Article

Towards Optimization of Energy Consumption of Tello Quadrotor with MPC Model Implementation

Rabab Benotsmane [1], József Vásárhelyi [2]

[1] Institute of Automation and Info-Communication; rabab.benotsmane@uni-miskolc.hu

[2] Institute of Automation and Info-Communication; vajo@mazsola.iit.uni-miskolc.hu

Abstract: For a decade, the studies of dynamic control for unmanned aerial vehicles took a large interest, where drones as a useful technology in different areas were always suffering from several issues like instability-high energy consumption of batteries - inaccuracy of tracking targets. Different approaches are proposed for dealing with the nonlinearity issues which present the most important features of this system. This paper describes our focus on the most common control strategies, known as model predictive control MPC, by developing a model based on the sensors embedded in our Tello quadrotor used for indoor purposes. The original controller of Tello quadrotor is supposed to be a slave, where the designed model predictive controller is created in MATLAB and imported to another embedded system, considered as a master; the objective of this model is to track the reference trajectory, almost keeping the stability of the system and ensure the low energy consumption. In the first part, a profound description of the modeling process of a dynamic model for drones is presented, explaining the design of MPC controller with both linear and nonlinear strategies built in MATLAB. In the final part, simulation and results are discussed regarding its behavior and performance, highlighting the MPC model's important role on drones' energy consumption profile.

Keywords: dynamic control; UAV; model predictive control; nonlinear MPC; trajectory tracking; energy consumption

1. Introduction

Drones, abbreviated as UAVs (Unmanned Aerial Vehicles), are unmanned aircraft capable of carrying out a mission autonomously by being self-piloted or semi-autonomous using a remote control [1], [2]. The primary function of these vehicles is to extend human vision beyond the natural horizon to accomplish works at risk or in hostile environments. The military has implemented the first applications of drones for surveillance or reconnaissance missions without the risk of human losses. More recently, civil applications have appeared, such as the prevention of forest fires, the inspection of engineering structures, the energy flow monitoring of motorway traffic, or the collection of meteorological data. However, the use of civilian drones remains limited because non-military drones still need to be fully integrated into civil airspace [2], [3].

The classification of the existing UAVs differs according to the countries and their army [4]. The UAVs can be classified according to size, endurance, flight altitude, function, mass, payload, etc. The most accepted classification divides UAVs according to their size and endurance, where mostly five categories of UAVs exist, from aircraft UAVs with a wingspan of about 30 meters to flapping wings a few millimeters long.

In this paper, our interest mainly focuses on quadrotor UAVs, known as drones with multiple rotors. Multi-rotor UAVs are certainly the best-known and most widespread aeromechanics configuration among autonomous aerial vehicles [5]. These aircraft are usually equipped with four rotors, but there are also some with six or even eight rotors. The mechanical simplicity of this type of vehicle makes it a widely used configuration for the realization of experimental platforms at reduced costs [6]. The operation of multi-rotor drones is quite particular, where the configuration ca be plus or cross configuration [7]. First, to compensate for the torque reaction, the rotation of the rotors' direction is reversed two by two. In addition, the translation and the rotation movements according to the three axes are done carefully by manipulating the rotational speeds of the different rotors. In fact, it is the difference in the lift that determines the inclination of the aircraft around the roll and pitch angles, allowing translational flight, Figure 1 shows the movement states of a quadrotor with cross configuration regarding the X, Y, and Z axes [8].

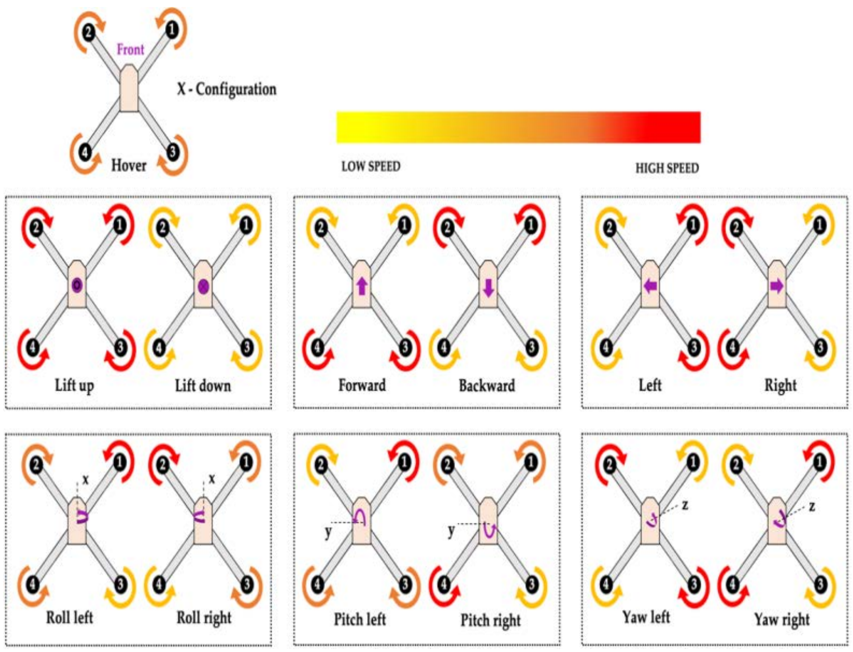

Figure 1. The possible motions of quadrotor with cross configuration

Due to the very good distribution of lift in the horizontal plane, multi-rotor UAVs are particularly suitable for hovering and low-speed flights. On the other hand, this configuration is not recommended for high-speed translational flights nor for driving in windy conditions. Moreover, this rotor configuration is restricted to small-size aircraft because the size of the rotors increases with the increase of the mass of the drone.

The main goal of drones is to achieve a high degree of autonomy to make decisions and be in a stable state, this can be done based on the embedded systems in the body part, which are the measurement system and the controller. The measurement system in the drone is responsible for the collection of information on the state of the drone and its environment, where in the drone, we can find three types of sensors [9], [10]:

- Sensors for measuring attitude, such as Gyroscopes to measure angular position and angular velocity, Accelerometers to measure acceleration, Magnetometers to measure the direction of the magnetic field and scan the area and detect metals in space,
- Sensors for measuring the velocity and position of the drone as an Altimeter, GPS, and Camera,
- Sensors for detecting obstacles around the drone as Ultrasonic sensors and Telemeter laser.

The obtained information from the measurement system is transferred to the control part as digital signals, where the data is processed by the control system. The control system based on a mathematical algorithm generates the control signals allowing the drone to move in an appropriate way. This is based on a mathematical representation of the drone's mechanical body and on the measurement signals from the onboard sensors. Its role is, above all, to guarantee the stability of the device during the autonomous flight phase. Despite a general principle that is relatively simple to understand, the design of such a system is rather complex and requires special attention. Indeed, UAVs, and more particularly rotary wing UAVs such as quadrotors, are under-actuated systems, sensitive to aerodynamic disturbances, and whose dynamics are highly non-linear [11], [12]. Moreover, it presents a significant coupling between the system's state variables and its control inputs. The coupling characterizes the fact that any change in a control input affects not only the variable of interest, but also the others [13].

Based on Equations of Newton Euler or Energetic Lagrange derivative, many scientists suggested how to design the dynamic modeling of drones, where from literature review, we can find the mathematical model for the kinematic and dynamic models, as it appears in [13]–[15]; taking into consideration the behavior of the quadrotor in different flight conditions (stationary, in translation, or in rotation). However, the main problem with these dynamic models is the difficulty to design a "simple" control algorithm due to the complexity of the model which is nonlinear. The first mission of the quadrotor is how to track a position from one point to another point in the 3D environment respectively to the given orientation. Under all the previous cited challenging criteria, this task makes the object deal with the study of control theory, by developing controllers that can success to track the trajectory and avoiding obstacles, all by optimizing the battery life of the drone.

The control strategy for the quadrotor scope represents a big issue in this research field, where many scientists suggested different implementations, some of them were focusing on linear control strategies as PID – LQ – H infinity – Linear MPC, where others preferred to deal with nonlinear control approaches as Backstepping control – Sliding mode control – Nonlinear MPC [16]–[21].

At the energy profile, one of the biggest drawbacks of the quadrotor is presented in the battery life, where usually the fact that time average mission does not exceed 15 min. Since existing controllers are typically designed to maximize the performance of tracking while providing a sufficient margin of stability. However, extracting the maximum performance generates greater consumption and, therefore, a reduction in the duration of battery life. For this reason, several researchers have focused on regulation using the linear quadratic controller since it minimizes, for its quadratic nature, energy consumption. In addition, several strategies can improve fuel efficiency. We can cite as an example:

- Minimization of errors: indeed, this is the classic approach; the less we commit errors in the pursuit of the desired trajectory, the less energy is spent [22];
- Fuzzy logic applied to the altitude control: Indeed, the fact of climbing or descending consumes double and even more because this movement causes the rotation of the four engines simultaneously, so one could think of establishing fuzzy logic between flight height and battery state of charge [23];
- The search for an optimal trajectory: energy consumption can be reduced using or avoiding air currents and thus finding the flight path least costly in terms of energy compared to a direct flight to the destination [23].

This paper presents our focus on developing a customized, powerful embedded control system for Tello EDU quadrotor model [24] quadrotor already has a basic embedded system that fulfills simple commands for take-off and landing motions with low accuracy and stability.

The paper includes the following parts: first, the process to identify the dynamic mathematical model of Tello drone with the actual parameters is presented, this model is purely nonlinear due to the strong coupling between the state variables and the control inputs, but in addition, its dynamics are under-actuated, i.e., the number of information which is 4 is less than the number of outputs which is six that presents the degrees of freedom. In the second part, after getting the dynamic model, we will show the methodology to build a control strategy. Two approaches are taken into consideration in our research: the first assumes that the system is linear, so we apply model predictive control. This approach requires linearizing the dynamic model around an equilibrium point hover position. The second approach assumes that we are dealing with reality and that the system is nonlinear. So, we applied nonlinear model predictive control to track the reference trajectory. A detailed comparison will be presented regarding the motion tracking, stability, and energy profile. We will show the reader how the inputs will lead to the effect of predictive model control on energy consumption. The input variables for the drone present the angular velocities of the four rotors. Then in the last section, the simulation results are discussed, and a comparison is made to understand the potential of using a nonlinear controller instead of a linear one.

2. Materials and Methods

The main aim of this work is to build a customized embedded system for Tello EDU quadrotor. The customized embedded system is based on a System on Chip from Xilinx-AMD and implements a control strategy based on MPC. Figure 2 presents the plan for our main target. To start the process, the research began by developing a suitable model predictive control for Tello EDU quadrotor. This type of drone is helpful for education and research purposes. The control model is based on the dynamic mathematical model of the drone and its fundamental parameters; therefore, this paper presents a methodology for building a suitable control strategy based on the following points:

- the conditions are taken into consideration to identify the dynamic mathematical model.
- parameter identification of Tello EDU quadrotor.
- use of linear and nonlinear model predictive control
- deep comparison to show the effectiveness of a nonlinear controller - how the optimization of energy consumption can be achieved regarding the control part.

The simulation is done using MATLAB Simulink, offering different functions that could help us achieve the research target efficiently, especially since setting parameters and analyzing the results in Simulink is more friendly than using MATLAB Script.

In the following, the paper will highlight the methods used to build the optimal controller, starting by modeling the dynamics of Tello quadrotor, then dealing with the control strategy part by getting familiar with a model predictive controller in general and how we succeed in tuning both control types (linear and nonlinear).

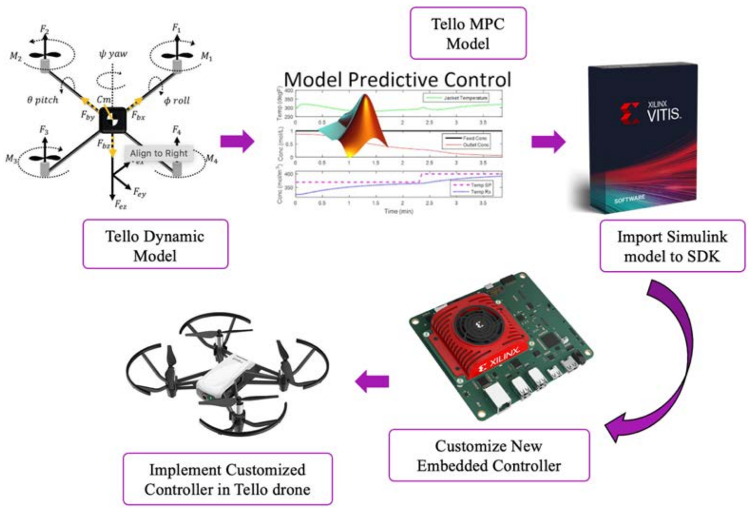
Tello MPC Model
Model Predictive Control
Tello Dynamic Model
Import Simulink model to SDK
Implement Customized Controller in Tello drone
Customize New Embedded Controller
ψ yaw
θ pitch
ϕ roll
XILINX VITIS

Figure 2. The proposed plan for building the customized embedded system for Tello Edu

2.1. *Tello Quad-rotor Model*

Tello EDU quadrotor model [24] is an impressive and programmable valuable drone for educational purposes. The system is dedicated to the students to learn programming languages such as Scratch, Python, and Swift and develop AI functions like a fly-in swarm, upgraded with

SDK 2.0, embedded with DJI's flight control technology, and supports electronic image stabilization.

Figure 3 presents the Tello quadrotor with its specifications. The drone features a vision positioning system and an onboard camera based on an advanced flight controller to hover in a precise place, suitable for flying indoors, in which its maximum time flight is around 13 minutes, and its maximum flight distance is 100 meters. These specifications are critical to see the advantages and the drawbacks that they have. For example, when we say the drone is for indoor activities, this characteristic will neglect the wind and gyroscopic effects as a disturbance in the system. Also, it is a motivation for us to optimize the energy profile of this drone where the time flight can be much longer in case of using optimal control strategies.

From Tello drone, our interest, as we mentioned earlier, is to develop a new predictive control model using its basic sensors identified as a complete system: the vision positioning system helps the aircraft to maintain its current position, where the main components are the camera and the 3D infrared module located on the underside of the plane.

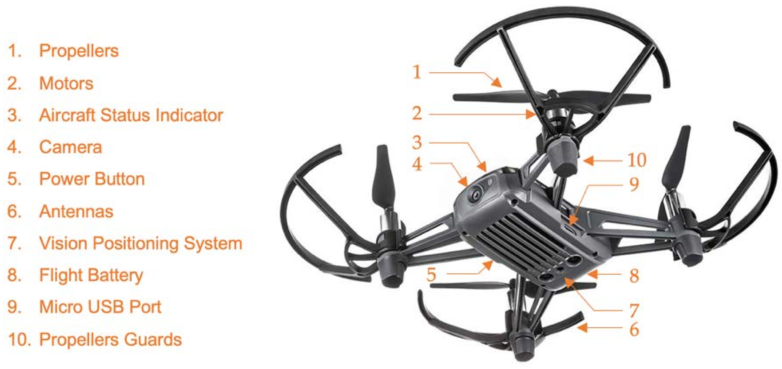

Figure 3. Tello drones basic components

2.2. *Quadroto Mathematical Model*

Designing a suitable controller for an appropriate drone requires first modeling the mathematical dynamics of the plant. This helps to understand how the drone moves and what forces and torques are applied. How many motions do we have? Which velocities are needed to execute different flight modes? Therefore, one can identify the inputs, outputs, the state's variables, and the disturbances of the central system.

Figure 4 describes the dynamic structure of Tello drone in the body and inertial frames, where the corresponding angular velocities, torques, and forces created by the four rotors (numbered from 1 to 4) are presented. The drone's movement is generated by differential thrust forces between F2 and F4 that generate the rotation around the X axis, known as roll motion. On the other hand, differential thrust forces between F1 and F3 will generate the rotation around the Y axis, known as pitch motion. In contrast, the rotation around the Z axis, the yaw motion, is caused by differential torques between clockwise and anticlockwise rotors, i.e., $(\tau_1 - \tau_2 + \tau_3 - \tau_4)$.

The modeling of UAVs is a delicate task since the system's dynamics are strongly nonlinear and fully coupled. Many literature reviews explained the methodology for modeling such systems (references?). This paper presents the exact strategy to get both linear and nonlinear dynamic models based on the Tello drone's main parameters. The calculation of the mathematical model is based on some simplifications; here are the different working hypotheses [25]:

- The structure of the quadrotor is assumed rigid and symmetrical, which induces that the matrix of inertia will be assumed diagonal.
- The propellers are assumed rigid in order to be able to neglect the effect of their deformation during rotation.
- The center of mass and the origin of the frame linked to the structure coincide.
- The lift and drag forces are proportional to the squares of the rotors' rotation speed, which is a very close approximation of the aerodynamic behavior.
- To evaluate the quadrotor's mathematical model, we use two frames, a fixed inertial frame to the earth $\mathrm{R_f}$ and the second mobile frame fixed in the quadrotor $\mathrm{R_m}$. The transformation matrix gives the passage between the body frame and the inertial frame T, which contains the orientation and the mobile position reference relative to the fixed reference.

Figure 4 shows the axis convention taken for Tello drone to model the mathematical dynamic, where the forces, inertial moments, and angular velocities are described.

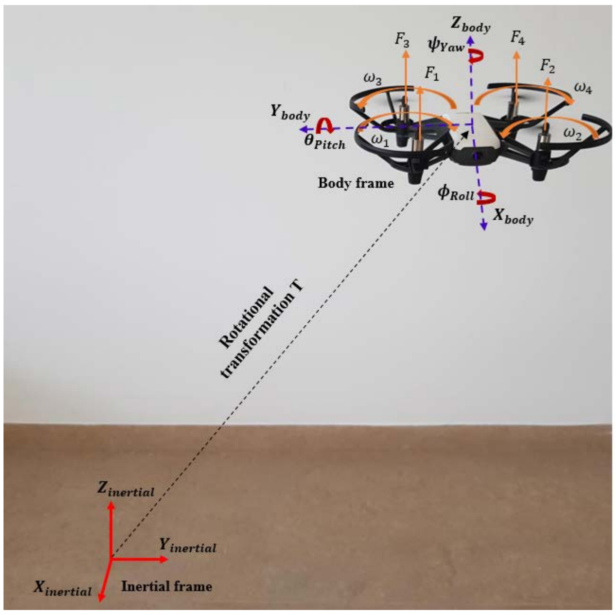

Figure 4. Modeling frame assignments for Tello quadrotor [25]

The quadrotor system is a light structural flight vehicle. Therefore, the gyroscopic effects resulting from the rotation of the rigid body and the four propellers should be included in the dynamic model. However, the dynamic model of the system is obtained under the assumption that the vehicle is a rigid body in the spatial domain with different forces and torques. The mathematical model presented here is based on Newton and Euler equations for the rigid body 3D motion [25].

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 November 2022 doi:10.20944/preprints202211.0181.v1

Let us describe $[x\,y\,z\,\phi\,\theta\,\psi]^T$ the vector containing the linear and angular position of the quadrotor in the inertial frame F_i, and $[u\,v\,w\,p\,q\,r]^T$ the vector containing the linear and angular velocities in the body frame F_b. From 3D body dynamics, it follows that the two reference frames are linked by the following relations:

$$v = \begin{bmatrix} v_x^i \\ v_y^i \\ v_z^i \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = R \cdot v_b = R \cdot \begin{bmatrix} u \\ v \\ w \end{bmatrix} \quad (1)$$

$$\Omega = \begin{bmatrix} \Omega_x^b \\ \Omega_y^b \\ \Omega_z^b \end{bmatrix} = \begin{bmatrix} p \\ q \\ r \end{bmatrix} = T \cdot \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \quad (2)$$

where v is the linear velocity and Ω the rotation velocity, R is the rotation matrix describes the rotation from the body reference system to the inertial reference, T is the matrix for angular transformation, where to obtain $[\dot{\phi} \quad \dot{\theta} \quad \dot{\psi}]^T$, we need the inverse of matrix T.

$$R = R_b^i = \begin{bmatrix} C\psi C\theta & C\psi S\psi S\theta - C\phi S\psi & C\phi S\theta C\psi + S\phi S\psi \\ C\theta S\psi & S\phi S\theta S\psi + C\phi C\psi & C\phi S\theta S\psi - S\phi C\psi \\ -S\theta & S\phi C\theta & C\phi C\theta \end{bmatrix} \quad (3)$$

$$T = \begin{bmatrix} 1 & 0 & -S\theta \\ 0 & C\phi & C\theta S\phi \\ 0 & -S\phi & C\phi C\theta \end{bmatrix},\ T^{-1} = \begin{bmatrix} 1 & S\theta T\theta & C\phi T\theta \\ 0 & C\theta & -S\theta \\ 0 & \frac{S\phi}{C\theta} & \frac{C\phi}{C\theta} \end{bmatrix} \quad (4)$$

where $C\theta$ and $S\phi$ and $T\phi$ represent cosinus and sinus and tangant. From the two relations, we can get the kinematic model for the quadrotor as follow:

$$\begin{cases} \dot{x} = u[C\psi C\theta] + v[C\psi S\psi S\theta - C\phi S\psi] + w[C\phi S\theta C\psi + S\phi S\psi] \\ \dot{y} = u[C\theta S\psi] + v[S\phi S\theta S\psi + C\phi C\psi] + w[C\phi S\theta S\psi - S\phi C\psi] \\ \dot{z} = u[-S\theta] + v[S\phi C\theta] + w[C\phi C\theta] \\ \dot{\phi} = p + q[S\phi T\theta] + r[C\phi T\theta] \\ \dot{\theta} = q[C\phi] + r[-S\phi] \\ \dot{\psi} = q\left[\frac{S\phi}{C\theta}\right] + r\left[\frac{C\phi}{C\theta}\right] \end{cases} \quad (5)$$

The vector $[u\,v\,w\,p\,q\,r]^T$ is obtained using IMU system, with the data obtained from the accelerometer and gyroscop sensors. In order to get the dynamic model, let us call all the physical effects, forces and the inertia moments applied on the system:

- The weight force of the quadrotor given by:
- $P = m \cdot g$ (6)
- where m is the total mass of the rigid body and g is the gravity.
- The thrust forces created by the four rotors.
- These forces are perpendicular to the plane of the propellers, and are proportional to the square of the rotational speed of the motors, the thrust forces are given by:
- $F_i = b \cdot \omega_i^2$ (7)

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 November 2022 doi:10.20944/preprints202211.0181.v1

- where i = 1 … 4; b is the coefficient of lift it depends on the shape and the number of blades and the air density; ω is the angular velocity of the engine.
- The drag force is the coupling between the pressure force and the viscous frictional force.

In this case, two drag forces are acting on the system, the drag in the propellers and the drag along the axes (x, y, z).

The drag in the propellers acts on the blades; it is proportional to the density of the air, to the shape of the edges, and to the square of the rotation velocity of the propeller, and given by the following relation:

$$F_{da} = d \cdot \omega^2 \tag{8}$$

where: d is the drag coefficient; it depends on the manufacture of the propeller.

The drag along the axes (x, y, z) is due to the movement of the body of the quadrotor:

$$F_{dl} = K_{ft} \cdot \mathrm{v} \tag{9}$$

with: K_{ft} is the translational drag coefficient, and v is the linear velocity. Regarding the inertial moments applied to the system, we have the following:

- The inertial moments due to the thrust force: The rotation around the x and y axes is due to the inertial moment created by the difference between the thrust forces of rotors 2 and 4 and 1 and 3, respectively:

$$\begin{cases} M_x = l(F_4 - F_2) = lb(\omega_4^2 - \omega_2^2) \\ M_y = l(F_3 - F_1) = lb(\omega_3^2 - \omega_1^2) \end{cases} \tag{10}$$

where: l is the length of the arm between the rotor and the center of gravity of the quadrotor.

- The inertial moment due to the drag forces: The rotation around the z axis is due to a reactive torque by the drag torques in each propeller:

$$M_z = d(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2), \tag{11}$$

- The inertial moment resulting from aerodynamic friction is given by:

$$M_a = K_{fa} \cdot \Omega^2 \tag{12}$$

where: K_{fa} the coefficient of aerodynamic friction angular velocity.

- Gyroscopic effect This effect is defined as the difficulty of modifying the position or the orientation of the plane of rotation of a rotating mass. In our case there are two gyroscopic moments. The gyroscopic moment of the helices is given by the expression:

$$M_{gh} = \sum_{i=1}^{4} \Omega \wedge J_r [0 \quad 0 \quad (-1)^{i+1} \cdot \omega_i]^T \tag{13}$$

The gyroscopic moment due to the movements of the quadrotor is written:

$$M_{gm} = \Omega \wedge J\Omega \tag{14}$$

where: J_r is the moment of inertia of the rotor, J is the moment of inertia of the system.

Other aerodynamic effects, which can disturb the movement of the quadrotor, are not taking into consideration in this article, as their effects at low speed can be neglected as: blade flapping effect- air friction-ground effect.

By using Newton – Euler equations, the dynamics of a rigid body under external forces and moments in the fb can be formulated as follows:

- Dynamic equation of translation motion

$$\boldsymbol{m\dot{v}_{inertial}} = \sum \boldsymbol{F_{external}}, \tag{15}$$

where m is the masse of the rigid body and $v_{inertial}$ and $F_{external}$ are the linear velocity and the forces projected in the inertial frame respectively. By using the rotation matrix to transform forces from the rigid body frame to the inertial body frame, we get:

$$\boldsymbol{m\ddot{\xi}} = \boldsymbol{F_t} + \boldsymbol{F_{dl}} + \boldsymbol{F_g}, \tag{16}$$

$$F_t = R\begin{bmatrix}0 & 0 & \sum_{i=1}^{4} F_i\end{bmatrix}^T; \quad F_g = \begin{bmatrix}0 & 0 & -mg\end{bmatrix}^T \tag{17}$$

$$F_{dl} = \begin{bmatrix} -K_{ftx} & 0 & 0 \\ 0 & -K_{fty} & 0 \\ 0 & 0 & -K_{ftz} \end{bmatrix} \dot{\xi} \quad ; \text{where } \xi = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{18}$$

By replacing all the terms that we have presented, we get:

$$m\begin{bmatrix}\ddot{x} \\ \ddot{y} \\ \ddot{z}\end{bmatrix} = \begin{bmatrix} C\phi S\theta C\psi + S\phi S\psi \\ C\phi S\theta S\psi - S\phi C\psi \\ C\phi C\theta \end{bmatrix} \sum_{i=1}^{4} F_i - \begin{bmatrix} K_{ftx}\dot{x} \\ K_{ftx}\dot{y} \\ K_{ftx}\dot{z} \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}, \tag{19}$$

$$\begin{cases} \ddot{x} = \dfrac{1}{m}(C\phi S\theta C\psi + S\phi S\psi)\left(\displaystyle\sum_{i=1}^{4} F_i\right) - \dfrac{K_{ftx}}{m}\dot{x} \\ \ddot{y} = \dfrac{1}{m}(C\phi S\theta S\psi - S\phi C\psi)\left(\displaystyle\sum_{i=1}^{4} F_i\right) - \dfrac{K_{fty}}{m}\dot{y} \\ \ddot{z} = \dfrac{1}{m}(C\phi C\theta)\left(\displaystyle\sum_{i=1}^{4} F_i\right) - \dfrac{K_{ftz}}{m}\dot{z} - g \end{cases} \tag{20}$$

- Dynamic equation of rotation motion

By application of Euler's rotation equation, we have:

$$\mathbf{J\dot{\Omega}} + \boldsymbol{\Omega} \times \boldsymbol{J\Omega} = \sum \boldsymbol{M_{external}} \tag{21}$$

where Ω in this equation represent the rotation velocity presented in the inertial axes, J is the inertial tensor of the symmetric rigid body around its center of mass. $M_{external}$ represent the sum of moments acting on the vehicle,

$$J = \begin{bmatrix} J_{xx} & 0 & 0 \\ 0 & J_{yy} & 0 \\ 0 & 0 & J_{zz} \end{bmatrix} \tag{22}$$

where J_{xx}, J_{yy}, J_{zz} are calculated according to the mass distribution, by replacing the previous inertial moments, we get:

$$\mathbf{J\dot{\Omega}} = -\mathbf{\Omega} \times \boldsymbol{J\Omega} - \boldsymbol{M_a} - \boldsymbol{M_{gh}} + \boldsymbol{M_f}, \tag{23}$$

$$\text{where: } M_f = \begin{bmatrix} M_x = lb(\omega_4^2 - \omega_2^2) \\ M_y = lb(\omega_3^2 - \omega_1^2) \\ M_z = d(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{bmatrix}; M_a = \begin{bmatrix} K_{fax}\dot{\phi}^2 \\ K_{fay}\dot{\theta}^2 \\ K_{faz}\dot{\psi}^2 \end{bmatrix} \tag{24}$$

By replacing all the terms in equation, we get:

$$\begin{bmatrix} J_{xx} & 0 & 0 \\ 0 & J_{yy} & 0 \\ 0 & 0 & J_{zz} \end{bmatrix}\begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = -\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \wedge \left(\begin{bmatrix} J_{xx} & 0 & 0 \\ 0 & J_{yy} & 0 \\ 0 & 0 & J_{zz} \end{bmatrix}\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}\right) - \begin{bmatrix} -J_r\overline{\Omega}_r\dot{\theta} \\ J_r\overline{\Omega}_r\dot{\phi} \\ 0 \end{bmatrix} - \begin{bmatrix} K_{fax}\dot{\phi}^2 \\ K_{fay}\dot{\theta}^2 \\ K_{faz}\dot{\psi}^2 \end{bmatrix} + \begin{bmatrix} lb(\omega_4^2 - \omega_2^2) \\ lb(\omega_3^2 - \omega_1^2) \\ d(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{bmatrix} \tag{25}$$

Such that:

$$\begin{cases} \ddot{\phi} = \dfrac{-(J_{zz} - J_{yy})\dot{\theta}\dot{\psi} - J_r\overline{\Omega}_r\dot{\theta} - K_{fax}\dot{\phi}^2 + lb(\omega_4^2 - \omega_2^2)}{J_{xx}} \\ \ddot{\theta} = \dfrac{-(J_{zz} - J_{xx})\dot{\phi}\dot{\psi} - J_r\overline{\Omega}_r\dot{\phi} - K_{fay}\dot{\theta}^2 + lb(\omega_3^2 - \omega_1^2)}{J_{yy}} \\ \ddot{\psi} = \dfrac{-(J_{yy} - J_{xx})\dot{\phi}\dot{\theta} - K_{faz}\dot{\psi}^2 + d(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2)}{J_{zz}} \end{cases} \tag{26}$$

where: $\overline{\Omega}_r = \omega_1 - \omega_2 + \omega_3 - \omega_4$

- The matrix that relies between forces/ inertial moments and angular velocities of four motors:

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} F_t \\ M_x \\ M_y \\ M_z \end{bmatrix} = \begin{bmatrix} b & b & b & b \\ 0 & -lb & 0 & lb \\ -lb & 0 & lb & 0 \\ d & -d & d & -d \end{bmatrix}\begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix}, \tag{27}$$

The relation presented in this matrix is very useful for building controllers, were u_1, u_2, u_3, u_4 are the four inputs parameters that control the system. In our case, Tello EDU quadrotor is dedicated to the indoor application; therefore, in the following, the linear drag force F_{dl} and the inertial moment of the gyroscopic effect M_{gh}, are considered as external disturbances, which are totally neglected in order to simplify the model.

To summarize the previous work, the model plant is based on 4 inputs U and 6 outputs Y expressed as follows:

$$\text{Inputs: } \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} F_t \\ M_x \\ M_y \\ M_z \end{bmatrix} \text{ Outputs: } [x\ y\ z\ \phi\ \theta\ \psi]^T, \tag{28}$$

The model is simplified as follow:

$$\begin{cases} \ddot{x} = \frac{1}{m}(C\phi S\theta C\psi + S\phi S\psi)\left(\sum_{i=1}^{4} F_i\right) \\ \ddot{y} = \frac{1}{m}(C\phi S\theta S\psi - S\phi C\psi)\left(\sum_{i=1}^{4} F_i\right), \\ \ddot{z} = \frac{1}{m}(C\phi C\theta)\left(\sum_{i=1}^{4} F_i\right) - g \end{cases} \tag{29}$$

$$\begin{cases} \ddot{\phi} = \dfrac{-(J_{zz} - J_{yy})\dot{\theta}\dot{\psi} + lb(\omega_4^2 - \omega_2^2)}{J_{xx}} \\ \ddot{\theta} = \dfrac{-(J_{zz} - J_{xx})\dot{\phi}\dot{\psi} + lb(\omega_3^2 - \omega_1^2)}{J_{yy}} \\ \ddot{\psi} = \dfrac{-(J_{yy} - J_{xx})\dot{\phi}\dot{\theta} + d(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2)}{J_{zz}} \end{cases} \tag{30}$$

2.2.1. Identification parameters of Tello EDU drone

The mathematical dynamic model for the quadrotor taking into consideration the six outputs $[x\ y\ z\ \phi\ \theta\ \psi]^T$ is standard, the main change is reflected regarding the neglected disturbances and the parameter of every model, in our case, the main parameters that we had to identify for the dynamic model are presented in Table 1:

Table 1. Parameters of Tello quadrotor

Parameters	Values
Mass	0.8 [Kg]
J_{xx}	0.0097
J_{yy}	0.0097
J_{zz}	0.017
b	1
d	0.08
l	0.06 [m]
g	9.81

2.3. Control Strategy of Quadrotor System

The controller system is a mathematical algorithm that processes how the quadrotor should act to maintain the measurement equal to the set point. Despite its general principle being relatively simple, the design of such a system requires particular attention, given the complex and under-actuated dynamics of the quadrotor [5]. The literature on this subject is vast and very varied and can lead to a better understanding of the performance of the proposed controllers [26]. This part is devoted to some methods of linear and non-linear controls applied to the quadrotor.

- Linear Control Strategy: Linear control techniques are the first used for UAV flight control. Linear controllers are applied to a linearized model of the quadrotor; the linearization is made around an equilibrium point using Taylor series expansion and the Jacobean method, which will be presented later; among the command's laws known in this field [27]–[30]
- Nonlinear Control Strategy: Nonlinear controllers perform better because they are applied to the accurate quadrotor model, which presents all the critical nonlinearities of the dynamics. The follow-

ing presents the most used method in UAV flight control. Usually, the application of a nonlinear controller can achieve higher performance with a nonlinear model of the UAV plant, but the main disadvantage is that it needs higher power computation [20], [31]–[33].

Table 2 summarizes the most linear and nonlinear control strategies used in UAV flight control in the literature review. In our case, we are using MPC strategy, where two scenarios can be taken:

- 1- the first scenario assumes that the dynamic model is linear, so MPC is linear also; this is very familiar in science, where the dynamic model should be linearized around a defined point called the equilibrium point.
- 2- the second scenario: suggests taking the nonlinear model as it is and, according to the constraints, then building an efficient nonlinear MPC. The second method presents different complications and requires much more computation power.

Table 2. Parameters of Tello quadrotor

Controller type	**Linearity type**	**Stability and time response**	**Computation time**
PID Controller	Linear	Low stability in higher disturbances, acceptable response time.	Low
Linear Quadratic Regulator	Linear	Good stability, with slow response time.	Very low
H-infinity Optimal Control	Linear	High stability, medium response time.	Medium
Backstepping Control	Nonlinear	Not robust when the uncertainties are present in the model of the plant	Medium
Sliding mode Control	Nonlinear	Robust even with the presence of uncertainties and disturbances.	High
Adaptive Control	Nonlinear	Good stability with the presence of uncertainties.	High
Model Predictive Control (MPC)	It can be both	Stability is not guaranteed, high optimization for different constraints.	High

In this paper both extensions are presented in detail, to show the reader the main differences and results obtained from the simulation, especially at the energy profile level.

The obtained dynamic model shown above is purely nonlinear, to go forward with the control field, first, we identify the states of the system as follow, where we can have plenty suggestions. In This paper we will go forward with this notation:

$$X = [x \quad y \quad z \quad \phi \quad \theta \quad \psi \quad \dot{x} \quad \dot{y} \quad \dot{z} \quad \dot{\phi} \quad \dot{\theta} \quad \dot{\psi}]^T, \tag{31}$$

$$\dot{\vec{X}} = \hat{f}(\vec{X}, \vec{u}) \tag{32}$$

where $\hat{f}(\vec{X})$ is the state variables function, and $\hat{f}(\vec{u})$ is the input control variables function.

2.3.1. Model Predictive Control Concept

Model predictive control MPC is an advanced automatic control technique useful mostly for industrial applications. MPC concept is based on using the linear or nonlinear dynamic model of the plant inside the controller in real-time, this is to anticipate the future behavior of the system. The main particularity of MPC is that the output calculation must be solved online. The optimization process is based on the inputs/outputs of the system that minimize a cost function to predict the future. The prediction is executed from the internal model of the controller over a finite time window called the prediction horizon. The solution of the optimization problem is presented as a control vector. Using updated system data, the controller can solve the problem again on the next interval period, therefore MPC is called also moving horizon or receding horizon optimal control.

An MPC solves an optimization problem, specifically a quadratic program (QP), it consists of the following three components: a state dynamics model; a cost function, which is the objective that we would like to achieve; and constraints, which are presented as boundaries to limit the future behavior for the inputs and outputs, where can set soft or hard constraints.

MPC operates as follows: using the open-loop plant model and a chosen prediction horizon N, calculate the set of future states $x_{k+1}, \dots \dots x_{k+N}$ and control inputs $u_k, \cdots, u_{k+N-1}$ that minimizes a cost function subject to state and input constraints. Then, the first element u_k in the set of calculated inputs is used as the input to the plant, and the sequence of calculations is repeated at the next sampling instant.

MPC controller has shown its efficiency in the application with short sampling periods where in the field of numerical controls has given good results in terms of speed and accuracy [34]. Figure 5 shows the internal closed loop of MPC with the plant, where Figure 6 shows the internal behavior concept of MPC.

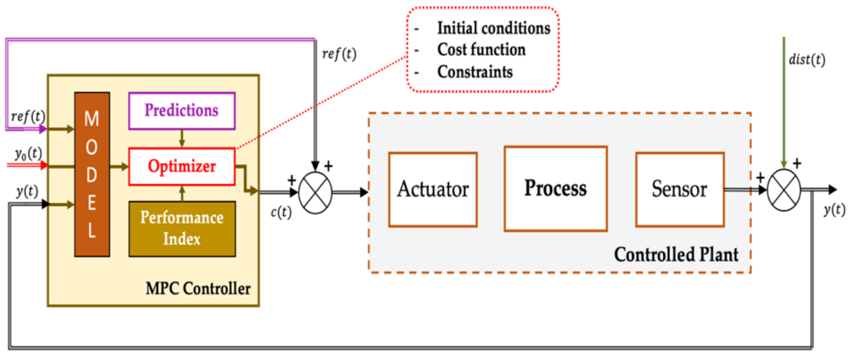

Figure 5. MPC closed control loop

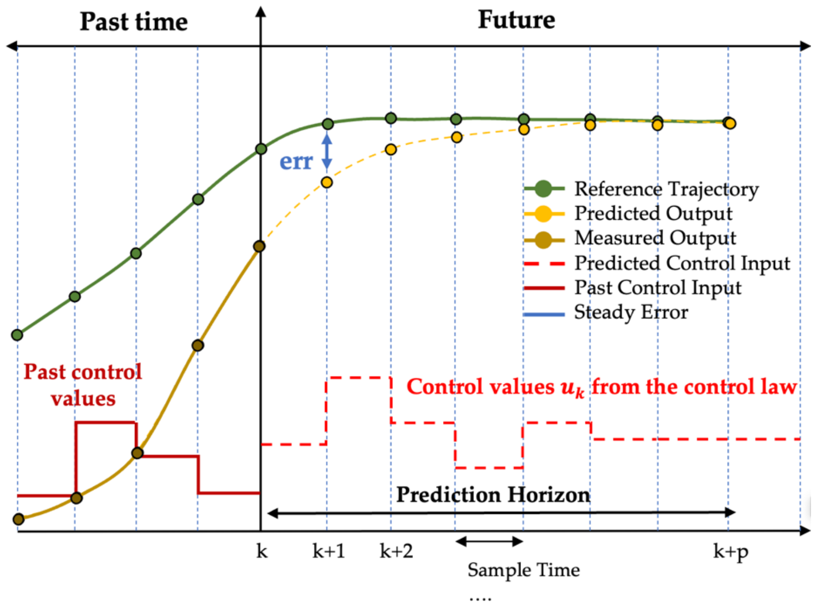

Figure 6. Concept of Model Predictive Control

The standard cost function denotes Θ usually is the sum of four terms, each focusing on a particular aspect of controller performance, as follows:

Every term in Θ represents an objective sub-function:

$$\Theta(z_k) = \Theta_y(z_k) + \Theta_u(z_k) + \Theta_{\Delta u}(z_k) + \Theta_\varepsilon(z_k), \quad (33)$$

where:

- $\Theta_y(z_k)$: Cost function for Output Reference Tracking
- $\Theta_u(z_k)$: Cost function for Manipulated variable tracking
- $\Theta_{\Delta u}(z_k)$: Cost function for manipulated variable move suppression
- $\Theta_\varepsilon(z_k)$: Cost function for constraint violation
- z_k: The quadratic program

In our case we are interested in minimizing the following two terms $\Theta_y(z_k), \Theta_u(z_k)$, where we would like to track the reference output, so the error between the reference and the output signal should be minimized, where the second term is minimized in order to the controller keep selected manipulated variables (MVs) at or near specified target values, in general the problem statement is written as follow:

$$\min_{u_t,\dots,u_{t+N-1}} \{\sum_{k=0}^{N-1} \|r(t) - y_{t+k}\|^2 + \rho\|u_{t+k} - u_r(t)\|^2\} \quad (34)$$

Subject to:

- $x_{t+k+1} = f(x_{t+k}, u_{t+k})$ presents the dynamic of the system.
- $y_{t+k} = g(x_{t+k}, u_{t+k})$ presents the outputs of the system
- $u_{min} \le u_{t+k} \le u_{max}$
- $y_{min} \le y_{t+k} \le y_{max}$
- $x_t = x(t), k = 0, \dots, N-1$ presents state variables

In the following, we will present the design of linear and Nonlinear MPC using MATLAB. The software provides some facilities regarding

MPC controller, where the optimization problem is solved using a quadratic program (QP) – at each control interval. The solution determines the manipulated variables (MVs) to be used in the plant until the next control interval. The definition of our cost function in the software is written as follow:

$$\Theta(z_k) = \Theta_y(z_k) + \Theta_u(z_k), \tag{35}$$

$$\Theta(z_k) = \sum_{j=1}^{n_y} \sum_{i=1}^{p} \left[\left\{ \frac{w_{i,j}^{y}}{s_j^{y}} \left[r_j(k+i|k) - y_j(k+i|k) \right] \right\}^2 + \left\{ \frac{w_{i,j}^{y}}{s_j^{y}} \left[u_j(k+i|k) - u_{j,target}(k+i|k) \right] \right\}^2 \right] \tag{36}$$

where: k is the current control interval; p is the Prediction horizon (number of intervals); n_y: Number of plant output variables. z_k: Quadratic program decision given by:

$$z_k{}^T = [u(k|k)^T \ldots\ldots u(k+p-1|k)^T \quad \varepsilon_k] \tag{37}$$

$y_j(k+i|k)$: Predicted value of jth plant output at ith prediction horizon step, in engineering units. $r_j(k+i|k)$: Reference value for jth plant output at ith prediction horizon step, in engineering units; s_j^y: Scale factor for jth plant output, in engineering units; $w_{i,j}^y$: Tuning weight for jth plant output at ith prediction horizon step (dimensionless).

2.3.2. Linear Model Predictive Control

Linear MPC is the traditional model which the basic of its cost function is a quadratic function, the model traits linear dynamic systems, and does not require higher computation to solve the optimization problem. The basic step to continue with linear MPC is to linearize the dynamic model of the quadrotor, so the methodology is presented as follow:

- Linearization of the quadrotor system

Our aim from this part is to generate a linear state space representation from a nonlinear model written in eq (29) and eq (30) (The nonlinear model is linearized around an equilibrium point X_e " hover position ", the linearization is performed on a simplified model, where small oscillations are considered as: sin(angle) = angle and cos(angle) = 1, and using Taylor series expansion the new state space representation after the linearization is written in the following form:

$$\begin{cases} \dot{\vec{X}} = A\vec{X}' + B\vec{u}' \\ \vec{y} = C\vec{X}' + D\vec{u}' \end{cases} \quad \text{where} \quad \begin{matrix} \vec{X}' = \vec{X} - \vec{X}_e \\ \vec{u}' = \vec{u} - \vec{u}_e \end{matrix}, \tag{38}$$

where $\vec{X}$ is the 'State vector' of $(n \times 1)$, $\vec{u}$ is the 'Input (or control) vector'of $(m \times 1)$, $\vec{y}$ is the 'Output vector' of $(p \times 1)$, A is the 'System Matrix' of $(n \times n)$, B is the 'Input Matrix' of $(n \times p)$, C is the 'Output Matrix' of $(q \times n)$, D is the 'Feed forward Matrix' of $(q \times p)$.

In hover position, we have the following state vector and input vector:

$$X_e = [x_e \quad y_e \quad z_e \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]^T, \tag{39}$$

$$u_e = [mg \quad 0 \quad 0 \quad 0]^T \tag{40}$$

After the linearization calculation we get the following four matrices for the state space representation:

$$A = \frac{\partial f(X,u)}{\partial X}\bigg|_{\substack{x=x_e\\u=u_e}} = \begin{bmatrix} 0&0&0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&0&0&0&1\\ 0&0&0&0&g&0&0&0&0&0&0&0\\ 0&0&0&-g&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0 \end{bmatrix}, \tag{41}$$

$$B = \frac{\partial f(X,u)}{\partial u}\bigg|_{\substack{x=x_e\\u=u_e}} = \begin{bmatrix} 0&0&0&0\\ 0&0&0&0\\ 0&0&0&0\\ 0&0&0&0\\ 0&0&0&0\\ 0&0&0&0\\ 0&0&0&0\\ 1/m&1/J_{xx}&0&0\\ 0&0&1/J_{yy}&0\\ 0&0&0&1/J_{zz} \end{bmatrix} \tag{42}$$

The outputs variable that we would like to track, and control are the following: [x y z ϕ θ ψ], therefore the matrix C and D are written as follow:

$$C = \begin{bmatrix} 1&0&0&0&0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0&0&0 \end{bmatrix}, \tag{43}$$

$$D = zero(6,4) \tag{44}$$

If the disturbance is taking into consideration, so the matrix dist is added to the equation of the state space as follow:

$$\dot{\vec{X}} = A\vec{X}' + B\vec{u}' + dist\vec{d}, \tag{45}$$

$$dist = \frac{\partial f(X,u,d)}{\partial d}\bigg|_{\substack{x=x_e\\u=u_e}} \tag{46}$$

where $\vec{d}$ is the vector of the disturbances applied on the quadrotor. In the case presented in this paper, we neglect all the disturbances.

- Design Linear MPC for Tello Quadrotor System

Using MATLAB Simulink, we designed the dynamic system and the linear MPC to track the following outputs variables: [x y z ϕ θ ψ], we started by reconfigure the main inputs and outputs of the controller. Figure 7 shows the necessary inputs and outputs for MPC controller, as inputs we have: - mo: 6 measured outputs that present the state vector [x y z ϕ θ ψ], - md: 0 measured disturbances where we neglected all the disturbances applied on the quadrotor, as outputs: - mv: 4 manipulated variables $[u_1, u_2, u_3, u_4]$, which present the input control variables.

Inside MPC plant model, different recommendations can be taking into consideration, depending on the characteristics of the dynamic system that we have, in our case the constraints are the weights, and the control horizon were setting as presented in the Table 3. The weight refers to the priority of the variables, where the high weight indicates the higher priority to track.

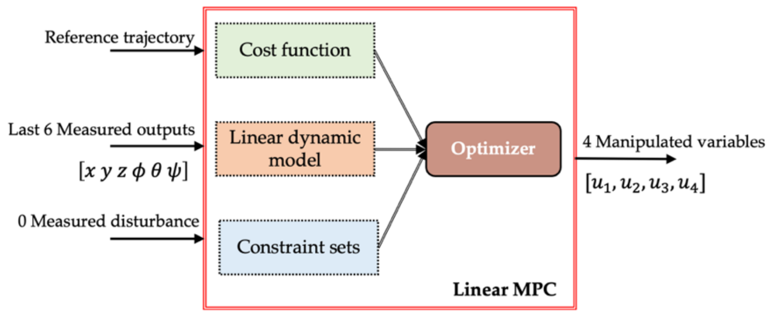

Figure 7. MPC controller inputs - outputs

Table 3. The required parameters of MPC for Tello quadrotor

MPC Parameters	
Sample time (T_s)	0.1 s
Prediction Horizon (P)	20 s
Control Horizon (M)	3 s
MPC Constraints	
Boundaries of u_1	[-10 – 12] N.m
Boundaries of u_2	[-10 – 12] N.m
Boundaries of u_3	[-10 – 12] N.m
Boundaries of u_4	[-10 – 12] N.m
MPC Weights	
Translation motion along x	1
Translation motion along y	1
Translation motion along z	1
Pitch motion along ϕ	0.1
Roll motion along θ	0.1
Yaw motion along ψ	0.1

2.3.3. Nonlinear Model Predictive Control

The principle of nonlinear model predictive control is the same as the linear one, the key differences are:

- the prediction model can be nonlinear and include time-varying parameters.

- the equality and inequality constraints can be nonlinear.
- the scalar cost function to be minimized can be a nonquadratic (linear or nonlinear) function of the decision variables.

When the dynamic model is linear, the global analytical solution of the optimization problem exists and is relatively simple to determine [35]. But when this model is nonlinear, it is not certain that the optimization algorithm converges to the global analytical solution and if a solution is found, there is not sure that it is fast enough. Indeed, the use of a nonlinear model involves solving a nonlinear optimization problem whose complexity is related to the degree of accuracy of the model.

Several numerical techniques were developed to guarantee the convergence of the optimization algorithm towards an approximation of the optimum overall. While some of these methods, such as Eleanor's method, have proven their efficiency, they are however not appropriate for control of drones due to their running time. Among the techniques that can be carried on board vehicles we can cite the gradient method, the quadratic programming method sequential or the particle swarm method. Some techniques are based also on the approximation of certain signals by their Taylor series expansion [36]–[40].

While traditional linear MPC controllers optimize control actions to minimize a quadratic cost function, nonlinear MPC controllers support generic custom cost functions. For example, the cost function can be specified as a combination of linear or nonlinear functions of the system states and inputs. To improve computational efficiency, we can also specify an analytical Jacobian for the custom cost function [41].

Using a custom cost function, the algorithm of NMPC will be able to maximize profitability and minimize energy consumption.

- Design Nonlinear MPC for Tello quadrotor system

As we did in the previous part with linear MPC, the design of NMPC was done in MATLAB Simulink environment, MATLAB Simulink already has a basic architecture block function of such controller. We had to reconfigure the internal structure according to our system and tune the main parameters to achieve high performance. According to the nonlinear dynamic system of Tello quadrotor, the controller should receive different input signals: the reference trajectory – 12 state variables – last 4 manipulated variables that present the four control inputs variables$[u_1', u_2', u_3', u_4']$ as presented in Figure 8, in our case we neglected the gyroscopic effect and the other disturbances applied on the system, whereas outputs we receive the new updated 4 manipulated variables.

The design process for NMPC was as follows: - an object function nlobj has been created in the internal block with setting the necessary parameters and using nlmpc function a nonlinear MPC controller is designed. As drawback of nonlinear controllers is the required huge time simulation, to avoid this issue and guarantee a high performance for the system an algorithm to calculate Jacobian function was inserted to the controller block function, Jacobian function guarantee to have fast and efficient simulation.

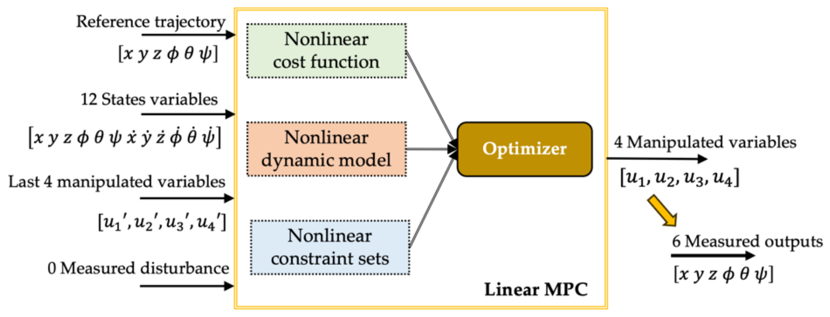
Reference trajectory
$[x\ y\ z\ \phi\ \theta\ \psi]$
12 States variables
$[x\ y\ z\ \phi\ \theta\ \psi\ \dot{x}\ \dot{y}\ \dot{z}\ \dot{\phi}\ \dot{\theta}\ \dot{\psi}]$
Last 4 manipulated variables
$[u_1', u_2', u_3', u_4']$
0 Measured disturbance
Nonlinear cost function
Nonlinear dynamic model
Nonlinear constraint sets
Optimizer
Linear MPC
4 Manipulated variables
$[u_1, u_2, u_3, u_4]$
6 Measured outputs
$[x\ y\ z\ \phi\ \theta\ \psi]$

Figure 8. Nonlinear MPC controller inputs – outputs

Table 4 identify the necessary parameters for the nonlinear MPC that fit Tello quadrotor. The weight parameter in the controller refers to the priority of the variables to track; usually in the quadrotor, we set higher priority for the position variables than the orientation variables, in our case we are looking for the best performance, so we set higher priority for all six variables.

Table 4. The required parameters of Nonlinear MPC for Tello quadrotor

Nonlinear MPC Parameters	
Sample time (T_s)	0.1 s
Prediction Horizon (P)	18 s
Control Horizon (M)	3 s
Nonlinear MPC Constraints	
Boundaries of u_1	[0 – 12] N.m
Boundaries of u_2	[0 – 10] N.m
Boundaries of u_3	[0 – 10] N.m
Boundaries of u_4	[0 – 10] N.m
Nonlinear MPC Weights	
Translation motion along x	1
Translation motion along y	1
Translation motion along z	1
Pitch motion along ϕ	1
Roll motion along θ	1
Yaw motion along ψ	1

3. Simulation Results

In this section, several simulations for three different trajectories are presented to validate the comparison between linear and nonlinear MPC based on the performance analyses of the following inputs and outputs:

- error position tracking
- velocity tracking
- control inputs tracking.

The design of the linear and nonlinear dynamic model of the drone beside the linear and nonlinear MPC models were designed in MATLAB Simulink, where different functions were written in MATLAB Script as: trajectory function, which provides to the control loop the reference desired trajectory; Jacobian function, which guarantees solving the optimization problem and avoiding the singularities.

Three generated trajectories are defined: the first represents a straight curvature motion of the drone, the second trajectory is presented as a circle motion, while the third trajectory is a helical motion, the initial states for both controllers were set as follow: $X_i = [0 \quad 2 \quad 2 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]^T$.

For Linear MPC, the control input bounds were set as presented in Table 3 between minimum value of -10 and maximum value of 12. This choice leads to give the controller more freedom to track the trajectory, where if the minimum value is set to 0. The controller will behave in an aggressive scenario. For Nonlinear MPC the control input boundaries were set between minimum value of 0 and maximum value of 12. Another setting was made regarding the control inputs tracking, where for Nonlinear MPC, the tracking control inputs can be done successfully so the track values where set to [4.9 4.9 4.9 4.9], while for Linear MPC, this option was neglected because it will definitely disturb the controller to achieve the most priority task, which is tracking the position and orientation outputs. From the first attempt and without looking to the simulation results, the recognition of these limitations could identify that Nonlinear MPC is more flexible than Linear MPC in term of setting conditions.

3.1. Straight Curvature Reference Trajectory

Figure 9 presents the simulation results of the straight curvature reference trajectory, the error position tracking, the velocity tracking, and the control inputs tracking for linear and nonlinear MPC. From Figure 9, we observe that regarding error position tracking LMPC performs better regarding the three axis X, Y, and Z, where for orientations angles, the signal is disturbed comparing to NLMPC where the oscillations are minimized. These oscillations describe the instability of the drone. Also, at level of tracking velocity, where both of controllers are behaving nearly the same but always NMPC is able to minimize the peak amplitude of the oscillations. From the energy profile perspective, the comparison of the control inputs tracking leads to identify that both controllers reach the maximum and the minimum boundaries set earlier, to know which controller is using more torque, the mean value of Δu for every execution is calculated, this will be presented in the following table (Table 5).

3.2. Circle Reference Trajectory

Figure 10 presents the simulation results of circle motion as reference trajectory. From this Figure, the differences start to rise clearly, where it is obvious that NLMP is tracking the positions and the velocities in stable behavior, where the oscillations are more presented with LMPC, otherwise NLMP has succeed to minimize them. Also at control inputs tracking level, LMPC is using the maximum and minimum

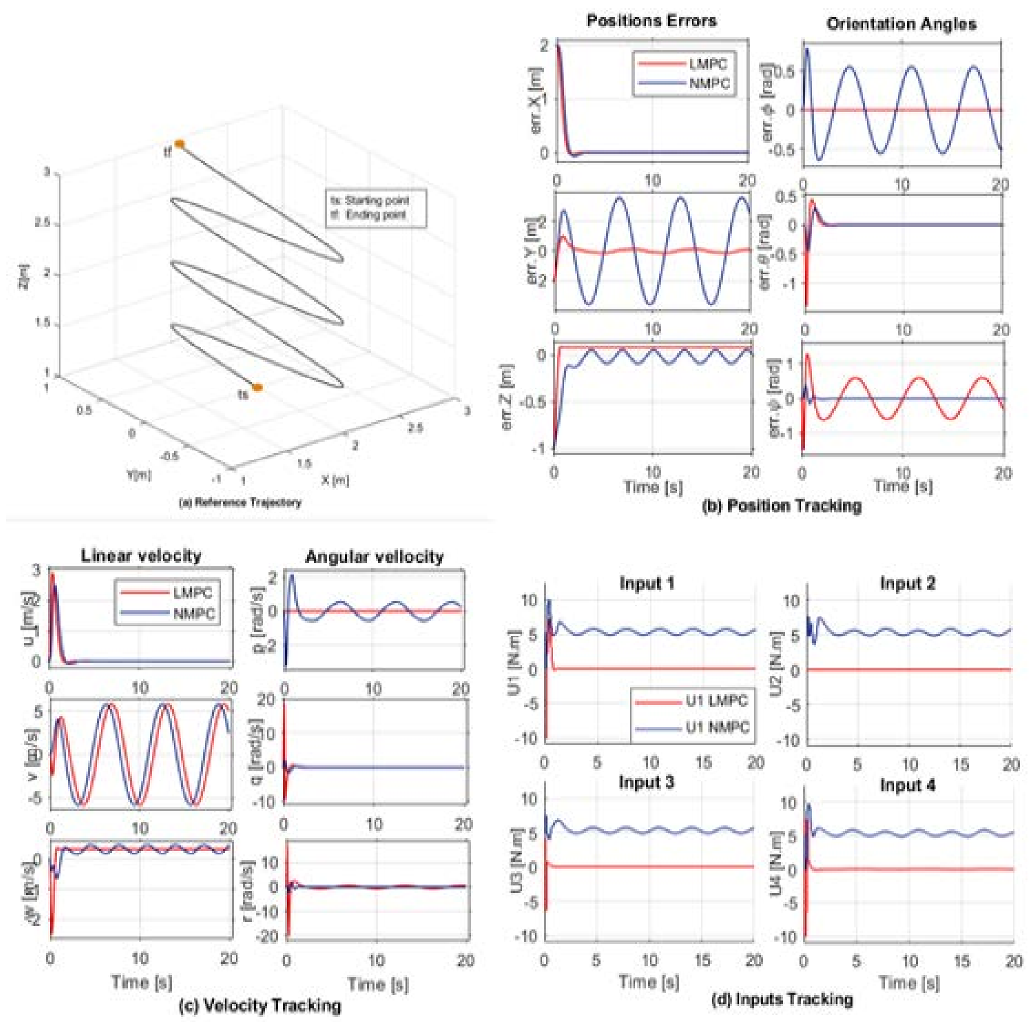

Figure 9. Straight curvature trajectory there is problem in the picture

boundaries so it can reach the minimum value – 10 N.m where NMPC is acting well with a limited boundaries with minimum value 0 N.m. Also NMPC is able to track the set target control inputs [4.9 4.9 4.9 4.9].

3.3. Helical Reference Trajectory

Figure 11 presents the simulation results of helical motion as reference trajectory. This motion describes typically the real life, where the motion is depending on the three axes simultaneously, therefore we observe that NMPC succeed to track the three positions smoothly and with minimum peak amplitude, while in the orientation tracking an overshoot is appeared in the first two seconds, regarding the velocity tracking. NMPC also provide good results comparing to LMPC, where the tracking control inputs is well achieved by NMPC. LMPC is always using all the boundaries to execute the suitable control behavior.

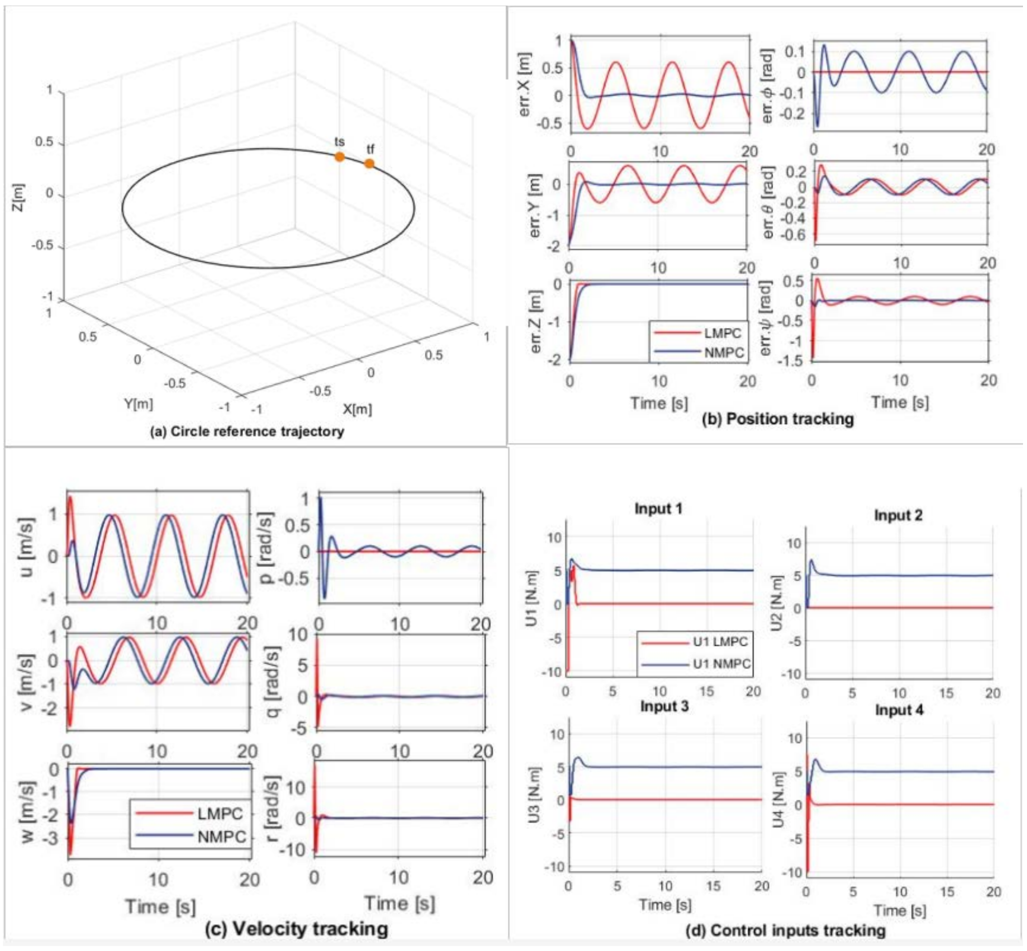

Figure 10. Circle reference trajectory

3.4. Performance measurments Δu *and* $ExTime$

In order to analyze the energy profile of both controllers, a calculation of the mean average value of the control effort Δu and the execution time of the simulation Ex_{Time} was made.

Table 5 shows the results of the two values when every trajectory is executed. These two parameters identify the consumption energy for both controllers, where regarding the execution time, we found that LMPC is very slow comparing to NMPC, meanwhile the average of control effort for LMPC is less than the control effort required by NMPC with average differences. This can be explained by the fact that NMPC has different missions to achieve where it is tracked not only the trajectory but also try to maintain a target value for the control inputs. A solution can be highlighted in the future by optimizing a cost function that minimizes the energy consumption of the drone, by minimizing the applied control efforts directed to the four motors.

The calculation of Δu was by ***mean*** function, where the execution simulation time was calculated by ***timeit*** function.

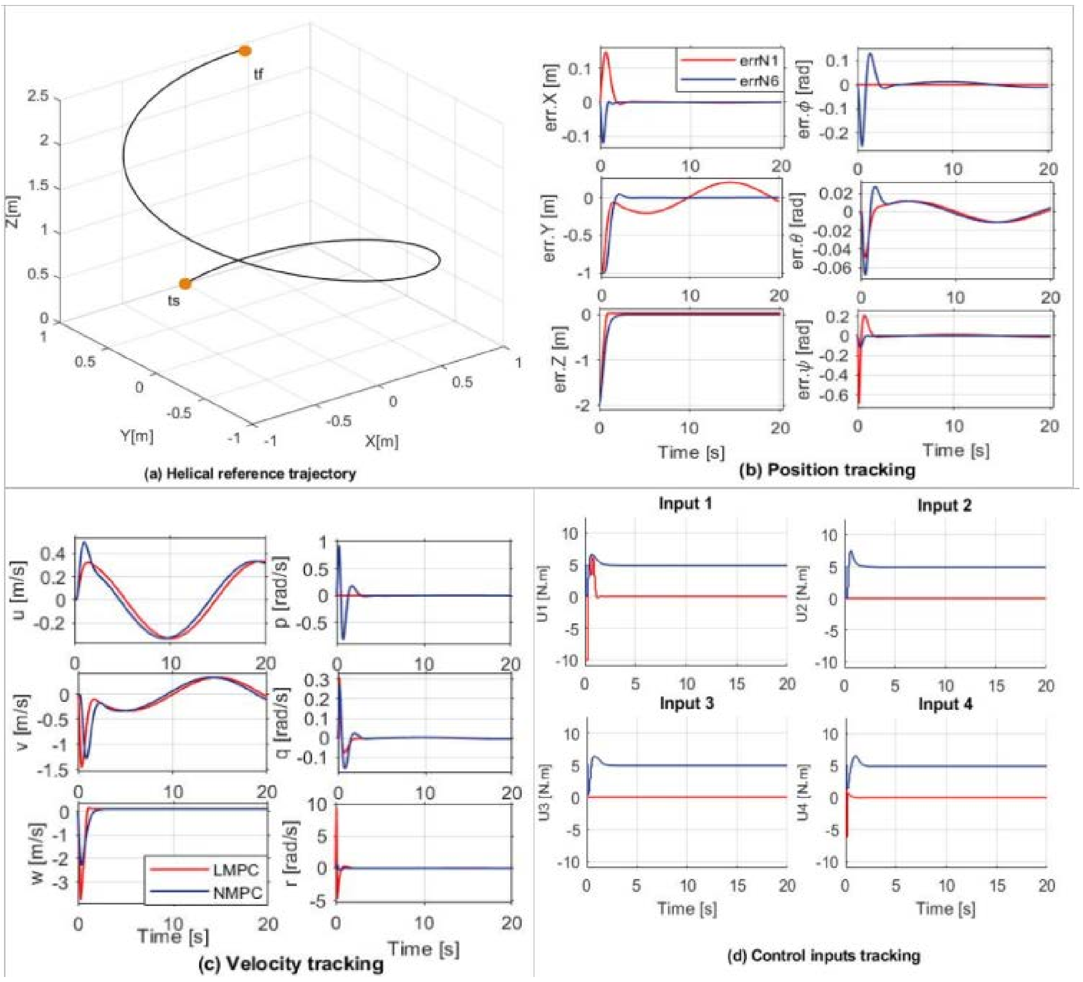

Figure 11. Circle reference trajectory

Table 5. Calculation of the average control effort value and the execution time

Trajectory / Controller	1st Trajectory 1		2nd Trajectory		3rd Trajectory	
Δu[N.m]/Ex Ex_{Time}***[ms]***	Δu	Ex_{Time}	Δu	Ex_{Time}	Δu	Ex_{Time}
LMPC	3.1678	64	2.7054	63	1.4229	62
NLMPC	5.9188	6.8	5.2379	6	4.8955	5.7

4. Summary and Conclusion

This paper presented a methodology to design MPC controller with the purpose to track trajectory of the quadrotor system. The dynamic mathematical modeling part for the quadrotor was deeply described in the first section, where in the second section a theoretical comparison between linear and nonlinear MPC was presented, which conducted us to look after the efficiency of these both control laws especially regarding the energy profile. The design of both controller models was built and configured using MATLAB Script and Simulink, the controllers were simulated under the execution of three different trajectories, to see the resulted behavior. As a conclusion, we sum up that NMPC can provide a very promoting results comparing to LMPC, which is not flexible and rise different limitations in the control drone field.

As future work, our main goal is to build a master controller hardware embedded system for Tello quadrotor, the hardware will be embedded by such a control law, as we identified how NMPC is fast to process the commands, it would be great to increase the potential of it regarding the optimization of energy consumption, which presents a drawback of the quadrotor system, the target would be to add to NMPC a new cost function that minimize the voltage from the control efforts which is directed to the four rotors.

Author Contributions: *"Conceptualization, R.B., V.J.; methodology, R.B., V.J.; formal analysis, R.B., V.J.; invited author by guest editor, V.J.; writing—original draft preparation, R.B., V.J.; writing—review and editing; R.B., V.J."*

Funding: "This research received no external funding"

References

[1] R. K. Barnhart, S. B. Hottman, D. M. Marshall, and E. Shappee, *Introduction to Unmanned Aircraft Systems*, 1st ed. Boca Raton: CRC Press, 2016. doi: 10.1201/B11202.

[2] K. Nonami, F. Kendoul, S. Suzuki, W. Wang, and D. Nakazawa, *Autonomous flying robots: Unmanned aerial vehicles and micro aerial vehicles*. Tokyo: Springer Japan, 2010. doi: 10.1007/978-4-431-53856-1/COVER.

[3] G. D. Padfield, C. A. Kluever, T. M. Young, A. J. Keane, A. Sóbester, and J. P. Scanlan, "Design of Unmanned Aerial Systems," *Design of Unmanned Aerial Systems*, Mar. 2020, doi: 10.1002/9781119508618.

[4] K. Dalamagkidis, *Classification of uavs*. Dordrecht: Springer Netherlands, 2014. doi: 10.1007/978-90-481-9707-1_94/COVER.

[5] M. Idrissi, M. Salami, and F. Annaz, "A Review of Quadrotor Unmanned Aerial Vehicles: Applications, Architectural Design and Control Algorithms," *Journal of Intelligent & Robotic Systems* , vol. 104, no. 2, pp. 1–33, Jan. 2022, doi: 10.1007/S10846-021-01527-7.

[6] F. M. Agugliaro, M. Jacewicz, M. ˙ Zugaj, R. Gł˛, and P. Bibik, "Quadrotor Model for Energy Consumption Analysis," *Energies 2022, Vol. 15, Page 7136*, vol. 15, no. 19, p. 7136, Sep. 2022, doi: 10.3390/EN15197136.

[7] X. Zhang, X. Li, K. Wang, and Y. Lu, "A survey of modelling and identification of quadrotor robot," *Abstract and Applied Analysis*, vol. 2014, 2014, doi: 10.1155/2014/320526.

[8] X. Zhang, X. Li, K. Wang, and Y. Lu, "A survey of modelling and identification of quadrotor robot," *Abstract and Applied Analysis*, vol. 2014, 2014, doi: 10.1155/2014/320526.

[9] E. Balestrieri, P. Daponte, L. de Vito, and F. Lamonaca, "Sensors and Measurements for Unmanned Systems: An Overview," *Sensors 2021, Vol. 21, Page 1518*, vol. 21, no. 4, p. 1518, Feb. 2021, doi: 10.3390/S21041518.

[10] S. Guan, Z. Zhu, and G. Wang, "A Review on UAV-Based Remote Sensing Technologies for Construction and Civil Applications," *Drones 2022, Vol. 6, Page 117*, vol. 6, no. 5, p. 117, May 2022, doi: 10.3390/DRONES6050117.

[11] G. Barbaraci, "Modeling and control of a quadrotor with variable geometry arms," *J Unmanned Veh Syst*, vol. 3, no. 2, pp. 35–57, 2015, doi: 10.1139/JUVS-2014-0012.

[12] Z. Tahir, M. Jamil, S. A. Liaqat, L. Mubarak, W. Tahir, and S. O. Gilani, "State Space System Modeling of a Quad Copter UAV," *Indian J Sci Technol*, vol. 9, no. 27, Jul. 2016, doi: 10.17485/IJST/2016/V9I27/96613.

[13] A. Chovancová, T. Fico, E. Chovanec, and P. Hubinský, "Mathematical Modelling and Parameter Identification of Quadrotor (a survey)," *Procedia Eng*, vol. 96, pp. 172–181, Jan. 2014, doi: 10.1016/J.PROENG.2014.12.139.

[14] M. R. Mokhtari and B. Cherki, "A new robust control for minirotorcraft unmanned aerial vehicles," *ISA Trans*, vol. 56, pp. 86–101, May 2015, doi: 10.1016/J.ISATRA.2014.12.003.

[15] Z. Bodó and B. Lantos, "Modeling and Control of Outdoor Quadrotor UAVs," *SISY 2018 - IEEE 16th International Symposium on Intelligent Systems and Informatics, Proceedings*, pp. 111–116, Nov. 2018, doi: 10.1109/SISY.2018.8524697.

[16] G. J. J. Ducard and M. Allenspach, "Review of designs and flight control techniques of hybrid and convertible VTOL UAVs," *Aerosp Sci Technol*, vol. 118, p. 107035, Nov. 2021, doi: 10.1016/J.AST.2021.107035.

[17] A. N. Chand, M. Kawanishi, and T. Narikiyo, "Adaptive pole placement pitch angle control of a flapping-wing flying robot," *Advanced Robotics*, vol. 30, no. 16, pp. 1039–1049, Aug. 2016, doi: 10.1080/01691864.2016.1196609.

[18] N. Chen, F. Song, G. Li, X. Sun, and C. Ai, "An adaptive sliding mode backstepping control for the mobile manipulator with nonholonomic constraints," *Commun Nonlinear Sci Numer Simul*, vol. 18, no. 10, pp. 2885–2899, Oct. 2013, doi: 10.1016/J.CNSNS.2013.02.002.

[19] A. Bogdanov *et al.*, "State-dependent riccati equation control of a small unmanned helicopter," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2003, doi: 10.2514/6.2003-5672.

[20] G. v. Raffo, M. G. Ortega, and F. R. Rubio, "Robust Nonlinear Control for Path Tracking of a Quad-Rotor Helicopter," *Asian J Control*, vol. 17, no. 1, pp. 142–156, Jan. 2015, doi: 10.1002/ASJC.823.

[21] J. P. How, E. Frazzoli, and G. V. Chowdhary, *Linear Flight Control Techniques for Unmanned Aerial Vehicles*. Springer, Dordrecht, 2015. doi: 10.1007/978-90-481-9707-1_49.

[22] Z. Bodó and B. Lantos, "Integrating Backstepping Control of Outdoor Quadrotor UAVs," *Periodica Polytechnica Electrical Engineering and Computer Science*, vol. 63, no. 2, pp. 122–132, Feb. 2019, doi: 10.3311/PPEE.13321.

[23] F. Gao *et al.*, "Optimal trajectory generation for quadrotor teach-and-repeat," *IEEE Robot Autom Lett*, vol. 4, no. 2, pp. 1493–1500, Apr. 2019, doi: 10.1109/LRA.2019.2895110.

[24] Ryzerobotics User Manual, "User Manual 2018.09," pp. 1–22, 2018.

[25] R. Benotsmane, A. Reda, and J. Vasarhelyi, "Model Predictive Control for Autonomous Quadrotor Trajectory Tracking," *2022 23rd International Carpathian Control Conference, ICCC 2022*, pp. 215–220, 2022, doi: 10.1109/ICCC54292.2022.9805883.

[26] R. Roy, M. Islam, N. Sadman, M. A. P. Mahmud, K. D. Gupta, and M. M. Ahsan, "A Review on Comparative Remarks, Performance Evaluation and Improvement Strategies of Quadrotor Controllers," *Technologies 2021, Vol. 9, Page 37*, vol. 9, no. 2, p. 37, May 2021, doi: 10.3390/TECHNOLOGIES9020037.

[27] G. Farid, M. Hongwei, S. M. Ali, and Q. Liwei, "A REVIEW ON LINEAR AND NONLINEAR CONTROL TECHNIQUES FOR POSITION AND ATTITUDE CONTROL OF A QUADROTOR," *Mechatronic Systems and Control 2017*, vol. 45, no. 1, pp. 43–57, Jan. 2017, doi: 10.2316/JOURNAL.201.2017.1.201-2819.

[28] B. J. Emran and H. Najjaran, "A review of quadrotor: An underactuated mechanical system," *Annu Rev Control*, vol. 46, pp. 165–180, Jan. 2018, doi: 10.1016/J.ARCONTROL.2018.10.009.

[29] O. Araar and N. Aouf, "Full linear control of a quadrotor UAV, LQ vs H∞," *2014 UKACC International Conference on Control, CONTROL 2014 - Proceedings*, pp. 133–138, Oct. 2014, doi: 10.1109/CONTROL.2014.6915128.

[30] A. Koszewnik, "The parrot UAV controlled by PID controllers," *Acta Mechanica et Automatica*, vol. 8, no. 2, pp. 65–69, 2014, doi: 10.2478/ama-2014-0011.

[31] J. B. Rawlings, E. S. Meadows, and A. D. K. R. Muske, "Nonlinear Model Predictive Control: A Tutorial and Survey," *IFAC Proceedings Volumes*, vol. 27, no. 2, pp. 185–197, May 1994, doi: 10.1016/S1474-6670(17)48151-1.

[32] A. Isidori, L. Marconi, and A. Serrani, "Robust nonlinear motion control of a helicopter," *IEEE Trans Automat Contr*, vol. 48, no. 3, pp. 413–426, Mar. 2003, doi: 10.1109/TAC.2003.809147.

[33] S. J. Qin and T. A. Badgwell, "An Overview of Nonlinear Model Predictive Control Applications," *Nonlinear Model Predictive Control*, pp. 369–392, 2000, doi: 10.1007/978-3-0348-8407-5_21.

[34] C. B. E. F. Camacho, *Model Predictive Control*, no. July. London: Springer London, 2000. doi: 10.1007/978-0-85729-398-5.

[35] P. Ru and K. Subbarao, "Nonlinear Model Predictive Control for Unmanned Aerial Vehicles," *Aerospace 2017, Vol. 4, Page 31*, vol. 4, no. 2, p. 31, Jun. 2017, doi: 10.3390/AEROSPACE4020031.

[36] M. Kamel, M. Burri, and R. Siegwart, "Linear vs Nonlinear MPC for Trajectory Tracking Applied to Rotary Wing Micro Aerial Vehicles," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 3463–3469, Jul. 2017, doi: 10.1016/J.IFACOL.2017.08.849.

[37] M. Ławryńczuk and R. Nebeluk, "Computationally Efficient Nonlinear Model Predictive Control Using the L1 Cost-Function," *Sensors 2021, Vol. 21, Page 5835*, vol. 21, no. 17, p. 5835, Aug. 2021, doi: 10.3390/S21175835.

[38] P. D. Domanski, "Performance Assessment of Predictive Control—A Survey," *Algorithms 2020, Vol. 13, Page 97*, vol. 13, no. 4, p. 97, Apr. 2020, doi: 10.3390/A13040097.

[39] A. Arce Rubio, A. Seuret, A. Mannisi, Y. Ariba, and A. Arce, “Optimal control strategies for load carrying drones,” vol. 6, pp. 183–197, 2016, doi: 10.1007/978-3-319-32372-5.

[40] Y. al Younes and M. Barczyk, “Nonlinear Model Predictive Horizon for Optimal Trajectory Generation,” *Robotics 2021, Vol. 10, Page 90*, vol. 10, no. 3, p. 90, Jul. 2021, doi: 10.3390/ROBOTICS10030090.

[41] R. Benotsmane and J. Vásárhelyi, “Nonlinear Model Predictive Control for Autonomous Quadrotor Trajectory Tracking,” in *Lecture Notes in Mechanical Engineering*, 4th ed., K. , C. Á. Jármai, Ed. Springer, Cham, 2023, pp. 24–34. doi: 10.1007/978-3-031-15211-5_3.