Robust Trajectory Tracking using High Order Sliding Mode Controllers-Observers for QuadCopter

NGUYEN Quang Huy

Supervised by

Ali Zemouche
Université de Lorraine

July 30, 2021

Abstract

This paper deals with the design of a supertwisting sliding mode controller based on High ordre sliding observer for the position control. That only the inertial coordinates and yaw angle are available for measurement, a high ordre sliding observer is designed which allows estimating on-line the velocity of the vehicle. The problem is solved in two steps: first, a high order sliding mode observer design is carried out by means a change of coordinates of the quadrotor system. Second, a dynamic sliding mode controller is proposed using the state estimate of the quadrotor . We show the asymptotic stability of the couple observer-controller. Finally, result of simulations and real time experiment are developed for showing the performance of the proposed observerbased control.

Contents

List of Figures

Chapter 1

Introduction

Unmanned Air Vehicles (UAVs) are being integrated into our society at an increasing rate. The amount of industries that use drones is getting larger every year. Besides the military sector, which was probably the first adopter of drone technology, they are now also being used in the industries such as search and rescue, delivery service, media, civil engineering, etc.

1.1 Motivation and Objectives

In research, the quadcopter is an exemplary design for small unmanned aerial vehicles with six degrees of freedom but only four independent inputs, thus, make it critically underactuated. To gain the six degrees of freedom, rotational and translational motions are coupled. As a result, the dynamics of this flying object are highly nonlinear, particularly under the effect of aerodynamics. Besides that, the quadrotor has microscopic friction to prevent its movement, so it must yield its damping to block the move and remain in a steady state. As a consequence, the design of controllers for the quadcopter becomes an extremely problematic task. A vast volume of controllers has been developed for quadrotors in literature, such as PID [10], H-infinity [12], Backstepping [13], Sliding Mode Control [5], [6] and Fuzzy logic [4]. Among them, SMC has been widely used because of its capability to robustly control systems under uncertainties and disturbances. Even though, the chattering phenomenon remains a significant disadvantage of the method. To eliminate chattering, high-order sliding modes (HOSM) [14], [3], have been offered as a most likely preferable solution.

Among them, super-twisting sliding mode (STSM) [7] is a unique continuous sliding mode algorithm, which ensures all essential properties of the first-order SMC together with chattering rejection. However, the performance of STSM depends on the knowledge of the bound of perturbations. In practical scenarios, the drones are affected by disturbances, uncertainties, modeling errors and parameter variations that may downgrade the control efficiency, but their boundaries are not obvious.

1.2 Contribution

The main topic of this work is devoted to the combination of the following sliding mode techniques: i) Modified SuperTwisting Sliding Mode Controller and ii) High Order Sliding Mode Observer. The mathematical model of the drone is derived by adopting possible vital variables while some others are considered to be uncertainties. The controller mentioned above is proposed to robust trajectory tracking for an Unmanned Aerial Vehicle the robustness while rejecting disturbances and parametric variations as well as decreasing affection of the chattering phenomenon. This control performance demonstrated by real-time experiment and comparison with the conventional Proportional- Integral-Derivative (PID) to show its advantageous feasibility.

1.3 Paper organization

The paper is organized as follows: The dynamic model of quadrotor under the external disturbances is described in Section 2. A discussion on the history of the Pixhawk hardware and a description of its firmware architecture concludes this chapter.

The development of the Super-Twisting Controller based on a High Order Sliding Mode Observer is showed and the application of the Controller to the aerial vehicle is carry out in Section 4. In subsection 4 covers the methodology how to implementing on the Pixhawk autopilot for simulation and flight testing. The simulation results are given in Sect. 4. Finally, the paper closes in Sect. 5 with conclusions.

Chapter 2

Quadrotor mathematical model

This chapter focuses on defining the coordinate frames used to control the drone and it establishes the mathematical model of a quadcopter

2.1 Coordinate frames

For the quadrotor, it is possible to use two reference systems. The first is fixed and the second is mobile. The fixed coordinate system, called also inertial, is a system where the first Newtons law is considered valid. As fixed coordinate system, we use the O_{NED} systems, where NED is for North-East-Down. As we can observe from the following *Figure* **??** a fixed ground (Earth) reference frame (E-frame), its vectors are directed to Nord, East and to the center of the Earth and a Body fixed frame (B-frame) in *Figure* 2.2 .It is assumed that the center of the reference frame is put in the center of the mass of the drone.

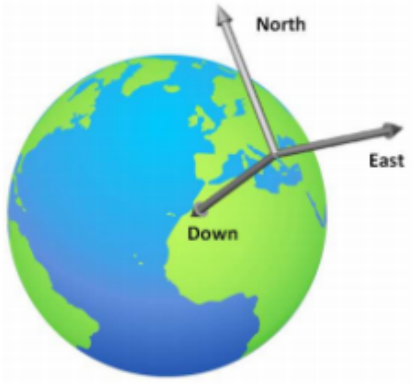


Figure 2.1: Flower one.

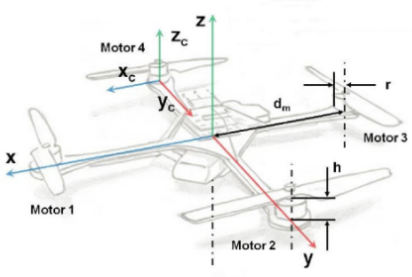
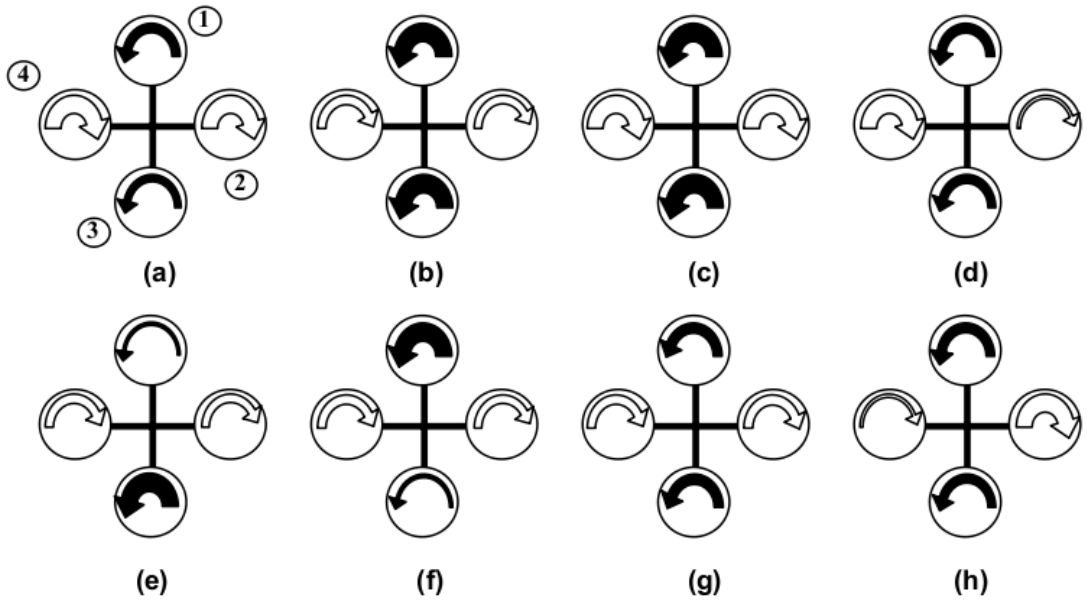


Figure 2.2: Flower two.

2.2 The control inputs

The attitude and position of the quadrotor can be controlled to desired values by changing the speeds of the four motors. There are 4 input signals that are introduced into the system: U1, U2, U3 and U4.



(a)	Yaw (anticlockwise direction)	**(e)**	Pitch (anticlockwise direction)
(b)	Yaw (clockwise direction)	**(f)**	Pitch (clockwise direction)
(c)	Take-off or take-up	**(g)**	Land or take-down
(d)	Roll (clockwise direction)	**(h)**	Roll (anticlockwise direction)

Figure 2.3: The UAV motors, their rotational direction

Based on these inputs, the four rotors of the UAV rotate accordingly. That means there must be a relationship between the input signals and the angular velocities of the rotors. In the system of equations (2.6), it can be seen very clearly how the control input signals are related to the angular velocities of the rotors, which are denoted as $\Omega_1, \Omega_2, \Omega_3$ and Ω_4 $[rad/s]$ for the motors 1, 2, 3 and 4 respectively. The values of c_T $[Ns^2]$ and c_Q $[Nms^2]$ are aerodynamic coefficients of thrust and drag, respectively. The value of l $[m]$ is the distance between the center of the quadrotor and the center of a propeller. Finally, the equation 2.2 adds up the rotational velocities of all the rotors. Since the propellers 1 and 3 rotate counter-clockwise and the propellers 2 and 4 rotate clockwise, the motors 1 and 3 have the opposite sign compared to the motors 2 and 4

$$\begin{aligned} U_1 &= c_T(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ U_\phi &= c_T l(\Omega_4^2 - \Omega_2^2) \\ U_\theta &= c_T l(\Omega_3^2 - \Omega_1^2) \\ U_\psi &= c_Q(\Omega_1^2 + \Omega_3^2)(\Omega_2^2 + \Omega_4^2) \end{aligned} \tag{2.1}$$

such that

$$\Omega = (\Omega_1 + \Omega_3) - (\Omega_2 + \Omega_4) \tag{2.2}$$

2.3 Equation of montion

The motion of the quadcopter can be divided into two subsystems; rotational subsystem (roll, pitch and yaw) and translational subsystem (altitude and x and y position). The rotational subsystem is fully actuated while the translational subsystem is underactuated.

2.3.1 Translational Equations of Motion

Using Newtons second law the translational equation of motion for the quadcopter is formulated and they are derived in the Earth inertial frame.

$$m\ddot{\eta} = mgZ_E + RU_1Z_E + F_p \tag{2.3}$$

η	Quadcopters distance from the inertial frame : $\eta = [x \quad y \quad z]^T$
m	Quadcopters mass.
g	The gravitational constant : $g = 9.81m/s^2$
U_1	Non-gravitational forces acting on the quadcopter in the body frame
F_p	The drag forces caused by external wind and gusts $F_p = [F_x \quad F_y \quad F_z]^T$
Z_E	Vector unit of z axis in World frame. $Z_E = [0 \quad 0 \quad 1]^T$

$$R = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\cos\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \tag{2.4}$$

By substituting (2.4) in (2.3) :

$$\begin{pmatrix} m\ddot{x} \\ m\ddot{y} \\ m\ddot{z} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -mg \end{pmatrix} + \begin{pmatrix} \cos\phi\cos\psi\sin\theta + \sin\phi\sin\psi \\ \cos\phi\sin\psi\sin\theta - \sin\phi\sin\psi \\ \cos\theta\cos\phi \end{pmatrix} U_1 + \begin{pmatrix} F_x \\ F_y \\ F_z \end{pmatrix} \tag{2.5}$$

2.3.2 Rotational Equations of Motion

The rotational equations of motion are derived in the body frame (B) using the Newton-Euler method with the following general formalism

$$J\dot{\omega} = \omega \times J\omega + M_G + M_B + M_r \tag{2.6}$$

Where

I	The inertia matrix of the rigid body : $J = diag(I_x, I_y, I_z)$
ω	Angular velocities in Body-frame : $\omega = [p \quad q \quad r]^T$
M_G	Gyroscopic moments due to rotors inertia : $M_G = \omega_r \times [0 \quad 0 \quad J_r\omega_r]^T$
M_B	Moments acting on the quadcopter in the body $M_B = [U_\phi \quad U_\theta \quad U_\psi]^T$
M_r	The drag moments caused by external wind and gusts M_r =$[M_\phi \quad M_\theta \quad M_\psi]^T$

we can substitute in (2.6) :

$$\begin{pmatrix} I_x\dot{p} \\ I_y\dot{q} \\ I_z\dot{r} \end{pmatrix} = \begin{pmatrix} (I_y - I_z)pr \\ (I_z - I_x)pr \\ (I_x - I_y)pq \end{pmatrix} + \begin{pmatrix} -J_r q\Omega \\ -J_r p\Omega \\ 0 \end{pmatrix} U_1 + \begin{pmatrix} U_\phi \\ U_\psi \\ U_\psi \end{pmatrix} + \begin{pmatrix} M_\phi \\ M_\psi \\ M_\psi \end{pmatrix} \tag{2.7}$$

Where

J_r rotors inertia.
Ω rotors relative speed : $\Omega = (\Omega_1 + \Omega_3) - (\Omega_2 + \Omega_4)$

To acquire information about the angular velocity of the quadcopter, typically an on-board Inertial Measurement Unit (IMU) is used which will in turn give the velocity in the body coordinate frame (p,q,r). To relate the Euler rates $\Phi = [\dot{\phi}, \dot{\theta}, \dot{\psi}]^T$ that are measured in the inertial frame and angular body rates $\omega = [p,q,r]^T$, a transformation is needed in order to transform from Φ to ω as follows:

$$\begin{pmatrix} p \\ q \\ r \end{pmatrix} = T \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} \tag{2.8}$$

With

$$T = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\psi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix}$$

Considering that the UAV will not do aggressive maneuvers but only small roll and pitch angles variations (Assumption 5 below), we can assume that T is very close to the identity matrix $T = I^{3x3}$. Thus the attitude dynamics model (2.6) can be expressed in the following equation as:

$$\begin{pmatrix} I_x\ddot{\phi} \\ I_y\ddot{\theta} \\ I_z\ddot{\psi} \end{pmatrix} = \begin{pmatrix} (I_y - I_z)\dot{\theta}\dot{\psi} \\ (I_z - I_x)\dot{\phi}\dot{\psi} \\ (I_x - I_y)\dot{\phi}\dot{\theta} \end{pmatrix} + \begin{pmatrix} -J_r\dot{\theta}\Omega \\ -J_r\dot{\phi}\Omega \\ 0 \end{pmatrix} U_1 + \begin{pmatrix} U_\phi \\ U_\psi \\ U_\psi \end{pmatrix} + \begin{pmatrix} M_\phi \\ M_\psi \\ M_\psi \end{pmatrix}$$

For this work , we are focus only in the Translation or Position dynamic model.

2.4 Pixhawk Autopilot

Wikipedia : PX4 autopilot is an open-source autopilot system oriented toward inexpensive autonomous aircraft.

Low cost and availability enable hobbyist use in small remotely piloted aircraft. The project started in 2009 and is being further developed and used at Computer Vision and Geometry Lab of ETH Zurich (Swiss Federal Institute of Technology) and supported by the Autonomous Systems Lab and the Automatic Control Laboratory. Several vendors are currently producing PX4 autopilots and accessories

An autopilot allows a remotely piloted aircraft to be flown out of sight. All hardware and software is open-source and freely available to anyone under a BSD license. Free software autopilots provide more flexible hardware and software. Users can modify the autopilot based on their own special requirements.

The open-source software suite contains everything to let airborne system fly including:

- QGroundControl and MAVLink Micro Air Vehicle Communication Protocol
- 2D/3D aerial maps (with Google Earth support) Drag-and-drop waypoints

Figure 2.4: Holybro pix32 Flight Controller

2.5 PX4 Architecture

PX4 software [1] is divided into two main layers: the flight stack and the middleware. The middleware consists of drivers for the embedded sensors, communication with the external world (computer, ground control station (GCS) et cetera), the uORB publish-subscribe message bus and the simulation layer.The flight stack is elaborated on in the following subsection.

2.6 Experimental design

The PX4 flight stack is a collection of guidance, navigation and control algorithms for autonomous drones. The flight stack includes controllers for multirotor, fixed wing and Vertical Take Off and Landing (VTOL) vehicles in addition to attitude and position estimators. Figure 2.5 provides a visual representation of the PX4 flight stack where each block is a module or application.

The estimators receive sensor readings, combine them and use an algorithm the extended kalman filter (EKF) to estimate the vehicle states. These estimated states are used by the navigator, an algorithm for autonomous flight control, the position

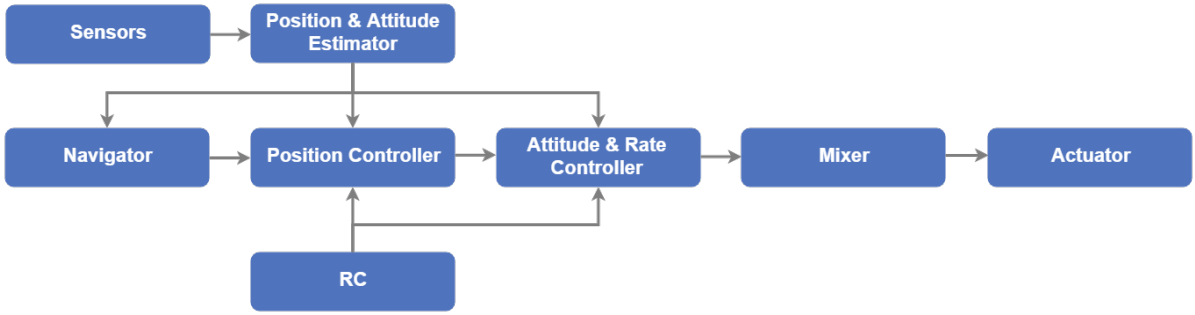


Figure 2.5: PX4 flight stack

controller and the attitude and rates controller. The position and attitude (and rates) controllers receive remote control (RC) input of the vehicle position and/or orientation desired by the user, via the radio receiver connected to the Pixhawk. The RC and the receiver are connected wirelessly through radio protocols at a particular radio frequency. The rates controller outputs commands, like turning left, to the mixer block which converts them into motor commands for each motor. Finally, these motor commands are sent to the ESCs to regulate the speed of each motor (actuator) as needed.

The position module is comprised of two loops: Proportional (P) controller loop for position error and a Proportional Derivative (PID) controller loop for velocity error.

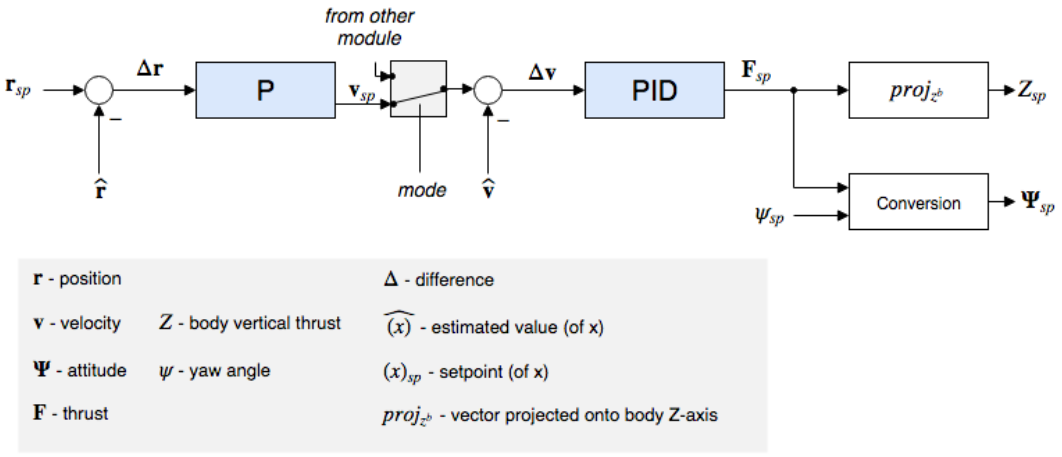


Figure 2.6: PX4 Position controller diagram

Our work is consist replace the cascasde P and PID controller for a block with STSMC to control position and velocity error.

Chapter 3

Controller Design and Implementation

The design and set up the HOSMO-STC controller are formulated in section 3.1. In section 3.2, the HOSMO-STC controller is implemented in MATLAB. Finally, section 3.3 the procedure for implementing this controller on the Pixhawk hardware for flight testing.

3.1 Control and observer design

In this Section we present the mathematical structure of the Super-Twisting Sliding Mode Controller and the High Order Sliding Mode Observer, which are the base for the Controller-Observer proposed in this work.
Hereafter, temporal dependence of the t variable will be omitted for simplicity. Consider a second order dynamic system defined as:

$$\begin{aligned} \dot{X}_1 &= X_2 \\ \dot{X}_2 &= f(X) + h(X)u + \xi(X) \\ y &= X_1 \end{aligned} \tag{3.1}$$

Here $X(t) = [x_1, x_2]^T \in R^2$ represents the state vector, $y(t) \in R$ is the system output, $u(t)$ is the input control signal, $f(X) \in R$ and $h(X) \in R$ are known functions. Moreover, the term $\xi(X) \in R$ includes both unmodeled dynamics as external disturbances. We assume that disturbance $\xi(X)$ is Lipschitz and satisfies $\dot{\xi} < \xi^+$ for a positive constant ξ.

3.1.1 High Order Sliding Mode Observer

The High Order Sliding Mode Observer used to estimate the dynamic x_2 as well as the disturbance ξ of system 3.1 is given as:

$$\begin{aligned} \dot{\hat{x}}_1 &= \hat{x}_2 + \lambda_1 \widetilde{X}_1^{2/3} sign(\widetilde{X}_1) \\ \dot{\hat{x}}_2 &= \hat{x}_3 + \lambda_2 \widetilde{X}_1^{1/3} sign(\widetilde{X}_1) + f(X) + h(X)u \\ \dot{\hat{x}}_3 &= \lambda_3 \widetilde{X}_1 sign(\widetilde{X}_1) \end{aligned} \tag{3.2}$$

where $\hat{x}_i$, $i = 1,...,3$, represents the estimated variables. The estimation errors are defined as :

$$\begin{aligned}
\dot{\tilde{X}}_1 &= x_1 - \hat{x}_1, \\
\dot{\tilde{X}}_2 &= x_2 - \hat{x}_1, \\
\dot{\tilde{X}}_3 &= x_3 - \hat{x}_1 \qquad with \quad x_3 = \xi
\end{aligned}$$

The dynamics of the estimation errors are obtained as follows :

$$\begin{aligned}
\dot{\tilde{X}}_1 &= \hat{x}_2 - \lambda_1 |\tilde{x}|_1^{2/3} sign(\tilde{x}_1) \\
\dot{\tilde{X}}_2 &= \hat{x}_3 - \lambda_2 |\tilde{x}|_1^{1/3} sign(\tilde{x}_1) \\
\dot{\tilde{X}}_3 &= -\lambda_3 |\tilde{x}|_1 sign(\tilde{x}_1)
\end{aligned}$$

The above equation is finite time stable which is already proved in literature [2] by geometric methods and [11] by using a quadratic and strict Lyapunov function. Therefore, the estimation errors $\dot{\tilde{X}}_1$, $\dot{\tilde{X}}_2$ and $\dot{\tilde{X}}_3$ will converge to zero in a finite time $t \geq T_0$ if gains λ_1, λ_2 and λ_3 are chosen appropriately [9]. From the error variables definition, after finite time t T0, it results that $\hat{x}_1 = x_1$, $\hat{x}_2 = x_2$ and $\hat{x}_3 = \xi$.

3.1.2 Super-Twisting Sliding Mode Controller

It is well known that the main drawback of the First Order Sliding Mode Control is the chattering phenomenon that appears in the control law. As a solution to resolve this problem, authors in [8] presented Second Order Sliding algorithms, such as the Twisting algorithm, where the idea of acting on the superior derivatives of the sliding variable was introduced. On the other hand, in the case of the Second Order Sliding Mode Control, the following condition should be verified

$$s(X) = \dot{s}(X) = 0,$$

where $s(X)$ represents the sliding variable.

$$s = c_1 e_1 + \hat{e}_2 = 0$$

Where $c_1 \in R^+$, $e_1 = x_1 - x_1^d$, and $\hat{e}_2 = \hat{x}_2 - x^d$. The desired position and velocity are denoted by x_1^d and x_2^d respectively.

The temporal derivative of the sliding variable is given as:

$$\dot{s} = c_1 \dot{e}_1 + \dot{\hat{e}}_2$$

by substituting the dynamics of (3.1) and (3.2) we obtain

$$\dot{s} = c_1 \hat{x}_2 - c_1 \dot{x}_1^d + \int_0^t \lambda_3 sign(\tilde{x}_1) dt + \lambda_2 \tilde{x}_1^{1/3} sign(\tilde{x}_1) + f(X) + h(X)u - \dot{x}_2^d \qquad (3.3)$$

The main aim here, is to design a continuous control u, such that the SOSM occurs in finite time on the sliding surface. For this purpose control is selected according to the following proposition.

Proposition 1. The following control input leads to the establishment of SOSM on s in finite time, once $s = 0$ it further implies asymptotic stability of x_1 and x_2,

$$u = h(X)^{-1}[-c_1\hat{x}_2 + c_1\dot{x}_1^d - \int_0^t \lambda_3 sign(\widetilde{x}_1)dt - \lambda_2|\widetilde{x}_1|^{1/3}sign(\widetilde{x}_1) - f(X) + \dot{x}_2^d \\ - k_1 s^{1/2}sign(s) - \int_0^t k_2 sign(s)dt] \quad (3.4)$$

By properly selecting the gains k_1, k_2, k_3 and k_4, then $s = \dot{s} = 0$ in finite time, which implies asymptotic stability of the tracking errors e_1 and $\hat{e}_2$.
Substituting the Super-Twisting Controller 3.4 in 3.3 we obtain :

$$\dot{s} = -c_1\hat{x}_2 + c_1\dot{x}_1^d + \dot{x}_2^d - k_1|s|^{1/2}sign(s) - \int_0^t k_2 sign(s)dt \quad (3.5)$$

Therefore, the closed-loop Controller-Observer system is given by the following expressions

$$\Xi : \begin{cases} \dot{x}_1 = s - c_1 e_1 + \widetilde{x}_1 + x_2^d \\ \dot{s} = c_1\widetilde{x}_2 k_1|s|^{1/2}sign(s) + v \\ \dot{v} = \int_0^t k_2 sign(s)dt \end{cases} \quad (3.6)$$

$$\Pi : \begin{cases} \dot{\widetilde{x}}_1 = \hat{x}_2 - \lambda_1|\widetilde{x}_1|^{2/3}sign(\widetilde{x}_1) \\ \dot{\widetilde{x}}_2 = \hat{x}_3 - \lambda_2|\widetilde{x}_1|^{1/3}sign(\widetilde{x}_1) \\ \dot{\widetilde{x}}_3 = -\lambda_3|\widetilde{x}_1|sign(\widetilde{x}_1 \end{cases} \quad (3.7)$$

The estimation error of observer 3.7 converges to zero in finite-time. Consequently, we obtain for the error variables that $\widetilde{x}_1 = \widetilde{x}_2 = \widetilde{x}_3 = 0$. After time T_0, the closed-loop system is given by the dynamics:

$$\Xi : \begin{cases} \dot{x}_1 = s - c_1 e_1 + x_2^d \\ \dot{s} = c_1\widetilde{x}_2 k_1|s|^{1/2}sign(s) + v \\ \dot{v} = \int_0^t k_2 sign(s)dt \end{cases} \quad (3.8)$$

The lower two equation of 3.8 is a STA, by selecting appropriate gains $\lambda_1 > 0$ and $\lambda_2 > 0$, then $s = \dot{s} = 0$ in finite time, This implies that the tracking errors e_1 and e_2 are asymptotically stable. Finally, we obtain that x_1 and $\hat{x}_2$ converges asymptotically to x_1^d and x_2^d respectively. The block diagram for super-twisting control based on High ordre sliding observer is depicted in Fig 3.1.

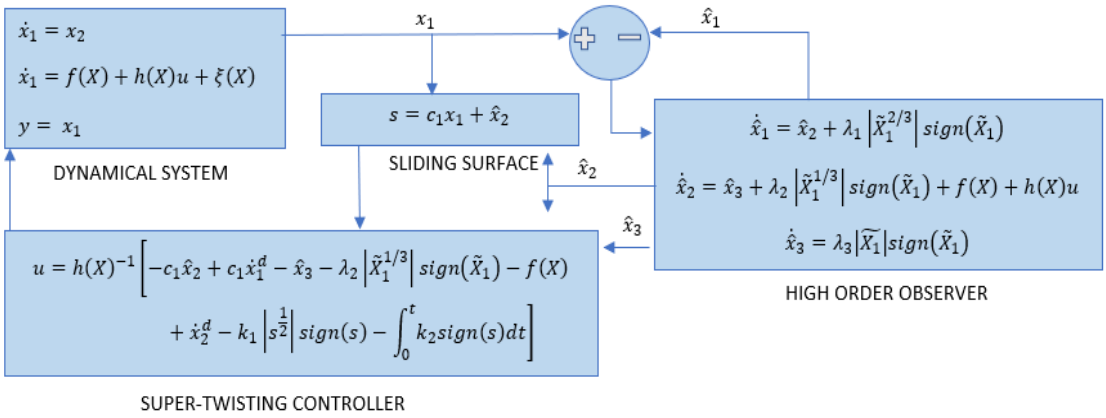


Figure 3.1: Block diagram of the Super-twisting Control based on HOSM Observer

3.2 Application of HOSMO based STC for the position control of Quadrotor

The Quadrotors translational dynamics in XYZ axis represented by 2.3 can be rewritten as :

$$\begin{aligned} \dot{x} &= v_x \\ \dot{v_x} &= u_x + \xi_x, \\ \dot{y} &= v_y \\ \dot{v_y} &= u_y + \xi_y, \\ \dot{z} &= v_z \\ \dot{v_z} &= u_z + \xi_z - g. \end{aligned} \tag{3.9}$$

Where u_x , u_y and u_z new virtual control input to obtain the total thrust and to find the desired angles (ϕ^d ,θ^d), the virtual control signals are defined as:

$$\begin{cases} u_x = (cos\ x_1\ sin\ x_3\ cos\ x_5 + sin\ x_1\ sin\ x_5)\frac{U_1}{m} \\ u_y = (cos\ x_1\ sin\ x_3\ sin\ x_5 + sin\ x_1 cos\ x_5)\frac{U_1}{m} \\ u_z = (-g + cosx_1\ cos\ x_3\ )\frac{U_1}{m} \end{cases} \tag{3.10}$$

The term F_p treated as a unmodeled dynamic and external disturbances can be included in the term ξ_z

In order to track a desired trajectoire x_d, y_d and z_d, we employ the Super-Twisting Controller based on the High Order Sliding Mode Observer presented in Section II. The dynamics of observer (3.2) applied to the translations dynamics (3.9) are given by

:

$$\begin{cases} \dot{\hat{x}} = \hat{v}_x + \lambda_{x1}|\widetilde{x}|^{2/3} sign(\widetilde{x}) \\ \dot{\hat{v}}_x = \hat{\xi}_x + \lambda_{x2}|\widetilde{x}|^{1/3} sign(\widetilde{x}) + u_x \\ \dot{\hat{\xi}}_x = \lambda_{x3}|\widetilde{x}| sign(\widetilde{x}) \end{cases} \tag{3.11}$$

$$\begin{cases} \dot{\hat{y}} = \hat{v}_y + \lambda_{y1}|\widetilde{y}|^{2/3} sign(\widetilde{y}) \\ \dot{\hat{v}}_y = \hat{\xi}_y + \lambda_{y2}|\widetilde{y}|^{1/3} sign(\widetilde{y}) + u_y \\ \dot{\hat{\xi}}_y = \lambda_{y3}|\widetilde{y}| sign(\widetilde{y}) \end{cases} \tag{3.12}$$

$$\begin{cases} \dot{\hat{z}} = \hat{v}_z + \lambda_{z1}|\widetilde{z}|^{2/3} sign(\widetilde{z}) \\ \dot{\hat{v}}_z = \hat{\xi}_z + \lambda_{z2}|\widetilde{z}|^{1/3} sign(\widetilde{z}) + u_z - g \\ \dot{\hat{\xi}}_z = \lambda_{z3}|\widetilde{z}| sign(\widetilde{z}) \end{cases} \tag{3.13}$$

Let us define a sliding variable as

$$s_x = c_x(x - x^d) + \hat{v}_x - v_x^d \quad , \quad s_y = c_y(y - y^d) + \hat{v}_y - v_y^d \tag{3.14}$$
$$s_z = c_z(z - z^d) + \hat{v}_z - v_z^d.$$

where c_x, c_y and $c_z > 0, \in R^+$. The virtual control (3.10) is given by

$$u_x = -c_{1x}\hat{x}_2 + c_{1x}\dot{x}^d - \int_0^t \lambda_{3x} sign(\widetilde{x})dt - \lambda_{2x}|\widetilde{x}|^{1/3} sign(\widetilde{x}) + \dot{v}_x^d - k_{1x} {s_x}^{1/2} sign(s_x) - \int_0^t k_{2x} sign(s_x)dt$$

$$u_y = -c_{1y}\hat{y}_2 + c_{1y}\dot{y}^d - \int_0^t \lambda_{3y} sign(\widetilde{y})dt - \lambda_{2y}|\widetilde{y}|^{1/3} sign(\widetilde{y}) + \dot{v}_y^d - k_{1y} {s_y}^{1/2} sign(s_y) - \int_0^t k_{2y} sign(s_y)dt$$

$$u_z = -c_{1z}\hat{z}_2 + c_{1z}\dot{z}^d - \int_0^t \lambda_{3z} sign(\widetilde{z})dt - \lambda_{2z}|\widetilde{z}|^{1/3} sign(\widetilde{z}) + \dot{v}_z^d - k_{1z} {s_z}^{1/2} sign(s_z) - \int_0^t k_{2z} sign(s_z)dt$$

Virtual controllers above guarantee that x, y and z tracks asymptotically to x^d, y^d and z^d, respectively. The necessary angles ϕ^d and θ^d that enables the system to track desired positions x^d, y^d and z^d are:

$$\begin{cases} U_1 &= \sqrt{u_x^2 + u_y^2 + \left(u_z^2 + g\right)^2} \\ \phi_d &= \arctan\left(\cos\theta_d \left(\frac{u_x \sin\psi_d - u_y \cos\psi_d}{u_z + g}\right)\right) \\ \theta_d &= \arctan\left(\frac{u_x \cos\psi_d + u_y \sin\psi_d}{u_z + g}\right) \end{cases}$$

The discontinuous switching component $\text{sgn}(s)$ is used to maintain trajectories on the surface $s = 0$. However, because there is uncertainty in every system, s is typically always varying about 0, causing the control to rapidly switch signs so that $s = 0$ can be maintained. This creates an undesired effect known as chattering, which can be seen in

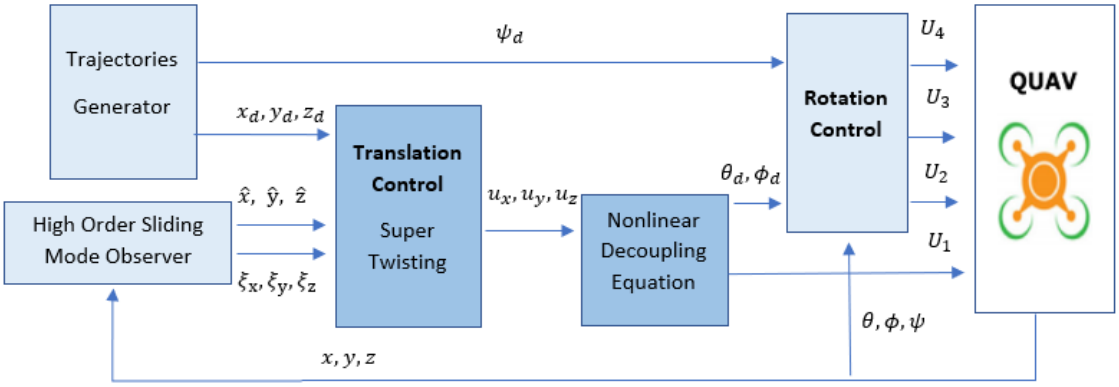


Figure 3.2: Block Diagram of the Super Twisting Controller based on a High Order Sliding Mode Observer for the Unmanned Aerial System

the control effort from the inner loop stabilization controller above. To alleviate this, a margin is created around the surface $s = 0$ so that the sensitivity is reduced and as long as trajectories are in the neighborhood of $s = 0$ the control is successful. This can be implemented by replacing the signum function $\text{sgn}(s)$ with a saturation function, as shown below. The transition region can be controlled by scaling the surface by a variable ε, such as $\text{sat}(\frac{s}{\varepsilon})$, where $0.1 \leq \varepsilon \leq 1$ is a typical range.

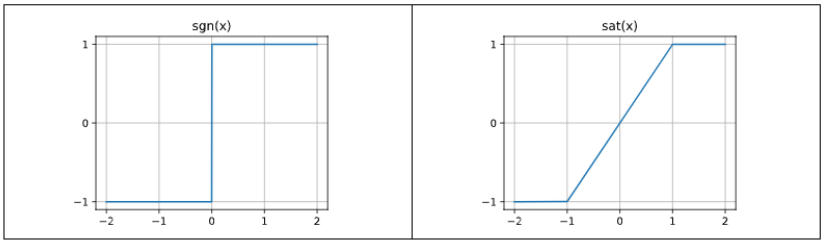


Figure 3.3: Sign and Saturation function

3.3 Implementation

On successfully program the controller in C++, in Microsoft Visual Studio, the next step was to move the relevant variables, vectors and functions to the position control module on the PX4 firmware

This phase of the controller implementation was done in a Linux environment using the Ubuntu 18.04 LTS operating system.

The default firmware was cloned from the PX4 firmware GitHub repository to a directory on the laptop.

The next step was to run the PX4 firmware on a simulated quadcopter

3.3.1 Gazebo

Gazebo is an open-source based multicopter simulator, that enables a user to fly a vehicle running the PX4 firmware in a simulated environment. The following command was executed in the terminal to run SITL using Gazebo:

```
cd /path/to/PX4-Autopilot
make px4_sitl_gazebo
```

On successful run, the PX4 shell console was opened which enables the user to type in commands to take off and land the simulated quadcopter

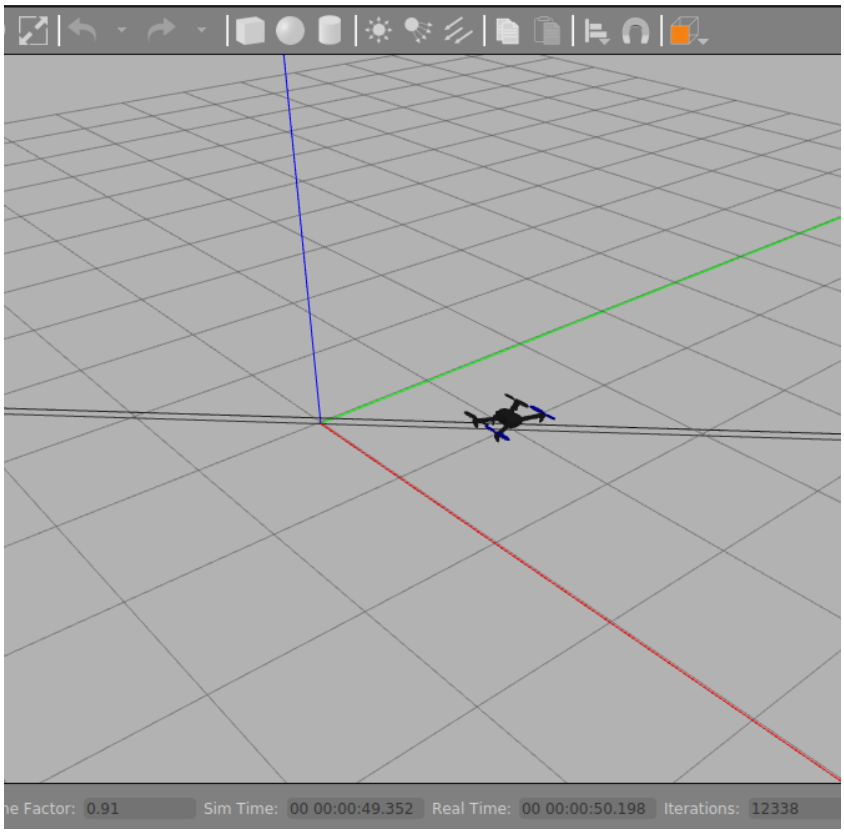


Figure 3.4: Gazebo world simulation

The ground station software, QGroundControl (QGC), was connected to Gazebo to fly a mission with the quadcopter or use a joystick .The flight logs from the SITL simulation were logged and downloaded using QGC.

3.3.2 Building the PX4 firmware

Building the PX4 firmware was done from the Ubuntu terminal window. The following commands were executed in the order shown below to build the firmware:

```
cd Firmware
make px4fmu-v2_default
```

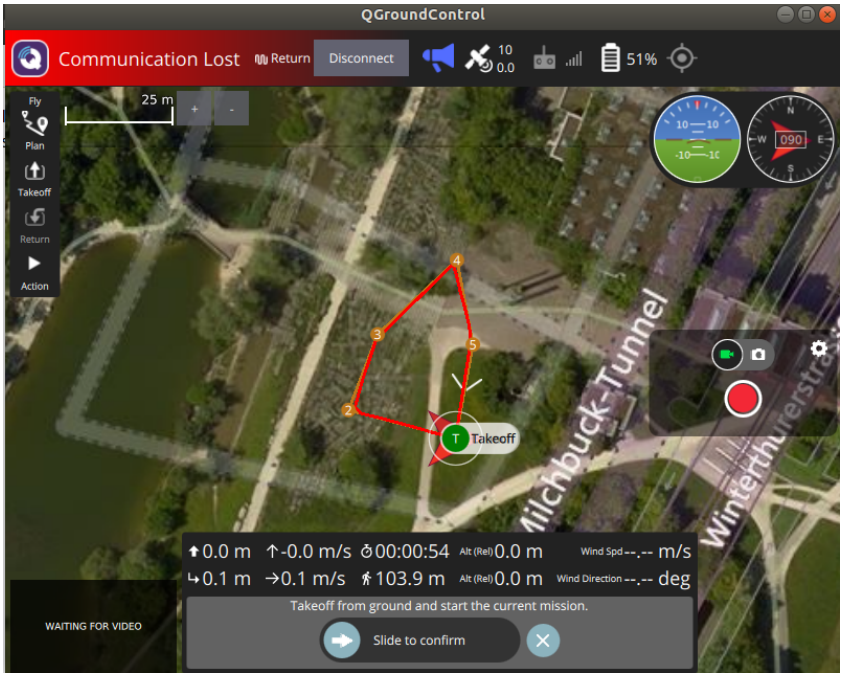


Figure 3.5: QGround Control

After successful SITL simulations, the next step was to upload the modified PX4 firmware onto the Pixhawk for flight testing. The Pixhawk was connected to the computer via a USB cable. A terminal window was opened and the directory containing the PX4 firmware is navigated to. Finally, the following command was executed in order to build and upload the firmware to the Pixhawk:

```
make px4fmu-v2_default upload
```

All of these steps contributed in successfully uploading the modified PX4 firmware, with the STSMC position controller, onto the Pixhawk.

Chapter 4

Simulation and Experiments

In this section, the position dynamical model of the vehicle established in Eq.2.3, 2.6 is used to check the effectiveness and to show the performance of the proposed control laws under external constant and time-varying disturbances.

The results obtained from running the simulations in MATLAB are presented in section 4.1.Section 4.2 presents the results obtained from the SITL simulations. Finally, the flight test results from flying the quadcopter running the modified PX4 firmware are plotted and discussed in section 4.3

4.1 MATLAB

The values dynamic parameters used in simulations in this paper are displayed in Table 4.1, 4.2, 4.3 .

Parameter	Value	Unit
m	0.8	kg
l	0.2	m
g	9.81	m/s^2
$I_x = I_y$	0.007	Ns^2/rad
I_z	0.01	Ns^2/rad

Table 4.1: QUADROTOR MODEL PARAMETERS

Position gains	Value
$k1x$	2
$k2x$	3
$k1y$	2
$k2y$	3

Table 4.2: SUPER-TWISTING CONTROLLER PARAMETERS

Parameter	Value
λ_{1x}	5
λ_{2x}	20
λ_{3x}	6
λ_{1y}	5
λ_{2y}	20
λ_{3y}	6
λ_{1z}	5
λ_{1z}	20
λ_{1z}	2

Table 4.3: HIGH ORDER SLIDING MODE OBSERVER PARAMETERS.

The desired trajectories employed in the simulation test are defined as follows

$$X^d = \begin{cases} 0 & ,0 < t < 0.1 \\ 2sin(t) & ,t > 1 \end{cases} \tag{4.1}$$

$$Y^d = \begin{cases} 2 & ,0 < t < 0.1 \\ 2cos(t) & ,t > 1 \end{cases} \tag{4.2}$$

In this scenario, the presence of the external perturbations given as follows:

$$D_x = \begin{cases} 1 + sin(0.5t), & 15 < t < 25 \\ 1, & 30 < t < 35 \\ 2, & t > 50 \\ 1, & otherwise \end{cases} \tag{4.3}$$

$$D_y = \begin{cases} 1 & ,15 < t < 25 \\ 1 + sin(0.5t), & 30 < t < 35 \\ 2, & t > 50 \\ 1, & ,otherwise \end{cases} \tag{4.4}$$

As it can be seen in Fig.4.1, Fig.4.2, this control strategy has very high success. In tracking the x, y, z reference position values, one can observe overshoot at the beginning of the test period.That can be explained by the fact that the drone starts its journey from quite a long distance away from the trajectory. However, once it reaches the path that it needs to follow, the velocities of the UAV stabilize and track the reference values very smoothly even under the influence of external disturbances.

Figure 4 shows the Quadrotors translational coordinates in a 3-dimensional environment, where we can see that the performance of these dynamics is as expected. The virtual control signals U_x, U_y andU_z (generated by the position controller) given by Fig :4.3:

Fig.4.4 and Fig.4.5 show the performance of the high order sliding mode observer algorithm in the reconstruction of the system states variables, and of the proposed disturbance.

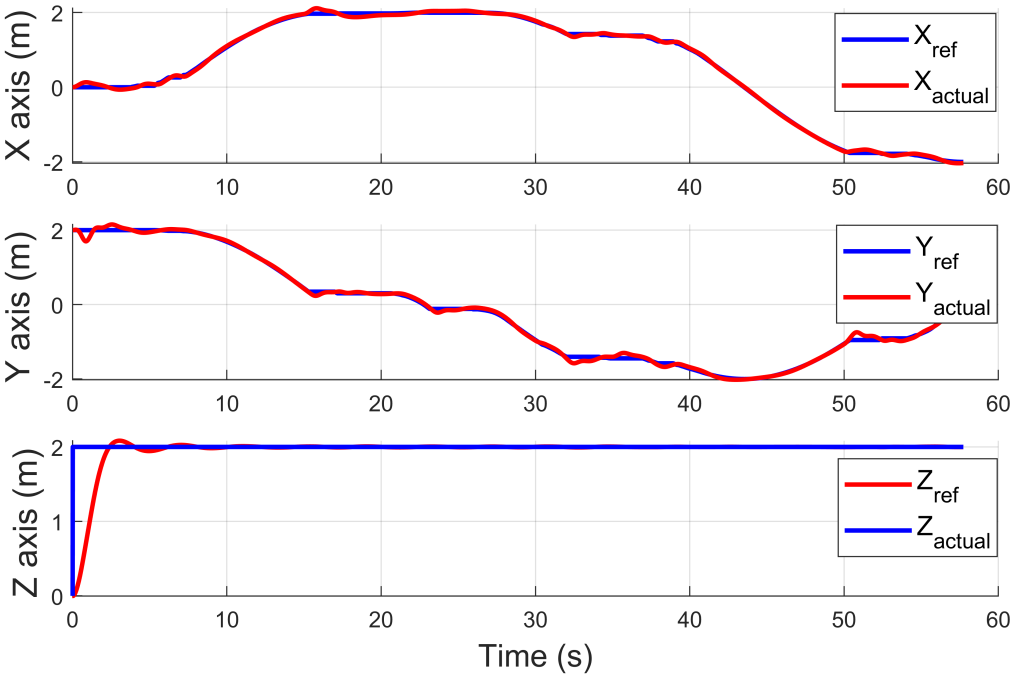


Figure 4.1: Holybro pix32 Flight Controller

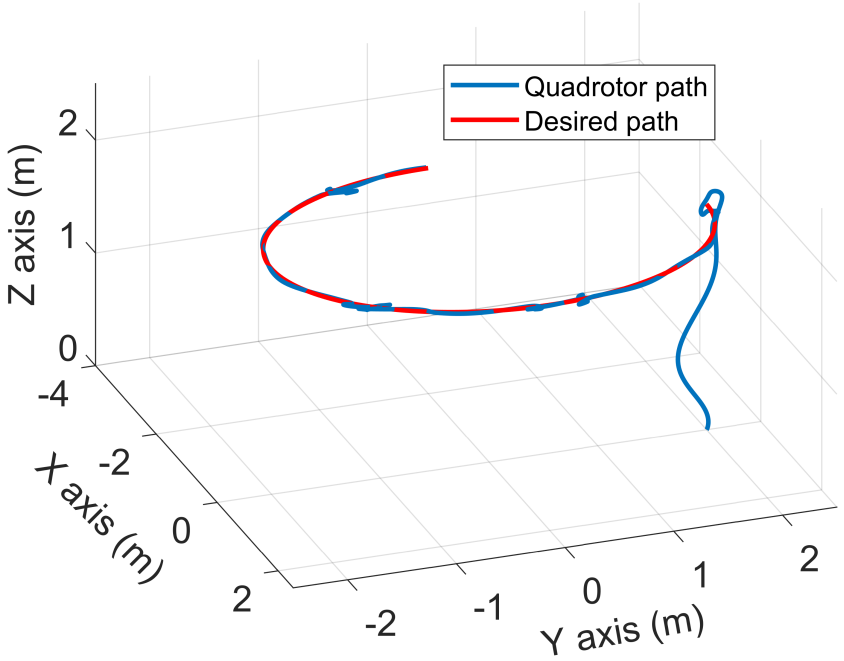


Figure 4.2: Quadrotors translational positions in 3D.

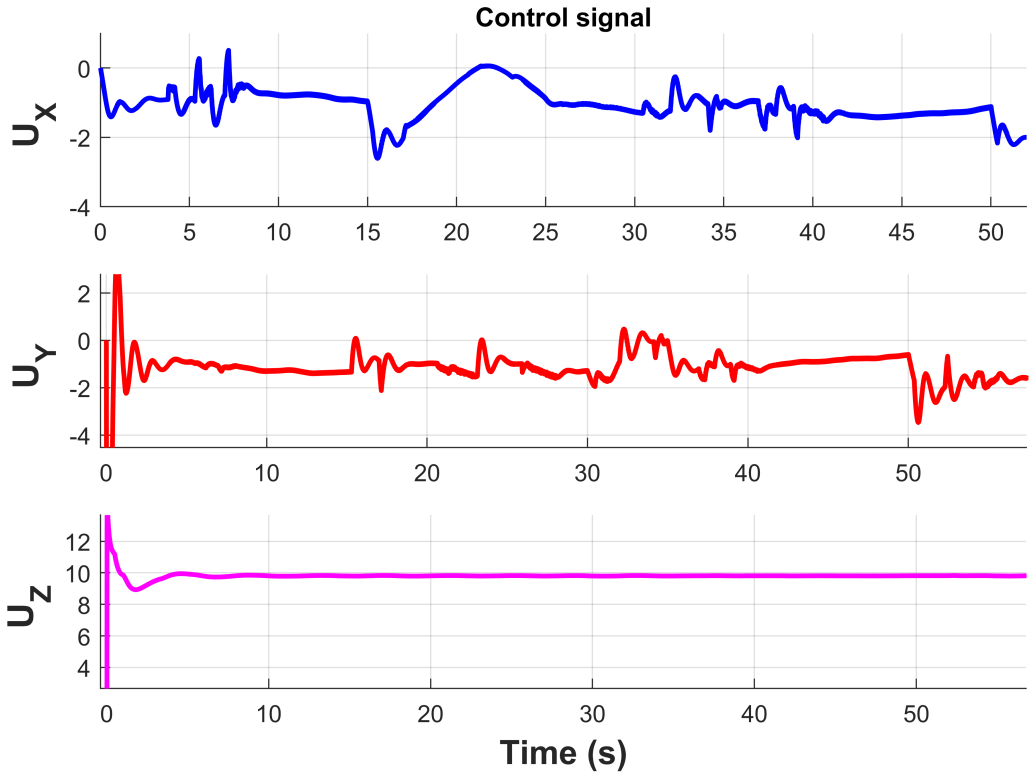


Figure 4.3: Virtual control signals U_x and U_y

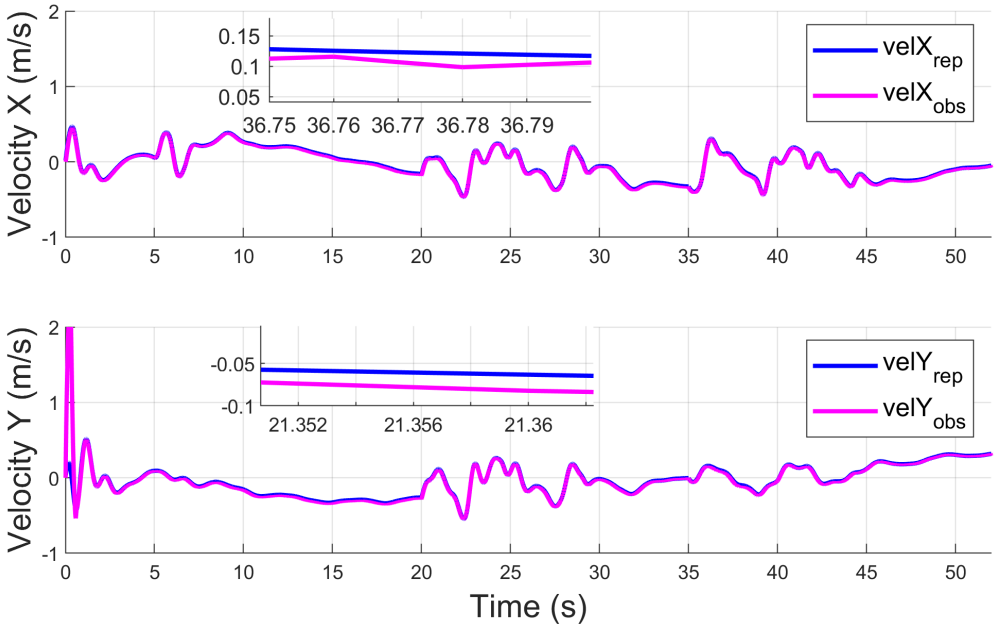


Figure 4.4: Real velocity v, and estimate velocity $\hat{v}$

Otherwise, The effects of the disturbance are estimated on both X and Y . It can be noted that the sign function exhibits high-frequency switching, this can cause consequences on the continuity of the final control. Compared to saturated function, the estimated disturbances lose some precision, but it's much more smooth to feed that into the control.

4.2 Gazebo Results

The SITL simulation results were obtained by flying the simulated quadcopter running the modified PX4 firmware in Gazebo through flight mission waypoints created in QGroundControl.

In the this scenario presented in Fig. 4, an external wind speed of 4 m/s is applied. We can notice in the moment take off , quadrotor have a overshoot in z-direction but after that controllers succeed to compensate the wind effect and maintain the requested path of the system.

4.3 Experimental Results

For the implementation of the proposed algorithms in outdoor environments we used a Ublox Neo-M8N GPS (20$) module that provides location accuracy of around 2 to 1 meters accuracy in the XY-plane. It's not a good choice but with a small project that works quite well.

The result obtains in an environment under wind speed 2.8 m/s.

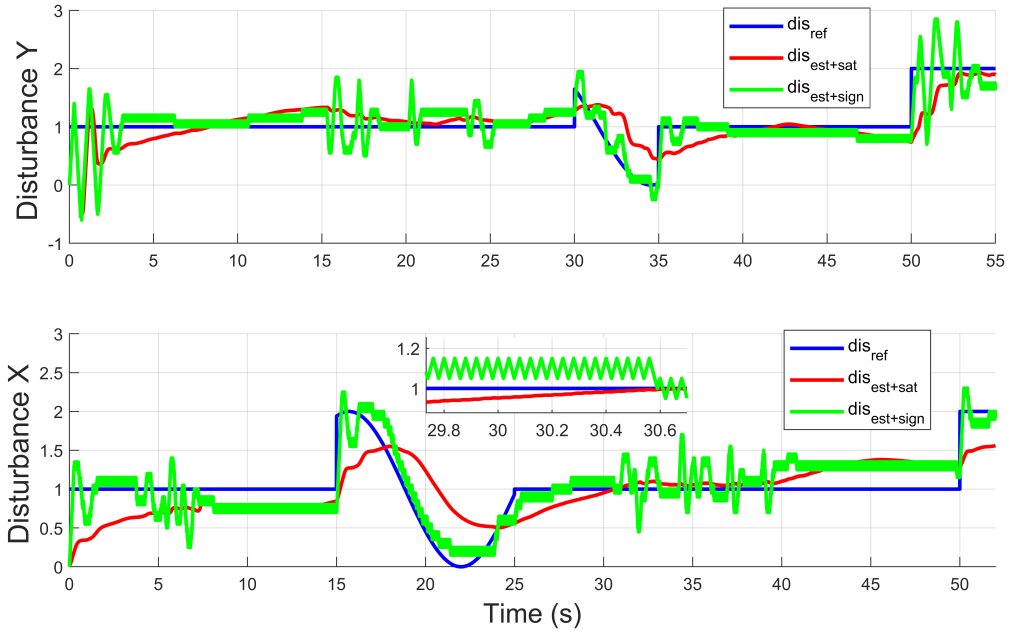


Figure 4.5: Estimation of the disturbances induced on the acceleration dynamics of X and Y axis

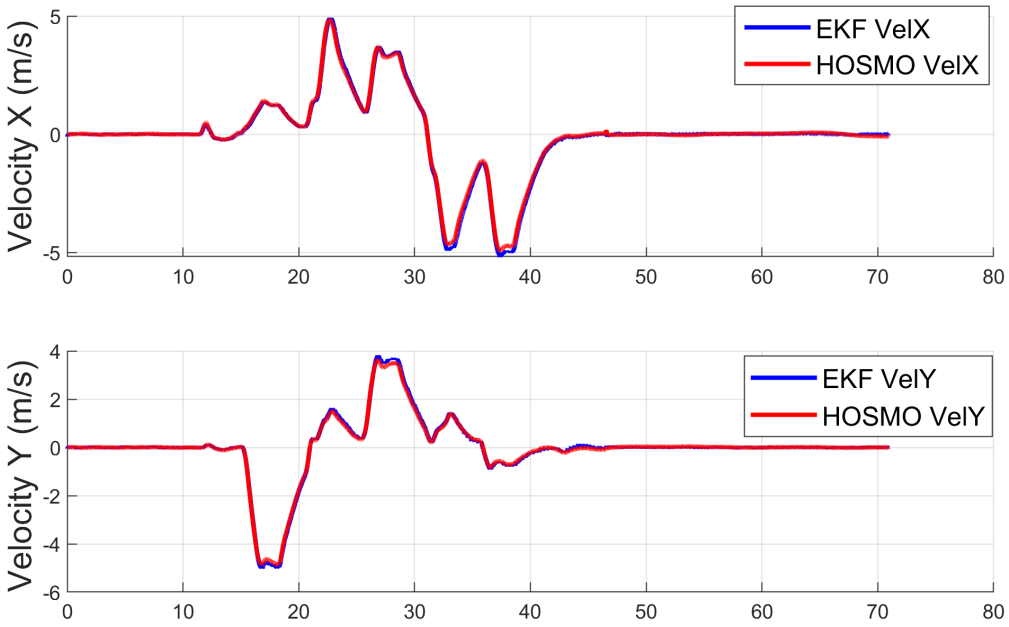


Figure 4.6: Real velocity v and estimate velocity $\hat{v}$

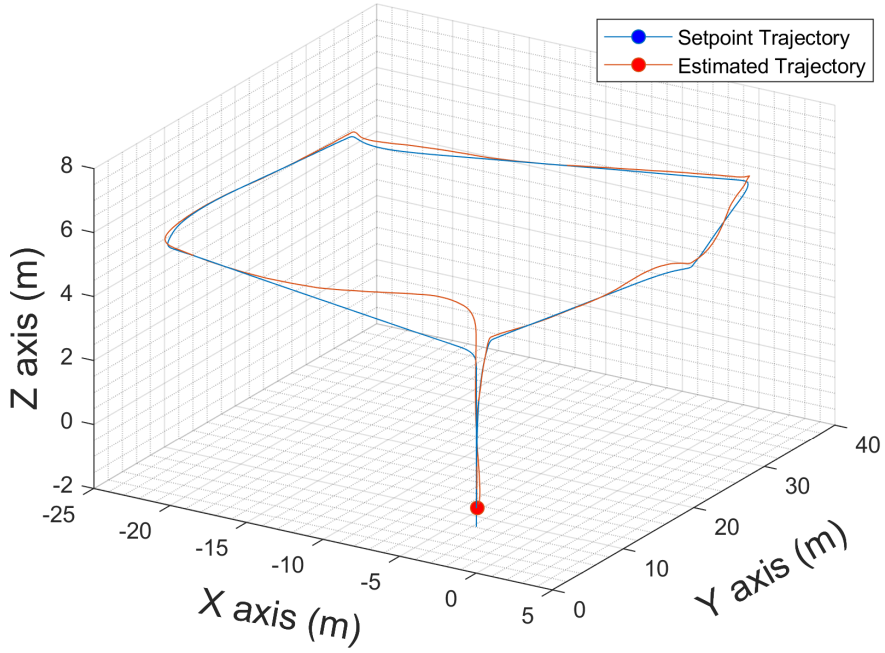


Figure 4.7: 3D trajectories

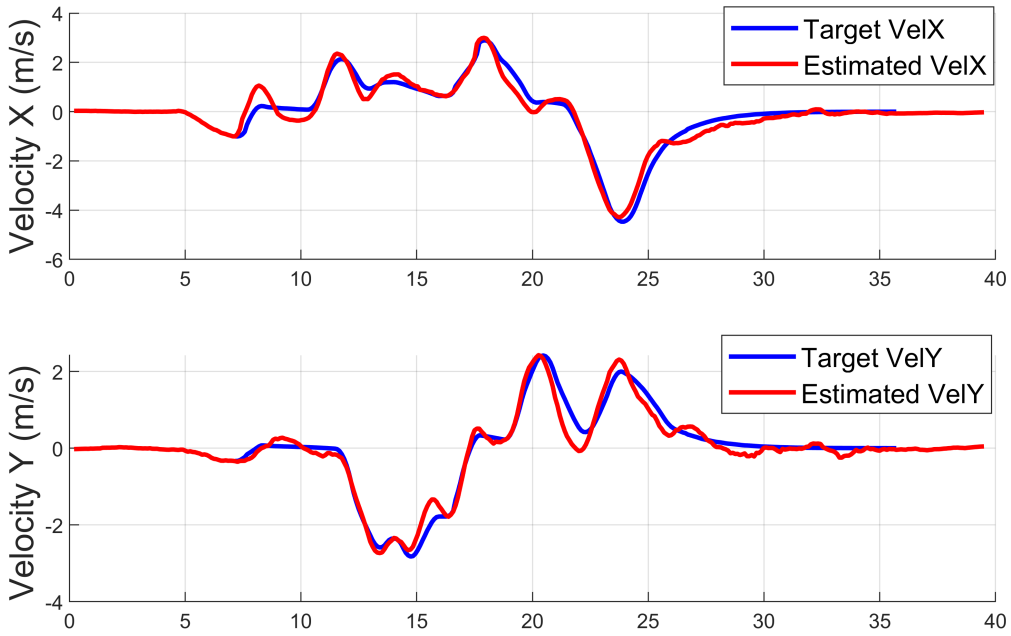


Figure 4.8: Setpoint velocity v, and estimate velocity $\hat{v}$

Despite these flight conditions, it can be observed that the estimated velocity by HOSMO, running on the Pixhawk, was able to closely track the velocity setpoint.

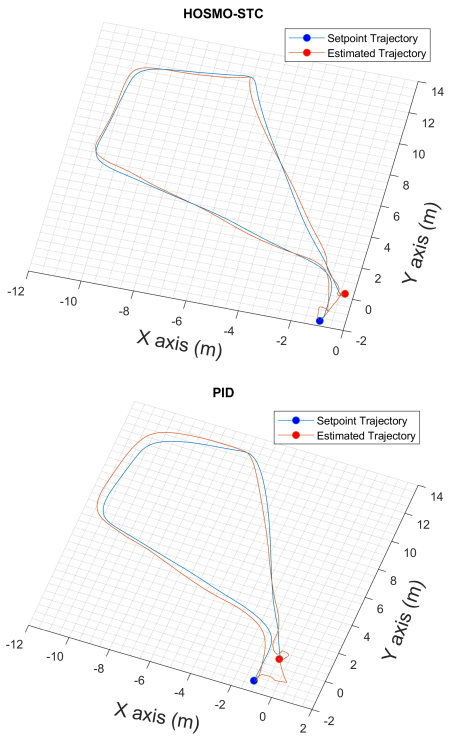


Figure 4.9: Trajectory tracking used PID vs HOSMO-STC controller

By comparing the root mean squared of the quadrotors positions errors in TABLE 4.4 , Generally, the trajectory tracking error of the HOSM controlled system is less than

	RMSE x (m)	RMSE y (m)
HOSMO-STC	0.2189	0.3033
PID	0.2256	0.4102

Table 4.4: Position mean squared errors in x,y directions

that of the PID controlled system. Obviously is not as perfect as in the simulations due to the sensor noises and model imprecisions.
A video showing the conducted experiments can be observed in https://youtu.be/GYm3GgSro2w.
All code associated to my rapport is share in :
https://github.com/kslhuy/Slidingmode-sim

Chapter 5

Conclusions and Future Work

In this work, the Super-Twisting Controller has been proposed for the trajectory tracking problem of an UAV type Quadrotor.The mathematical analysis demonstrate the High Order Sliding Mode Observer to estimate the translational velocities of the system and to improve the ability of system performance robustness.The Controller-Observer proposed ensures that the tracking errors converge to zero in finite-time. Finally, experimental results were obtained to show the performance of the control strategy and the experimental setup.

As future work we are planning to implement a new observer to improve estimation wind effect in order to perform outdoors experiment.

Bibliography

[1] https://docs.px4.io/master/en/. 2.5

[2] M. T. Angulo, J. A. Moreno, and L. Fridman. Robust exact uniformly convergent arbitrary order differentiator. *Automatica*, 49(8):2489–2495, 2013. 3.1.1

[3] A. Benallegue, A. Mokhtari, and L. Fridman. High-order sliding-mode observer for a quadrotor uav. *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal*, 18(4-5):427–440, 2008. 1.1

[4] B. Erginer and E. Altuğ. Design and implementation of a hybrid fuzzy logic controller for a quadrotor vtol vehicle. *International Journal of Control, Automation and Systems*, 10(1):61–70, 2012. 1.1

[5] I. González, S. Salazar, and R. Lozano. Chattering-free sliding mode altitude control for a quad-rotor aircraft: Real-time application. *Journal of Intelligent & Robotic Systems*, 73(1):137–155, 2014. 1.1

[6] I. Gonzalez, S. Salazar, R. Lozano, and J. Escareno. Real-time altitude robust controller for a quad-rotor aircraft using sliding-mode control technique. pages 650–659, 2013. 1.1

[7] I. Gonzalez-Hernandez, F. M. Palacios, S. S. Cruz, E. S. E. Quesada, and R. L. Leal. Real-time altitude control for a quadrotor helicopter using a super-twisting controller based on high-order sliding mode observer. *International Journal of Advanced Robotic Systems*, 14(1):1729881416687113, 2017. 1.1

[8] A. Levant. Sliding order and sliding accuracy in sliding mode control. *International journal of control*, 58(6):1247–1263, 1993. 3.1.2

[9] A. Levant. Higher-order sliding modes, differentiation and output-feedback control. *International journal of Control*, 76(9-10):924–941, 2003. 3.1.1

[10] J. Li and Y. Li. Dynamic analysis and pid control for a quadrotor. pages 573–578, 2011. 1.1

[11] J. Moreno. Lyapunov function for levant's second order differentiator. *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 6448–6453, 2012. 3.1.1

[12] J. P. Ortiz, L. I. Minchala, and M. J. Reinoso. Nonlinear robust h-infinity pid controller for the multivariable system quadrotor. *IEEE Latin America Transactions*, 14(3):1176–1183, 2016. 1.1

[13] L. Pollini and A. Metrangolo. Simulation and robust backstepping control of a quadrotor aircraft. page 6363, 2008. 1.1

[14] V. Utkin, A. Poznyak, Y. Orlov, and A. Polyakov. Conventional and high order sliding mode control. *Journal of the Franklin Institute*, 357(15):10244–10261, 2020. 1.1