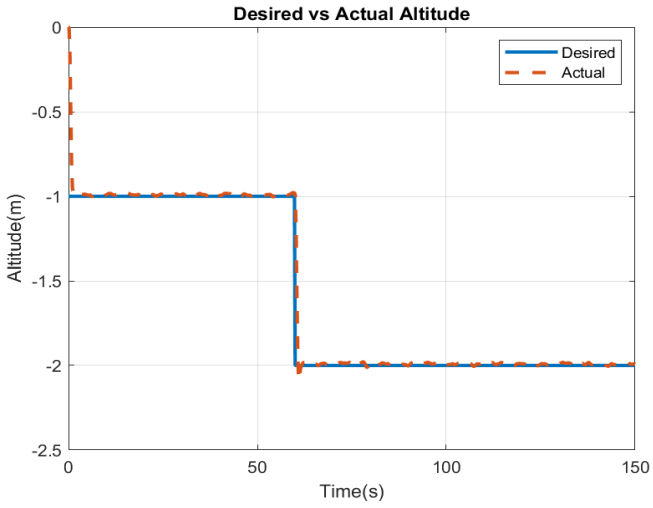

Fig. 7. shows the actual and desired altitude values

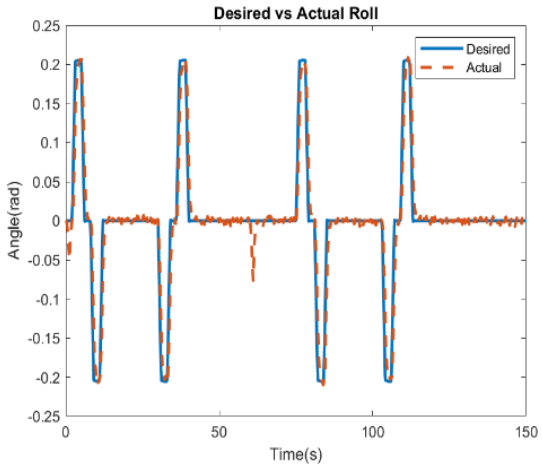

Fig. 8. shows the actual and desired roll values

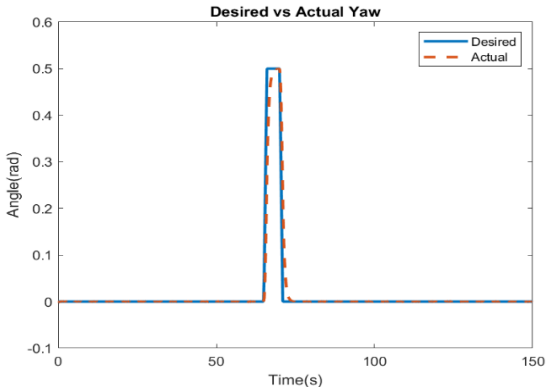

Fig. 9. shows the actual and desired yaw values

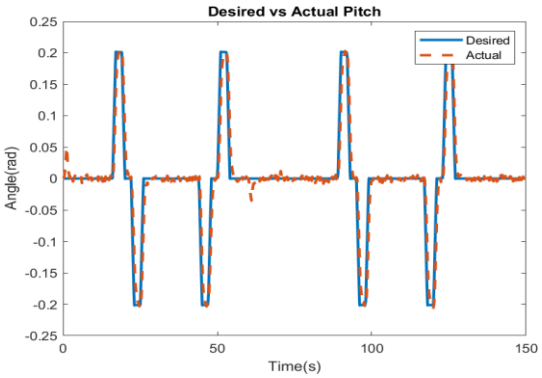

Fig. 10. shows the actual and desired pitch values

Fig. 7. show the desired and achieved altitudes, the SMC accomplishes pretty good results in maintaining the desired altitude. And the disturbances were kept to a minimum. In Fig. 8. and Fig.10. the roll and pitch angles were successfully accomplished with negligible disturbances. We can see that these two angles were affected, and slight disturbances emerged around 61.75 s, which is the exact moment that the desired altitude velocity, respectively was changed. In Fig. 9. the desired and actual yaw performance were shown. The controller, on the other hand, has kept yaw angles stable throughout the flight.

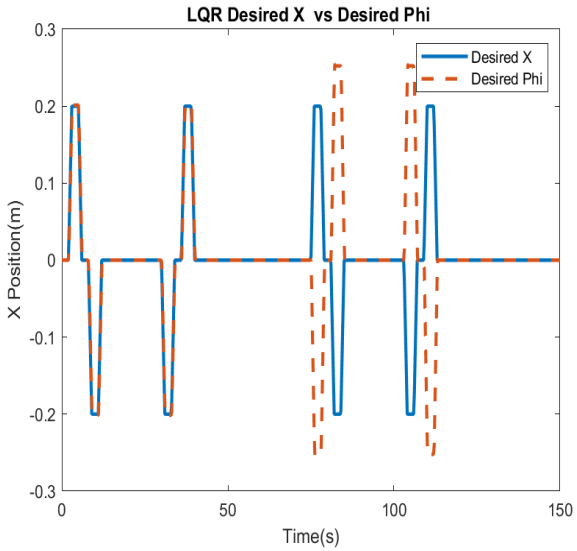

Fig.11. Controlling X Position with LQR

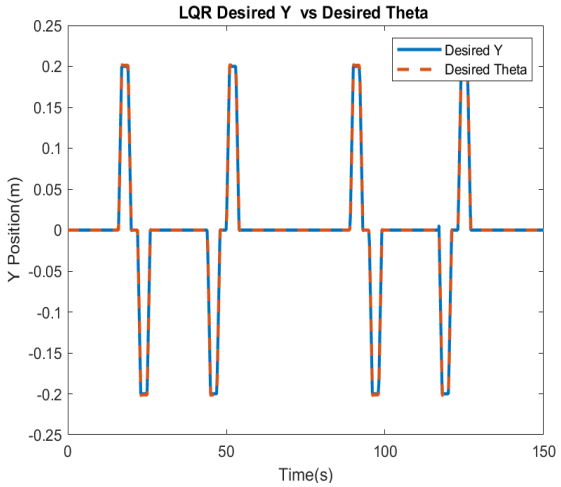

Fig.12. Controlling Y Position with LQR

As shown in Fig. 12, 13. The LQR performance is due to the use of weight matrices (Q, R) that were chosen to provide fast maintenance of desired roll and pitch angles; the controller follows the angles perfectly up to 75s but shows a poor performance for roll angles between 75s and 85.1 interval and 100s 110.2, causing system fluctuations.

An investigation has been conducted to determine the cause of the poor performance of the LQR X position controller. The reason was discovered after many attempts, and it was the cosine function in Eqn.31. The simulation results were shown in Fig.13 after rewriting the equations as shown in Eqn. 38 and simulating it in MATLAB/Simulink.

$$\theta_d = \arcsin((U_x \cos\psi - U_y \sin\psi) \quad (38)$$

The roll and pitch controllers' performance was improved by removing the cosine function, as shown in Fig.14. Although the deviations in Fig.8. and Fig.10. have been reduced, the other controllers continue to perform similarly.

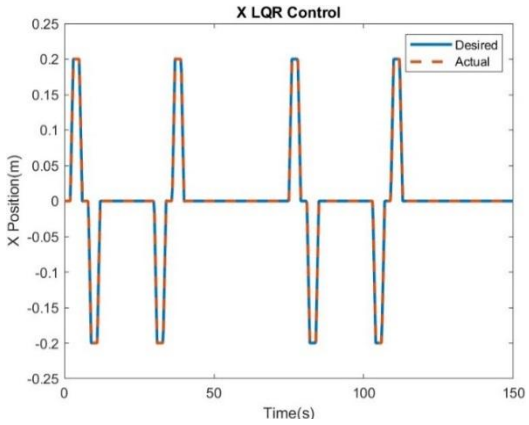

Fig.13. Controlling X Position with LQR

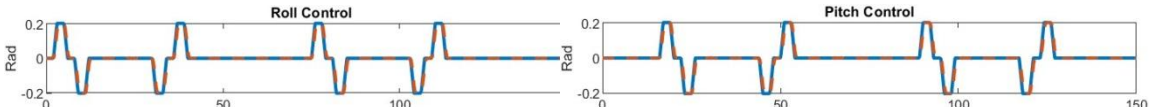

Fig.14. shows the actual and desired roll and pitch values

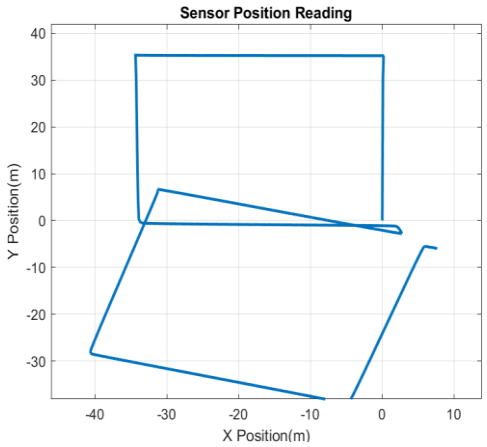

Fig.15. Quadcopter trajectory

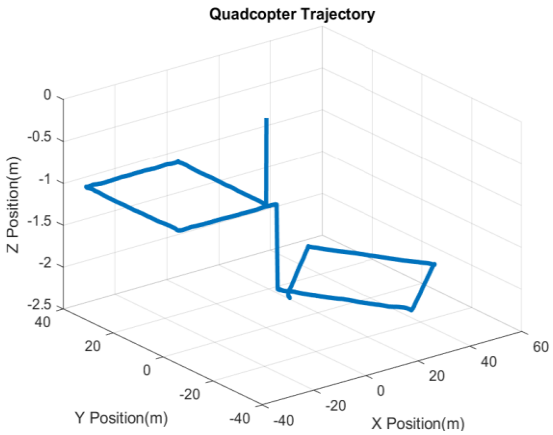

Fig.16. Quadcopter Sensor Position Reading

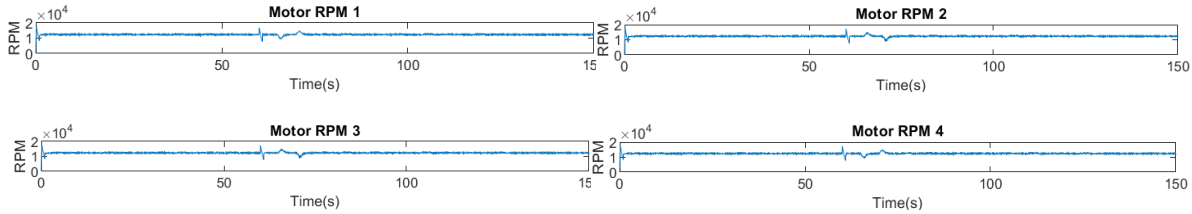

Fig.17. motors (speed)RPM

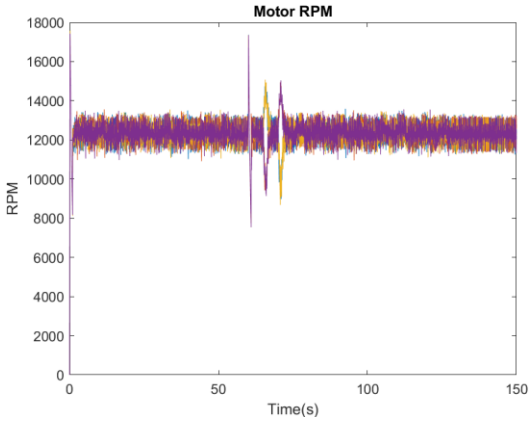

Fig.18. all motors speed

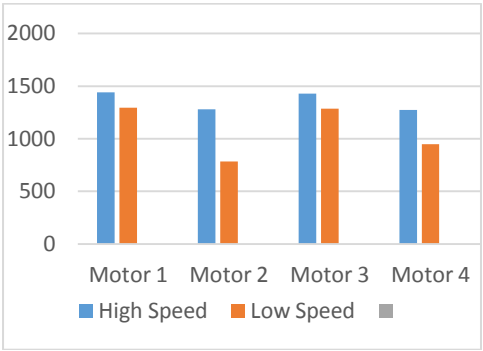

Fig.19. The motors' angular speed comparisons

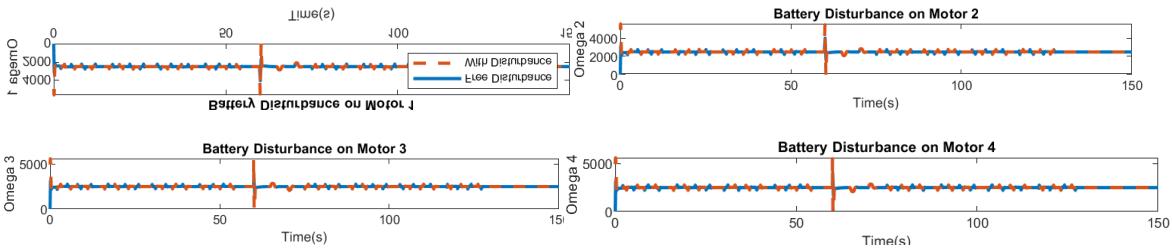

Fig.20. The motors' angular speed with disturbances caused by battery

A 3D trajectory was provided in Fig.15. as shown the trajectory was maintained successfully. The vehicle maintains the desired trajectory with only minor deviations. Instead of finishing the second square, the trajectory stopped before the last point and went down. And it appears that this is due to disturbances in the battery as will be shown later.

Fig.16. shows the performance of the entire system (the quadcopter with sensors and controls). It depicts the quadcopter's position in

the X, Y, and Z axes, as well as the importance of noting that a good trajectory was obtained with ignorable imprecision.

Fig.17. depicts the motors' speeds. As shown, the motors spin at roughly 12000 rpm, which was successfully maintained and controlled.

The battery disturbance and external disturbance were considered while simulating the motors' spinning. Fig.18. depicts the RPM fluctuations of a motor. Time 60s, 66s, and 70.2 s are all affected. The controller was able to keep up with the speed that had already been set.

The angular speeds of all four motors are shown in Fig.19. Even though the angular speed is disturbed at various time periods, these effects are negligible when compared to the controller.

It is shown in Fig.16. that each motor's desired angular speed is affected by battery disturbances. In the same way as in the preceding case, disturbances occur at the periods mentioned above.

Both SMC and LQR's controller rise, overshoot, and set-up times are shown in Table 4. (below).

X and Y positions are controlled by the LQR, as indicated, to achieve the desired values θ_d and ϕ_d.

Implementation:

a)The Quadcopter Hardware:

Multiple components are required to build a quadcopter. F450 quadcopter frame, 10X4.5 propellers, Readytosky 920 KVA 2212 brushless DC motor 920 KV, Simonk 30A ESC (electronic speed controller), MPU 6050 gyroscope sensors, Arduino Uno, FS-i6X transmitter/receiver, Lithium Polymer battery 2800 mAh, and iMAX RC B3 battery charger are among the components that used in this project. The hardware was connected together as shown in Fig.21.

After integrating all of the hardware together, the quadcopter was connected to the PC via USB cable after installing all of the necessary drivers and libraries required to establish communication between the hardware and the MATLAB/Simulink software. To test the designed controller, the simulink model shown in Fig.22. was built and deployed to the Arduino.

Table 4. Controllers rise and setting times and overshoot.

State Measures	Rise time(ms)	Overshoot (%)	Stelling time(ms)
X Position	801.59	Inf	-
Y Position	0.0	13.09	-
Altitude	-	2.0	213.163(fs)
Roll	2.053	15.875	-
Pitch	2.616	1.832	-
Yaw	1.473	0.505	-

Fig. 21. The Quadcopter Hardware Implementation

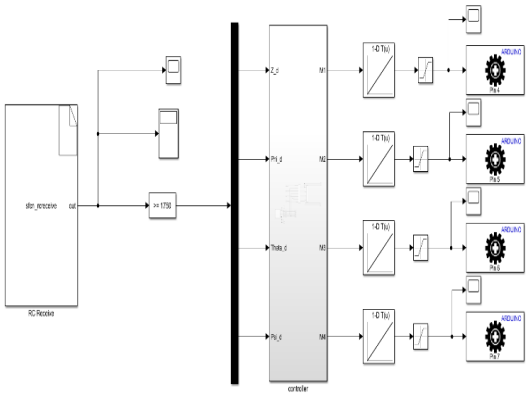

Fig. 22. The quadcopter controller block diagram in MATLAB/Simulink

The controller subsystem of the implemented model on real hardware shown above consists primarily of a controller block, a receiver, and Arduino pins. The Simulink-designed LQR and SMC, as well as the MPU6050 IMU sensor, are included in the controller model. The Simulink MPU6050 Arduino block reads data from the MPU6050 IMU sensor on the quadcopter hardware. The block outputs calibrated values for acceleration, angle rates, sensor status, and magenotometer. There is also a motor mixing algorithm within the control simulink model that controls the rotor speed based on the signals sent. Another crucial component is the RC receiver, which sends a servo-style PWM signal directly to the motor ESC, which is sent by radion transmitter. This block is intended to read four transmitter channels that control roll, pitch, yaw, and throttle signal.

b)The Quadcopter Testing:

After completing the simulation to demonstrate that the controller can stabilize the quadrotor with the current combination of sensors and designed controller. Here is an attempt to validate the results by implementing both the LQR controller and the SMC in the quadrotor prototype. All components were tested, but the main test was the sensor test, because noise from the sensor was a significant factor. The goal of angular rate sensor testing, also known as gyroscope (or gyro) testing, is to ensure that sensor errors and noise are sufficiently ignored.

Practically, the test was done by place the vehicle on a table and the then roll, pitch, and yaw sensors' readings were measured in order to calibrate the sensor. Following the previous test, all quadcopter motors were run at the same speed to see if there was any noise on the roll, pitch, and yaw signals. A neglectable noise appears on the roll, pitch, and yaw sensor readings when all the motors are running at the same speed, but the problem arises when attempting to obtain quadcopter movements. The vehicle was lifting the vehicle around its x and y axes to measure the roll and pitch angles. In addition, the vehicle was lifted and rotated to determine the yaw angle. Figs 23, 24, and 25. illustrate the quadcopter's roll, pitch, and yaw test results.

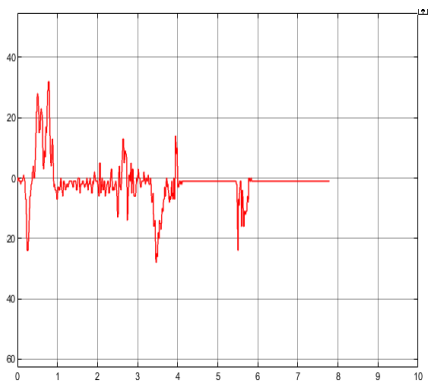

Fig. 23. Roll Sensor Reading

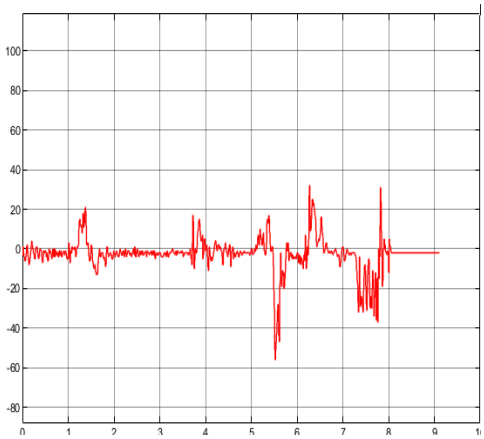

Fig. 24. Pitch Sensor Reading

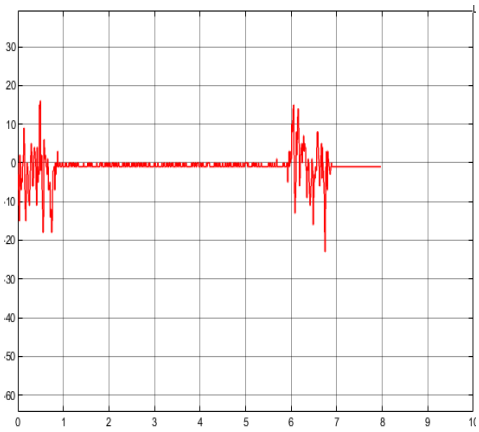

Fig. 25. Yaw Sensor Reading

As shown in Fig. 23., a noise appears just before to the quadcopter posative rolling between 0.1 and 0.3 seconds, resulting in negative rolling. At 2.55, 2.8, and 4 seconds, the same issure it seems. Fig. 24. Disturbances appear fairly throughout the quadcopter's running time in the sensor pitch reading, but there is noticeable noise at 1.5 to 1.7, 5.2, and 7.8 seconds. Fig. 25. The quadcopter sensor performs reasonably well in terms of yaw disturbances, as shown in the figure, but noise appears only occasionally at 0 to 0.5 and 6 to 6.5 seconds.

c)The Flight Controller:

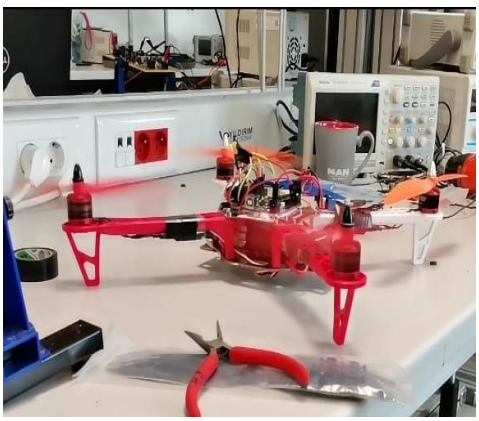

Fig. 26. The quadcopter flight test

Table 5. A comparison of quadcopter performance in Simulink and flight test

	Simulink Result	Flight Results
Roll	A bit disturbed	A bit disturbed
Pitch	A bit disturbed	Poor
Yaw	A bit disturbed	Poor
Throttle	Good	Good

Table 6. The battery's cell voltage

Cell 1	Cell 2	Cell 3
3.747V	3.752V	3.750V

A flight test was obtained after doing all the sensor tests to calibrate the quadcopter. The quadcopter has been mounted on a table so that it can be controlled; the thrust produced by the propellers forces the quadcopter to rotate around the axis and go up and down as described in the work principal section. The roll and pitch angle controllers were tested to ensure that they were tuned while the controller simulation model ran on the hardware. The same test was used to ensure the yaw angle, which was accomplished by lifting the quadcopter in a way that allowed it to freely rotate around its body axis. The quadcopter's rotation was also noticed. Finally, the throttle was tested by rapidly increasing all the quadcopter motor speeds, and the quadcopter altitude increased.

It is worth noting that, while the data indicates that the control system responded properly to the reported sensor values, the actual flight test produced very different results.

The above-mentioned flight test lasted 32 seconds due to a failure in one of the ESCs; additional analysis was carried out to identify the cause. The reason appeared to be the battery's voltage, as the voltage of the battery's cells differs from each other, as shown in Table 6.

3. CONCLUSION AND FUTURE WORKS

A combination of Linear Quadratic Regulator (LQR) and Sliding Mode Control (SMC) to control a quadcopter is comprehensively presented in this paper. For the control design, a mathematical model, based on the Newton-Euler equations, is derived. The system is subdivided into fully-actuated and under-actuated subsystems. The under-actuated subsystem mainly consists of two loops (inner loop and outer loop), which are used to control the position and attitude of the vehicle. LQR is used for the outer loop, while SMC is used for the inner loop. The full-actuated subsystem is an SMC used for its altitude and yaw angle control.

The proposed design improves the robustness of the controller while considering the effects of external disturbances caused by the quadcopter. In addition, the chattering phenomena caused by SMC are alleviated by the proposed control law. Furthermore, the LQR controller reduces computational time without affecting accuracy. A detailed simulation study is carried out and the obtained results are discussed. Although, the altitude and heading angle(yaw) maintained with neglectable distrbances, the roll and pitch angles showed small distrbances while changing

altitude, the SMC&LQR controller combination is robust and effectively stabilized the quadcopter and minimized external disturbances. While the LQR controller demonstrates ideal trajectory tracking in pitch angle, it performs badly between 75s and 110.2s, yet the whole controller was able to function well and overcome the difficulty.

A flight test of 32 seconds was completed, and the controller performed excellently in terms of roll angle stabilization. In contrast to the simulation, a small fluctuation in pitch angle was detected during the flight, as shown in sensor tests, but more testing is required to further analyze the problem. Unpredictable performance was observed by the yaw controller, as the controller shows some poor performance in comparison to simulation and tests, which would need to be thoroughly studied in future works. The controller performed reasonably well in altitude stabilization. Although the controller did a fairly good job of stabilizing the at all quadcopter, flying the vehicle for an extended period was difficult due to the battery's unpredictable behavior, which led to the failure of one of the ESCs. The voltages across the cells are too far apart, resulting in battery instability. Another issue was sensor noise, particularly when the quadcopter was performing pitch movements.

To summarize, the proposed design is very effective for such quadcopters. In future work, the current MATLAB/Simulink model controller could be improved by combining other controllers, such as Fuzzy with SMC, because LQR caused some disturbances that could be eliminated. The data can be filtered using the Kalman filter to obtain more accurate data from IMU sensors, which can improve pitch control performance. The SMC parameters are tuned using the trial-and-error method, so the controller parameters can be tuned using the genetic algorithm (GA) to ensure the best outputs. Adafruit BNO055 9-DOF, which appears to be more accurate and commonly used in quadcopter projects, could be used instead of the MPU 6050 sensor.

COMPETING INTERESTS

Authors have declared that no competing interests exist.

REFERENCES

1. Herrera M, Chamorro W, Gómez AP, Camacho O. Sliding Mode Control: An Approach to Control a Quadrotor. Proceedings - 2015 Asia-Pacific Conference on Computer-Aided System Engineering, APCASE. 2015:314–319. DOI: 10.1109/APCASE.2015.62.
2. Noordin MAM, Basri, Mohamed Z. Sliding mode control for altitude and attitude stabilization of quadrotor UAV with external disturbance, Indonesian Journal of Electrical Engineering and Informatics. 2019;7(2):203–210. DOI: 10.11591/ijeei.v7i2.1149.
3. Zhao R. Inner / Outer-Loop Control of a Drone under Disturbances and Uncertainties ; A Sliding Mode Control Approach; 2020.
4. Matouk D, Abdessemed F, Gherouat O, Terchi Y. Second-order sliding mode for position and attitude tracking control of quadcopter UAV: Super-twisting algorithm," International Journal of Innovative Computing, Information and Control. 2020;16(1):29–43. DOI: 10.24507/ijicic.16.01.29.
5. Zhang X, Li X, Wang K, Lu Y. A survey of modelling and identification of quadrotor robot, Abstract and Applied Analysis; 2014. DOI: 10.1155/2014/320526.
6. Okyere E, Bousbaine A, Poyi GT, Joseph AK, Andrade JM. LQR controller design for quad-rotor helicopters, The Journal of Engineering. 2019;17:4003–4007. DOI: 10.1049/joe.2018.8126.
7. Ortega–Vidal F, Salazar–Vasquez, Rojas–Moreno A. A comparison between optimal LQR control and LQR predictive control of a planar robot of 2DOF. in 2020 IEEE XXVII International Conference on Electronics, Electrical Engineering and Computing (INTERCON). 2020:1–4. DOI: 10.1109/INTERCON50315.2020.9220263.
8. Dhewa OA, Dharmawan A, Priyambodo TK. Model of linear quadratic regulator (LQR) control method in hovering state of quadrotor. Journal of Telecommunication, Electronic and Computer Engineering. 2017;9(3):135–143,.
9. Oner KT, Cetinsoy E, Sirimoglu E, Hancer C, Ayken T, Unel M. LQR and SMC stabilization of a new unmanned aerial vehicle, World Academy of Science, Engineering and Technology. 2009;58: 373–378. DOI: 10.5281/zenodo.1056267.
10. Yuan C, Ghamry KA, Liu Z, Zhang Y. Unmanned aerial vehicle based forest fire

monitoring and detection using image processing technique, CGNCC 2016 - 2016 IEEE Chinese Guidance, Navigation and Control Conference. 2017:1870–1875. DOI: 10.1109/CGNCC.2016.7829074.

11. Ghamry KA, Zhang Y. Formation control of multiple quadrotors based on leader-follower method, International Conference on Unmanned Aircraft Systems, ICUAS. 2015:1037–1042. DOI: 10.1109/ICUAS.2015.7152394.

12. Project MEM. Modeling, Simulation and Complete Control of a Quadcopter. 2017:1–76.

Peer-review history:
The peer review history for this paper can be accessed here:
https://www.sdiarticle5.com/review-history/91186