Formal Verification of Quadcopter Flight Envelop Using Theorem Prover

Omar A. Jasim[1] and Sandor M. Veres[2]

***Abstract*— Quadcopter controllers are in use today and in practice they can often cope well in non adverse weather conditions such as lack of strong sudden gusts of wind around the corner of a building or no frequent demands of travel directions by remote control or a guidance law. Different payloads can alter the boundaries of the stable state space region of a drone, its flight envelop, beyond which its autopilot may not be able to regain stable control of the craft. For fixed gain autopilot controllers, reaching the boundary of the flight envelop can be caused by (1) external disturbance like gusts of wind and turbulence, (2) altered drone mass and its distribution and (3) reduction or misalignment of thrust output in the propulsion system caused ware after multiple uses of the drone. This paper introduces symbolic computation to map out the numerical boundaries of controller tolerances in terms of these three factors that affect the autopilots ability to retain stability of the craft. Proof theoretic methods are developed that can be applied to quadcopter of various nominal parameters.**

I. INTRODUCTION

Unmanned aerial vehicles (UAVs) such as quadcopters have received a considerable attention in resent years from researchers and engineers in both academia and industry. This is due to their usefulness in emerging applications through their flying abilities and low cost relative to manned aircraft. These systems are mostly auto-piloted systems that are now often used in applications such as surveillance, inspection, search and rescue, fire fighting, etc. Some applications need these vehicles to fly near structures which require flight accuracy, high manoeuvrability and speed of response abilities. Sometimes they also need to withstand high levels of disturbances under variable payloads. Therefore, these systems are need to be robust and safe to fly in order to accomplish the given task.

For manoeuvrability these UAV systems have highly nonlinear dynamics, and they are typically under-actuated, for instance quadcopters having four inputs and six degree of freedom. Combine this with altered dynamics through ware and varying payloads and robust control of such systems is becoming a challenging task. There have been a variety of controllers proposed are designed to tackle these challenges. Ultimately, the implemented controllers will need to be officially certified by aviation authorities in various countries, subject to agreements directives by organisations such as International Civil Aviation Organization (ICAO), Joint Authorities for Rulemaking on Unmanned Systems (JARUS), European Aviation Safety Agency (EASA) and member organisations such as the Civil Aviation Authority (CAA) in the UK. These organisations eventually needs to promote the creation of standards which eventually ensures systems safety in practice. This paper highlights the advantages of formal analysis applied not only to onboard software but also to the analysis and proof of performance of autopilot controllers in terms of their robustness of their control loops.

Until now most controllers have been derived by "pen and paper" by control engineers and encoded in software. Verification has been limited to checking that the manually defined controller is correctly encoded. In this paper we advocate the principle, probably first time in this context, that formal analysis through computer-based symbolic computation, can be and should be applied to prove the robustness properties of autopilot controllers. The results of this formal analysis can map out the boundaries of a safe flight envelop more precisely and the numerical results can then be used in decision making of advanced autopilots to abort or reduce the mission goals when it becomes apparent during the flight that there are dangers of the controller failing and potentially crashing the craft. In general, computations of flight envelop boundaries can be more laborious to carry out manually and can also be less reliable than proof theoretic computations on a computer.

Formal methods [1] are tools which use mathematical logic in addition to techniques form automated reasoning for specification and verification software and hardware systems using symbolic computations. A well-known technique in formal methods is automated theorem proving (ATP). ATP is used to prove mathematical formulae automatically through First-Order Logic (FOL) format [2]. These methods are widely used in control systems verification. However, the three stages of formally verifiable controller design is illustrated in [3]. Where control system verification consist mainly of two parts: model-based and code-based verification. In model-based verification, after formalising and designing the control system depending on given performance specifications, mathematical models are obtained. Then, control system can be implemented using computer aided design (CAD) tools and can be verified using formal software verification methods. Code then can be generated using several techniques such as in [4]–[6] where this process called code-based verification.

This paper focuses on model-based verification of the quadcopter controller stability using formal methods such as ATP and symbolic computation. We use Metatarski [7]

[1]Omar A. Jasim is a PhD. student with the department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, UK oajasim1@sheffield.ac.uk

[2]Sandor M. Veres is a professor with the department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, UK s.veres@sheffield.ac.uk

ATP to verify controller stability since it works on real numbers to prove algebraic inequalities for the flight envelop. As inequalities are widely used in controllers design and analysis such as in robust control and Lyapunov analysis, it is possible for Metitarski to verify these inequalities which ensures system robustness and stability.

There are some published work on verification of control systems using different formal methods. The nearest approach to our research is in [8] where Nichols plot Requirements Verifier (NRV) is used to automatically implement formal analysis using Maple and PVS [9] proof assistant. Akbarpour and Paulson [10] also used Metitarski to formally prove the validity of the control system of inverted pendulum and a disk drive reader using Nichols plot analysis. In [11], Denman and his colleagues verified the stability of flight autopilot controller in terms of Nichols plots using Metitarski.

A new attitude controller is presented for quadcopters to illustrate the power of controller verification by theorem proving. Our example is based on the well known robust inverse dynamics approach [12]–[15]. Controller design is analysed using the Lyapunov method to guarantee that the system is asymptotically stable. Then, controller stability is verified by translating the derivative of Lyapunov function to a FOL formula and implementing it in the Metitarski theorem prover.

II. QUADCOPTER DYNAMICS

The basic model of the quadcopter is shown in Fig. 1. The quadcopter from its name is consist of four motors, the front M_1 and rear M_3 motors rotate clockwise while the other two motors, M_2 and M_4, rotate counter-clockwise. This configuration enables the quadcopter vehicle to cancel the effect of the moments produces by each pair of motors. The unmanned quadcopter consists of two movements: the transitional and rotational. The first determines the vehicle position in the world (inertial) frame while the second, which we are considered in this paper, determines the vehicle attitudes. The quadcopter moves forwards and backwards when the propeller angular velocity ω_1 of M_1 reduces/increases and ω_3 of M_3 increases/reduces by the same amount while keeping the total thrust constant. The forward/backward motion is determined by the pitch angle θ around the Y_B-axis while the right/left motion is determined by the roll angle ϕ around the X_B-axis. Both pitch and roll angles are calculated from the position controller and passed to the attitude controller for calculating the rotational pitch and roll torques τ_θ, τ_ϕ respectively. The rotation around the Z_B-axis is determined according to the given yaw angle ψ by increasing/decreasing the propeller angular velocity of the motors pair M_1 and M_3 and decreasing/increasing it for the motors pair M_2 and M_4, since the yaw rotational torque τ_ψ is determined from the given ψ angle.

The derivation of the quadcopter attitude dynamics is based on Euler-Lagrange rigid body rotational dynamics for controlling the quadcopter rotational motion. From the

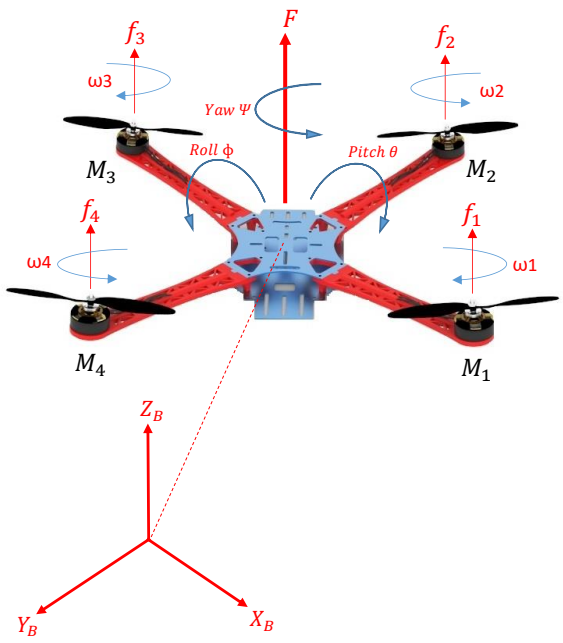

Fig. 1. Quadcopter configuration

Lagrangian definition, the rotational kinetic energy is

$$T_R = (1/2)\Omega^T I \Omega, \tag{1}$$

where $\Omega = [\Omega_x\ \Omega_y\ \Omega_z]^T \in \Re^3$ is the angular velocities vector in the rigid body frame $B = [X_B\ Y_B\ Z_B]^T$ and $I \in \Re^{3\times3} = diag[I_x\ I_y\ I_z]$ is the positive definite inertia matrix. Then from the kinematic relationship [16] between the Euler rates vector $\dot{\eta} = [\dot{\phi}\ \dot{\theta}\ \dot{\psi}]^T \in \Re^3$, since $\eta(t) = [\phi(t)\ \theta(t)\ \psi(t)]^T \in \Re^3$ is a vector represents Euler angles roll, pitch and yaw respectively, and the body angular velocities vector Ω, for the Euler angles sequence [Z Y X] is

$$\Omega = W\dot{\eta}\ ,\ \begin{bmatrix}\Omega_x\\ \Omega_y\\ \Omega_z\end{bmatrix} = \begin{bmatrix}1 & 0 & -S(\theta)\\ 0 & C(\phi) & C(\theta)S(\phi)\\ 0 & -S(\phi) & C(\theta)C(\phi)\end{bmatrix}\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix}, \tag{2}$$

where S and C is related to *sin* and *cos* respectively. From (1) and (2), we have the rotational energy

$$T_R = (1/2)(\dot{\eta})^T J(\eta)\dot{\eta}, \tag{3}$$

where,

$$J(\eta) = W^T I W = \begin{bmatrix} I_x & 0 & -I_x S(\theta) \\ 0 & I_y C^2(\phi) + I_z S^2(\phi) & (I_y - I_z)C(\phi)S(\phi)C(\theta) \\ -I_x S(\theta) & (I_y - I_z)C(\phi)S(\phi)C(\theta) & I_x S^2(\theta) + I_y S^2(\phi)C^2(\theta) + I_z C^2(\phi)C^2(\theta) \end{bmatrix} \tag{4}$$

is the Jacobian symmetric positive definite matrix (is invertible) which transfers the angular velocities Ω in the B-frame to their corresponding Euler rates $\dot{\eta}$. The quadcopter attitude dynamics in B-frame using Euler-Lagrange equation is:

$$J(\eta)\ddot{\eta} + C(\eta,\dot{\eta})\dot{\eta} + d = \tau, \tag{5}$$

where $\ddot{\eta}$ is Euler acceleration of the vehicle in the B-frame and

$$C(\eta,\dot{\eta}) = \dot{J}(\eta) - \frac{1}{2}\frac{\partial}{\partial \eta}(\dot{\eta}^T J(\eta)), \tag{6}$$

is the Coriolis matrix which contains the gyroscopic and centripetal terms where the total matrix is shown in [17]. $\tau = [\tau_\phi \;\; \tau_\theta \;\; \tau_\psi]^T \in \Re^3$ is the control torque vector (the three body moments control inputs vector) which produces the quadcopter motion. $d \in \Re^3$ is the vector represents the unknown disturbances.

Each motor has an angular velocity ω that produces vertical force f where

$$f_i = k\omega_i^2 \tag{7}$$

and moments

$$m_i = b\omega_i^2 \tag{8}$$

where k and b are the lift and drag constants respectively. The input to the system, τ, is

$$\tau = \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} \ell k(-\omega_1^2 + \omega_3^2) \\ \ell k(-\omega_2^2 + \omega_4^2) \\ b(-\omega_1^2 + \omega_2^2 - \omega_3^2 + \omega_4^2) \end{bmatrix}, \tag{9}$$

where ℓ is the length from the centre of mass of the quadcopter to each rotor. From (5) and (9), the attitude dynamics equation becomes

$$\ddot{\eta} = J^{-1}(\eta)[\tau - N(\eta,\dot{\eta}) - d], \tag{10}$$

where $N(\eta,\dot{\eta}) = C(\eta,\dot{\eta})\dot{\eta}$.

III. CONTROL DESIGN

A nonlinear controller is designed for the quadcopter using inverse dynamic control method with parameters uncertainty and disturbances. Robust control is also used to bound the uncertainty then Lyapunov function is used to guarantee asymptotic stability of the the control system. Assuming the roll ϕ and pitch θ angles are limited as

$$-\frac{\pi}{2} < \phi < \frac{\pi}{2}\ ,\ -\frac{\pi}{2} < \theta < \frac{\pi}{2} \tag{11}$$

and by defining the nonlinear control law as

$$\tau = \hat{J}(\eta)u + \hat{N}(\eta,\dot{\eta}) + \hat{d} + \gamma, \tag{12}$$

were u represents a new input vector to be designed later, $\hat{J}(\eta)$ is an estimated matrix of the Jacobian matrix $J(\eta)$, $\hat{N}(\eta,\dot{\eta})$ is the nominal vector of $N(\eta,\dot{\eta})$ and the additional term γ is added to render the uncertainty of the system which will be defined later; hence from (12), equation (5) becomes

$$J(\eta)\ddot{\eta} + N(\eta,\dot{\eta}) + d = \hat{J}(\eta)u + \hat{N}(\eta,\dot{\eta}) + \hat{d} + \gamma. \tag{13}$$

Assumption 1: Assume that an estimate $\hat{d}$ of the disturbance d is known, with an error term $\Delta d = \hat{d} - d$ which is known to be bounded by D and $\bar{D}$ as

$$\|\Delta d\| \le D, \quad \|d\| + D < \bar{D} \tag{14}$$

Assumption 2: Assuming that the error between the estimated vector $\hat{N}(\eta,\dot{\eta})$ and the actual $N(\eta,\dot{\eta})$ vector, $\Delta N(\eta,\dot{\eta})$, is also bounded by upper bound as

$$\|\Delta N(\eta,\dot{\eta})\| \le S. \tag{15}$$

Suppose that the desired rotational vector is η_d and $\dot{\eta}_d$ is to be controlled, then the tracking error defined as,

$$e = \eta_d - \eta \tag{16}$$

$$\dot{e} = \dot{\eta}_d - \dot{\eta} \tag{17}$$

where η and $\dot{\eta}$ are the measured Euler angles and Euler rates respectively. Given $\ddot{\eta}_d$, the $\dot{\eta}_d$ can be obtained by integration and the control input u in (12) is defined by

$$u = \ddot{\eta}_d + K_r\dot{e} + K_\eta e = \ddot{\eta}_d + K_r(\dot{\eta}_d - \dot{\eta}) + K_\eta(\eta_d - \eta) \tag{18}$$

where $K_r = diag[k_{r_1}\ k_{r_2}\ k_{r_3}] \in \Re^{3\times3}, K_\eta = diag[k_{\eta_1}\ k_{\eta_2}\ k_{\eta_3}] \in \Re^{3\times3}$ are positive-definite diagonal gain matrices. From (13), we have

$$\begin{aligned} \ddot{\eta} &= \hat{J}(\eta)J^{-1}(\eta)u + J^{-1}(\eta)[\Delta N(\eta,\dot{\eta}) + \Delta d] \\ &\quad + J^{-1}(\eta)\gamma \\ &= u + (\hat{J}(\eta)J^{-1}(\eta) - I)u + J^{-1}(\eta)[\Delta N(\eta,\dot{\eta}) + \Delta d] \\ &\quad + J^{-1}(\eta)\gamma \\ &= u - v + J^{-1}(\eta)\gamma \end{aligned}$$

where

$$v = [I - \hat{J}(\eta)J^{-1}(\eta)]u - J^{-1}(\eta)[\Delta N(\eta,\dot{\eta}) + \Delta d]. \tag{19}$$

From (16) - (19), we have the error dynamic as

$$\ddot{e} + K_r\dot{e} + K_\eta e = v - J^{-1}(\eta)\gamma, \tag{20}$$

then by setting $E \in \Re^{6\times1}$ as

$$E = \begin{bmatrix} e \\ \dot{e} \end{bmatrix} \tag{21}$$

the closed-loop error dynamics equation is obtained

$$\dot{E} = AE + B[v - J^{-1}(\eta)\gamma] \tag{22}$$

where

$$A = \begin{bmatrix} 0^{3\times3} & I^{3\times3} \\ -K_\eta^{3\times3} & -K_r^{3\times3} \end{bmatrix},\ B = \begin{bmatrix} 0^{3\times3} \\ I^{3\times3} \end{bmatrix}. \tag{23}$$

To bound the error, the uncertainty v need to be bounded and this can be achieved by using robust control technique then γ needs to be defined using Lyapunov function. The control input u in addition to the term γ should guarantee asymptotic stability for any v varying within the bounded range, were v is uncertain but an estimation on its range of variation can be obtained.

Assumption 3: From (19), the following assumptions have been chosen in order to bound the term v

$$sup(\|\ddot{\eta}_d\|) < H \tag{24}$$

$$\|I - \hat{J}(\eta)J^{-1}(\eta)\| \le \xi \le 1, \tag{25}$$

and for the matrix $J(\eta)$, in addition to the positive-definite matrix property, it should has an upper and lower limited bounds

$$\beta_{min} \leq \|J^{-1}(\eta)\| \leq \beta_{max}. \tag{26}$$

IV. STABILITY ANALYSIS

Lyapunov direct method [13] is used to define the term γ and to guarantee that the system error converges to zero. By setting the equilibrium point $E = 0$ where $V(0) = 0$ and defining the following positive-definite function

$$V(E) = E^T QE > 0 \ , \ \forall E \neq 0 \tag{27}$$

were $Q \in \Re^{6\times 6}$ is a symmetric positive-definite matrix. The time derivative of the function $V(E)$ along the trajectory of the error system is

$$\begin{aligned} \dot{V}(E) =& \dot{E}^T QE + E^T Q\dot{E} \\ =& E^T[A^T Q + QA]E + 2E^T QB(v - J^{-1}(\eta)\gamma), \end{aligned} \tag{28}$$

since A has eigenvalues with all negative real parts, hence for any symmetric positive-definite matrix P, we have

$$A^T Q + QA = -P, \tag{29}$$

which gives a unique solution Q. Therefore, the term $E^T[A^T Q + QA]E$ in (28) is negative and the equation will be

$$\dot{V}(E) = -E^T PE + 2E^T QB(v - J^{-1}(\eta)\gamma). \tag{30}$$

As the term $-E^T PE$ in the above equation is negative definite, then if $E \in G(B^T Q)$ the solution is converge. If $E \notin G(B^T Q)$ then γ must be chosen to render the second term of the above equation to less than or equal to zero. The term γ has been chosen as

$$\gamma = \begin{cases} \dfrac{\delta(E)}{\|B^T QE\|} B^T QE & \|B^T QE\| \geq \sigma \\ \dfrac{\delta(E)}{\sigma} B^T QE & \|B^T QE\| < \sigma \end{cases} \tag{31}$$

where $\delta(E)$ is a positive time-varying scalar. Assuming that $\|B^T QE\| \geq \sigma$, then we have

$$\begin{aligned} E^T QB(v - J^{-1}(\eta)\gamma) \leq& \|B^T QE\|\|v\| - \beta_{max}\delta(E)\|B^T QE\| \\ =& \|B^T QE\|(\|v\| - \beta_{max}\delta(E)) \end{aligned} \tag{32}$$

and if we choose $\delta(E)$ as

$$\delta(E) \geq \frac{\|v\|}{\beta_{max}} \tag{33}$$

then form (14), (15), (19), (24), (25), and (26), we have

$$\begin{aligned} \|v\| \leq& \|I - \hat{f}(\eta)J^{-1}(\eta)\|(\|\ddot{\eta}_d\| + \|K_r\|\|\dot{e}\| + \|K_\eta\|\|e\|) \\ & - \|J^{-1}(\eta)\|(\|\Delta N(\eta,\dot{\eta})\| + \|\Delta d\|) \\ \leq& \xi(H + \|K_r\|\|\dot{e}\| + \|K_\eta\|\|e\|) - \beta_{max}(S + D) \end{aligned} \tag{34}$$

from previous two equations, we get

$$\delta(E) \geq \frac{\xi}{\beta_{max}}(H + \|K_r\|\|\dot{e}\| + \|K_\eta\|\|e\|) - S - D \tag{35}$$

Finally, (30) becomes

$$\dot{V}(E) = -E^T PE + 2E^T QB(v - J^{-1}(\eta)\frac{\delta(E)}{\|B^T QE\|} B^T QE) < 0 \tag{36}$$

or

$$\dot{V}(E) = -E^T PE + 2E^T QB(v - J^{-1}(\eta)\frac{\delta(E)}{\sigma} B^T QE) < 0 \tag{37}$$

Definition 1: A set $Inv(\eta_d, \bar{D}) \subset R^6$ is called a *control enabled set*, if for any $[\eta^T, \dot{\eta}^T]^T \in Inv(\eta_d, \bar{D})$ there are continuous functions $\dot{\eta}_d \ \ddot{\eta}_d, \ t > 0$, so that at time t

$$u = \ddot{\eta}_d + K_r\dot{e} + K_\eta e = \ddot{\eta}_d + K_r(\dot{\eta}_d - \dot{\eta}) + K_\eta(\eta_d - \eta) \tag{38}$$

is realisable by the motors of the drone under the constraints of

$$\tau = \hat{J}(\eta)u + \hat{N}(\eta,\dot{\eta}) + \hat{d} + \gamma, \tag{39}$$

where τ is in (9) and $0 < \omega_1 < \omega_1^{max}$, $0 < \omega_2 < \omega_2^{max}$, $0 < \omega_3 < \omega_3^{max}$, $0 < \omega_4 < \omega_4^{max}$.

Control enabled sets can be numerically computed for various values of their parameters η_d and $\bar{D}$.

Theorem 1: Assuming (36)-(37) are verified to be satisfied over a control enabled set $Inv(\eta_d, \bar{D}) \subset R^6$, then the state evolution of $[\eta^T, \dot{\eta}^T]^T$ defined by

$$\ddot{\eta} = u - v + J^{-1}(\eta)\gamma, \ t > 0 \tag{40}$$

remains in $Inv(\eta_d, \bar{D})$ for any $\|\hat{d}\| \leq \bar{D}, \ t > 0$, for the controllers as defined by (19) and (9) and a suitable choice of adapted references $\dot{\eta}_d$ and $\ddot{\eta}_d$.

Proof: Fairly straightforward from (27)-(37).

The next section illustrates the application of the results in Simulink/Matlab.

V. SIMULATION

The controller is implemented in Simulink/Matlab for testing with the nonlinear quadcopter dynamics in (5). In order to test the attitude controller, the simulation based on a simple cascaded P position controller which calculates the roll and pitch angles from a given trajectory (X,Y,Z). The initial roll ϕ, pitch θ and yaw ψ angles are set to zero. According to the given trajectory, attitude controller shows that the measured roll, pitch and yaw angles are followed the references as can be seen in Fig. 2 - 7. The controller parameters are obtained and listed in Table I which are used in the verification process later.

From (29), the positive definite matrix P is chosen then the symmetric positive definite matrix Q is obtained as

$$P = \begin{bmatrix} 9*10^{-12} & 0 & 0 & 0 & 0 & 0 \\ 0 & 9*10^{-12} & 0 & 0 & 0 & 0 \\ 0 & 0 & 5*10^{-9} & 0 & 0 & 0 \\ 0 & 0 & 0 & 3*10^{-8} & 0 & 0 \\ 0 & 0 & 0 & 0 & 3*10^{-8} & 0 \\ 0 & 0 & 0 & 0 & 0 & 8*10^{-4} \end{bmatrix} \tag{41}$$

$$Q = \begin{bmatrix} 2*10^{-7} & 0 & 0 & 0 & 0 & 0 \\ 0 & 2*10^{-7} & 0 & 0 & 0 & 0 \\ 0 & 0 & 4.6*10^{-4} & 0 & 0 & 0 \\ 0 & 0 & 0 & 3.8*10^{-6} & 0 & 0 \\ 0 & 0 & 0 & 0 & 3.8*10^{-6} & 0 \\ 0 & 0 & 0 & 0 & 0 & 8.2*10^{-4} \end{bmatrix} \tag{42}$$

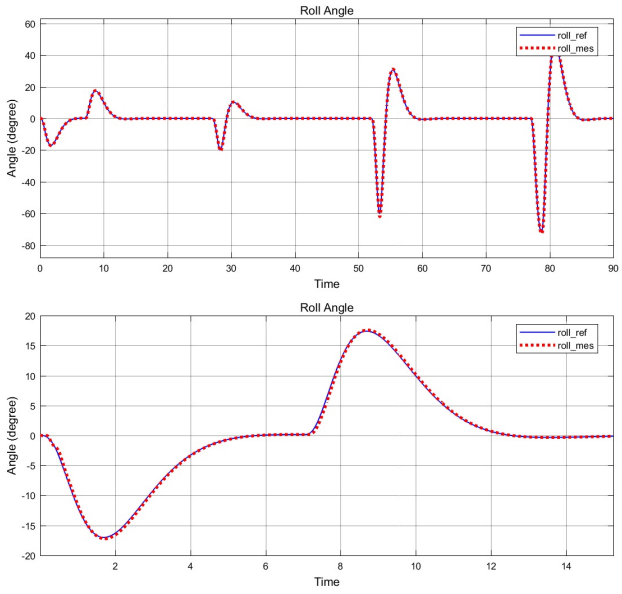

Fig. 2. Roll angle

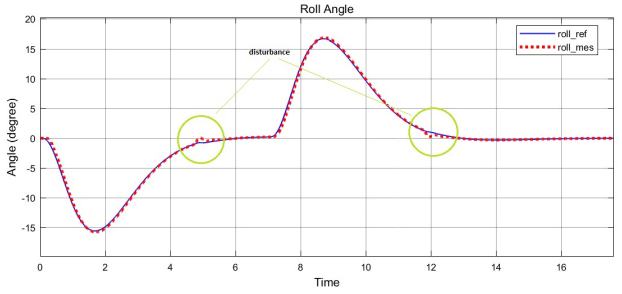

Fig. 3. Roll angle with disturbance

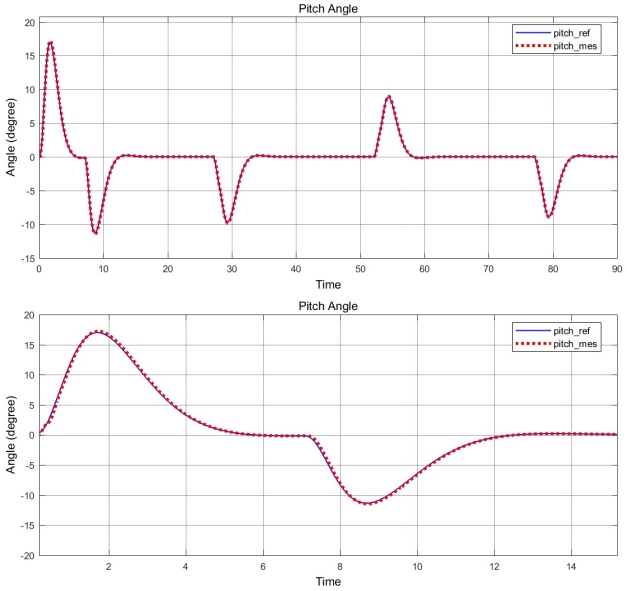

Fig. 4. Pitch angle

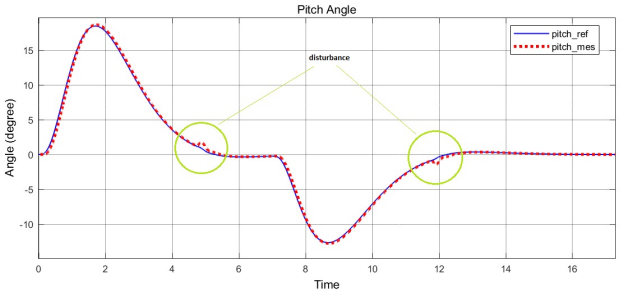

Fig. 5. Pitch angle with disturbance

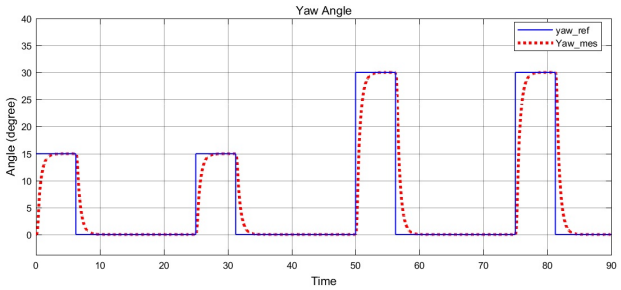

Fig. 6. Yaw angle

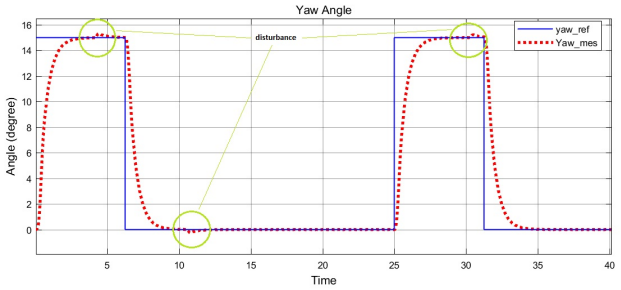

Fig. 7. Yaw angle with disturbance

TABLE I
QUADROTOR PARAMETERS

Parameter	Value
$\hat{I}_x$	$5.831 * 10^{-3}$
$\hat{I}_y$	$5.831 * 10^{-3}$
$\hat{I}_z$	$1.166 * 10^{-2}$
k_{η_1}	17.5
k_{η_2}	17.5
k_{η_3}	1.8
k_{r_1}	0.004
k_{r_2}	0.004
k_{r_3}	0.4826
ℓ	20 cm
k	$12 * 10^{-8}$
b	$9 * 10^{-6}$
H	1.2
ξ	0.5
S	$1 * 10^{-3}$
D	$1 * 10^{-3}$
β_{min}	173
β_{max}	170.5
σ	$9 * 10^{-13}$

VI. CONTROLLER STABILITY VERIFICATION

To ensure that the control system is asymptotically stable using symbolic computations, equation (36) and (37) should be strictly negative with the given assumptions. Simulation can not guaranteed that this is valid for all possible values as it is relying on numerical computations. Therefore, there is a need to check the validity of Lyapunov stability using symbolic computations. This can be done using theorem provers such as Metitariski. The following subsections will demonstrate Metitarski prover and the validity of the controller stability using this prover.

A. METITARISKI

Metitariski is an automated theorem prover based on a First-Order Logic(FOL) which works on real numbers field. It is designed to solve universally quantified inequalities problems including transcendental and some special functions such as *log*,*ln*, *exp*, *sin*, *cos*, *sqrt*, etc. This tool is useful especially in control laws as these functions and inequities on large scale real number are needed. As the above controller is designed with robust assumptions which include inequalities on real numbers to bound the variables in the control system in addition to Lyapunov function which also needs such inequalities, we have chosen Metitariski to verify the stability of the quadcopter under these assumptions. Metitarski is consisting of a resolution theorem prover (Metis) [18] which is works with disjunctions of inequalities and a decision procedure (QEPCAD) [19] which works on finding and removing inconsistent inequalities in the clauses. Metitarski is able to invoke three reasoning tools which are QEPCAD, Mathematica and Z3 [20] in order to perform the proof.

B. LYAPUNOV STABILITY VERIFICATION

Due to the limitations of Metitariski prover as it is a FOL system which means that it works on real scalar values without the ability to work with vectors and matrices, Lyapunov equations (36) and (37) have been simplified using Matlab symbolic toolbox and then formalised to the FOL format to accomplish the verification task. All codes that we formalized in Metitariski prover to verify the control system stability can be found in our web-repostory [1], as it is too long to be included here. An example of the code is shown below:

FOL format in Metitariski prover for E(1) value of equation (36)

```
%%%%%%%%%% Variables Definition %%%%%%%%%%

fof(QCD_Lyap_eq1_E1,conjecture, ![E_1,E_2,
E_3,E_4,E_5,E_6,Phi,Theta,V_1]:?[Delta_E_1]:

%%%%%%%%%%%%%% assumptions  %%%%%%%%%%%%%%

(E_1 != 0 & abs(E_1) <= 3.1415 & abs(E_4)
<=54.9778 & E_2 != 0 & abs(E_2) <= 3.1415
& abs(E_5) <=54.9778 & E_3 != 0 & abs(E_3)
<= 3.1415 & abs(E_6) <= 5.6547 & Phi > -90
& Phi <90 & Theta > -90 & Theta <90 & V_1
<=(0.5*(1.2+(0.004*abs(E_4))+(17.5*abs(E_1)))
- (170.5*(0.001+0.001))) & Delta_E_1 > 0
& Delta_E_1 >= ((0.5/170.5)*(1.2+(0.004*
abs(E_4))+(17.5*abs(E_1)))) - 0.001 - 0.001

%%%%%%%%%%%%%%%% implies  %%%%%%%%%%%%%%%%

=> (-((E_1)^2*0.000000000009) + ((
(4450461475223171*E_4)/590295810358705651712)-
((2720083094133*E_1)/302231454903657293676544)
*(V_1-((Delta_E_1/abs(((4450461475223171*E_4)
/1180591620717411303424)-((2720083094133*E_1)
/604462909807314587353088))) * (
((((2720083094133*E_1)/352462322708645150720 0
- (4450461475223171*E_4)/6884029740403225600)
* (3410592571054305266 5656*sin(Phi)^2
- 3410592571054305266 5656*sin(Phi)^4
+ 441201133756893860614960062919497109375
*sin(Theta)^2 -
441201133756893894720885773462549775031
*sin(Phi)^2*sin(Theta)^2+
3410592571054305266 5656*sin(Phi)^4*sin(Theta
)^2-88255364892869834128704086736896000000))
/(cos(Theta)^2*(4263240713817881583207
*cos(4*Phi)
+ 88255364892869833702380015355107841 6793))
- (sin(Theta)*((1152921504628125*E_3)/128
- (2388634804372428437 5*E_6)/8)
*(2100118578235409606 3*sin(Phi)^2
+ 21008395415778906 25))/
(cos(Theta)^2*(4263240713817881583207*cos(4*Phi)
+88255364892869833702380015355107841 6793))
-(cos(Phi)*sin(Phi)*sin(Theta)
*((1142499408066481811231449680479492 1875
*E_2)/33554432
-(584156051614821800514039035010263671875*E_5)
/2048))/(cos(Theta)*(4263240713817881583207*cos
(4*Phi)+88255364892869833702380015355107841 6793
))))))))) < 0)).

include('Axioms/general.ax').
include('Axioms/pow.ax').
include('Axioms/abs.ax').
include('Axioms/sin.ax').
include('Axioms/cos.ax').
```

As can be seen above, in the first line, fof related to first-order logic and the quantifiers (!) and (?) means *for any* and *for some* respectively, which are used to indicate variables quantifier. The symbol $=>$ means *implies* which indicate that the lines before this symbol are assumptions and after is the statement to be proven. After *implies*($=>$), Lyapunov equation (36) with the first element scalar value of the error vector $E(1)$, which is $E1$ in the above code, is implemented in Metitariski and it shows that the formula is satisfy the given assumptions for all possible values within the given bounds. All the error e and error rate $\dot{e}$ values in E vector are bounded based on the assumption in (11) as $0 < |E(1,2,3)| \leq 3.1415$, the error rates $0 < |E(4,5)| \leq 54.9778$ and $0 < |E(6)| \leq 5.6547$, where all values are in radians. Variables notated in Metitarski are shown in Table (II). The verification process performed for all error values in E vector for both (36) and (37) to complete the controller stability verification process.

[1] *https://github.com/Formal-Methods-of-Robotics/Quadcopter*

TABLE II

VARIABLES AND VECTORS NOTATIONS IN METITARSKI

Variable/Vector	Notation
ϕ	*Phi*
θ	*Theta*
ψ	*Psi*
$E(i)$	*E_i*
$v(i)$	*V_i*
$\delta(E)$	*Delta_E*

VII. CONCLUSION

We presented in this paper a model-based verification technique by symbolic computations to verify quadcopter stability based on Lyapunov's direct method using the Metitariski theorem prover. A nonlinear robust attitude controller is presented using inverse dynamics control method with system uncertainty and disturbances. The control system implemented in Simulink/Matlab and the results have been shown. The verification process results show that control system stability can be verified using ATP to guarantee asymptotic stability of the controller and to ensure that the system works within the given bounds and performance specifications.

REFERENCES

[1] E. M. Clarke and J. M. Wing, "Formal methods: State of the art and future directions," *ACM Computing Surveys (CSUR)*, vol. 28, no. 4, pp. 626–643, 1996.

[2] M. Fitting, *First-order logic and automated theorem proving*. Springer Science & Business Media, 2012.

[3] O. A. Jasim and S. M. Veres, "Towards formal proofs of feedback control theory," in *2017 21st International Conference on System Theory, Control and Computing (ICSTCC)*, Oct 2017, pp. 43–48.

[4] E. Feron, "From control systems to control software," *IEEE Control Systems*, vol. 30, no. 6, pp. 50–71, 2010.

[5] R. J. Jobredeaux, "Formal verification of control software," Ph.D. dissertation, Georgia Institute of Technology, 2015.

[6] T. Wang, "Credible autocoding of control software," Ph.D. dissertation, Georgia Institute of Technology, 2015.

[7] L. C. Paulson, "Metitarski: Past and future," in *International Conference on Interactive Theorem Proving*. Springer, 2012, pp. 1–10.

[8] R. Hardy, "Formal methods for control engineering: A validated decision procedure for nichols plot analysis," Ph.D. dissertation, University of St Andrews, 2006.

[9] S. Owre, J. M. Rushby, and N. Shankar, "PVS: A prototype verification system," in *International Conference on Automated Deduction*. Springer, 1992, pp. 748–752.

[10] B. Akbarpour and L. C. Paulson, "Applications of metitarski in the verification of control and hybrid systems," in *International Workshop on Hybrid Systems: Computation and Control*. Springer, 2009, pp. 1–15.

[11] W. Denman, M. H. Zaki, S. Tahar, and L. Rodrigues, "Towards flight control verification using automated theorem proving," in *NASA Formal Methods Symposium*. Springer, 2011, pp. 89–100.

[12] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot modeling and control*. Wiley New York, 2006, vol. 3.

[13] J.-J. E. Slotine, W. Li, *et al.*, *Applied nonlinear control*. prentice-Hall Englewood Cliffs, NJ, 1991, vol. 199, no. 1.

[14] L. Sciavicco and B. Siciliano, *Modelling and control of robot manipulators*. Springer Science & Business Media, 2012.

[15] B. L. Stevens, F. L. Lewis, and E. N. Johnson, *Aircraft control and simulation: dynamics, controls design, and autonomous systems*. John Wiley & Sons, 2015.

[16] B. Etkin and L. D. Reid, *Dynamics of flight: stability and control*. Wiley New York, 1996, vol. 3.

[17] G. V. Raffo, M. G. Ortega, and F. R. Rubio, "An integral predictive/nonlinear H∞ control structure for a quadrotor helicopter," *Automatica*, vol. 46, no. 1, pp. 29–39, 2010.

[18] J. Hurd, "First-order proof tactics in higher-order logic theorem provers," *Design and Application of Strategies/Tactics in Higher Order Logics, number NASA/CP-2003-212448 in NASA Technical Reports*, pp. 56–68, 2003.

[19] C. W. Brown, "QEPCAD B: a program for computing with semi-algebraic sets using cads," *ACM SIGSAM Bulletin*, vol. 37, no. 4, pp. 97–108, 2003.

[20] L. De Moura and N. Bjørner, "Z3: An efficient smt solver," *Tools and Algorithms for the Construction and Analysis of Systems*, pp. 337–340, 2008.