

# Aerial manipulator for high-accuracy street art painting

A report for seminar

Student Name:

**Ghadeer SHAABAN**

**Borhan HLEISS**

Supervisor

**Prof. Ahmad HABLY**

2021 - 2022

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Mathematical Model of DextAir</b>	<b>3</b>
2.1	Simplify of rotation matrix . . . . .	3
2.1.1	euler angles . . . . .	3
2.1.2	relation between angular velocity and derivatives of euler angles .	5
2.1.3	Approximation to simplify rotation matrix . . . . .	5
2.2	Force and moment applied by the spring . . . . .	5
2.2.1	Simplify force and moment applied by the spring . . . . .	6
2.3	Force and the torque generated by the propellers . . . . .	7
2.4	Simplification of the model . . . . .	9
<b>3</b>	<b>Simulation and results</b>	<b>10</b>
3.1	Block diagrams . . . . .	10
3.2	Simulink . . . . .	12
3.3	Simscape multibody . . . . .	12
3.4	Improving the simulink model . . . . .	15
3.4.1	Adding saturation to actuaters . . . . .	15
3.4.2	Add delay to rotors . . . . .	16
3.4.3	Add perturbation on spring . . . . .	17
3.5	Sliding mode control . . . . .	18
<b>4</b>	<b>Trajectory planning</b>	<b>20</b>
<b>5</b>	<b>Conclusion and further studies</b>	<b>22</b>

# List of Figures

1	center-of mass position for RAM system . . . . .	1
2	motors . . . . .	7
3	Dextair I/O . . . . .	10
4	Control input and propeller rotational speeds decoupling . . . . .	11
5	Feedback Linearization . . . . .	11
6	6 SISO controllers . . . . .	11
7	Simulink . . . . .	12
8	Simmechanic model . . . . .	13
9	3D view circle tracking . . . . .	14
10	XZ-plan view circle tracking . . . . .	14
11	3D view Rectangle tracking . . . . .	14
12	XZ-plan view Rectangle tracking . . . . .	14
13	3D view Heart tracking . . . . .	14
14	XZ-plan view Heart tracking . . . . .	14
15	circle tracking after adding saturation . . . . .	15
16	ellipse tracking . . . . .	16
17	rotors rotational speeds using PID . . . . .	16
18	tracking a circle after imposing the delay to the rotors. . . . .	17
19	tracking a circle after imposing the perturbation on spring. . . . .	17
20	tracking a circle using SMC . . . . .	19
21	rotors rotational speeds using SMC . . . . .	19
22	selecting waypoints via Grabit . . . . .	20
23	XZ-plan view flower tracking . . . . .	21
24	3D view flower tracking . . . . .	21

## List of Tables

1	rotation matrices . . . . .	4
2	model parameters . . . . .	12

## **Abstract**

The aim of this work is to design, analyze and test the behavior a control system assigned to a 6-DOF suspended aerial manipulator with a flexible cable. That is the preliminary phase of developing an aerial manipulator that will be used for the purpose of drawing graffiti on a wall. The simulation is built so that the aerial robot follows a trajectory that consists of a set of waypoints that can be modified in the code each time the desired path changes. A simplified demo was made using mathematical equations for the trajectory.

# 1 Introduction

Since the beginning of the dawn of earth, human beings have kept a continuous aim to explore the unknown and go beyond their ability constraints and their degrees of freedom. They have invented rockets to conquer the space, airplanes to fly in the sky, submarine to reach the deepest seas, and the race for making life on earth easier is still going on. One field of interest is developing robots for accomplishing tasks faster than humans, and might save lives and reduce some valuable time (first aid drones, delivery drones, bomb defusing wheeled robots...). When it comes to arts and accurate tasks like painting, this race become more challenging...

Numerous types of robots have been used for artistic purpose using various tools.

Rotorcraft Aerial Manipulator (RAM) system is composed of a Rotorcraft Unmanned Aerial Vehicle (RUAV) and a multi Degree-Of-Freedom (DOF) manipulator, which is designed to fulfill aerial manipulation. The main problems in (RAM) in that moments of inertia and center-ofmass position for RAM system vary when the manipulator moves (Figure 1). In [1], the influence of robot arm motion is treated as perturbations for RUAV in the process of overall system modeling. External force and torque acting on the RUAV body exerted by environment are also considered in system dynamics.

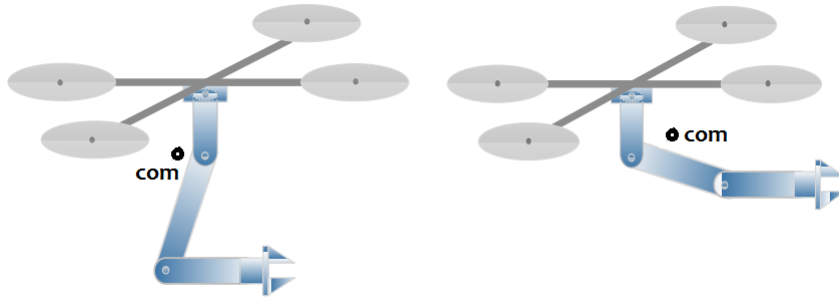


Figure 1: center-of mass position for RAM system

An important factor in this domain is the reachable workspace for the manipulator, the goal is to increase reachable workspace considering desirable factors like weight reduction, inertia and center of mass variation. An interesting work [2] achieves reducing the variation of inertia and center of mass of manipulator.

In [3], a manipulator attached to an unmanned aerial vehicle, where estimating the force acting on the end-effector is important. The work presents a contact force estimation method of an unmanned aerial manipulator, using the dynamic model of the manipulator, his Jacobian matrix and the torques acting in its joints.

In [4], a force controller via energy tanks is implemented for novel applications in aerial contour follow. This control approach allows the aerial vehicle to trace out a boundary whilst in continuous contact with a surface by means of an actively compliant manipulator.

Closed-Loop Inverse Kinematics algorithms has been widely and successfully applied in robot manipulators with structured workspaces, but in aerial manipulators, the lack of structured workspaces is highly demanding for the control algorithm and some extra requirements are needed. Thus, in [5], the standard proportional action in first-order CLIK algorithms is enhanced adding integral actions. Among others benefits, it provides a smoother behaviour needed for smart manipulation, zero steady-state tracking error, rejection to constant disturbances.

In [6], the working principle, the design and the architecture of all components of the cable suspended aerial manipulator were available. The main advantage from developing the suspended aerial manipulator in this paper with the respect to all existing solutions is that the vehicle was able to perform tasks at complex environment while keeping the overall system safe from obstacles. In addition of the carrier example in this paper, the platform developed can be mounted on other carriers like manned aerial vehicles and cranes.

In [7], the motivation for introducing a 2 DOF compliant arm attached to a one meter flexible link, similar to a passive pendulum, was the intention to perform inspection tasks at difficult access areas. These operations were facilitated also by providing visual feedback display.

In [8], shows that Aerial manipulator with elastic suspension can have more energy efficiency than untethered aerial manipulator, in addition of the high accuracy and fast dynamics.

In [9], proposes an algorithm to generate optimal design for omnidirectional aerial vehicles after it have showed the conditions for the omnidirectional-thrust property. Also, this paper proposes a position and orientations tracking nonlinear controller, to demonstrate the lowest possible inputs for the optimized model.

In addition, a lot of works work on kinematic and dynamic model and control of UAV with manipitaor, such as [10], [11]. Robust control is used in [12], where the dynamic coupling from an attached manipulator is treated as disturbance for the UAV. In [13] trajectory tracking control is designed, this is based on Lyapunov theory and Sliding Mode Control (SMC).

For our purpose, which is to draw a street graffiti on a wall, we used DextAir UAV with elastic suspension, since its unique design and most necessarily reachable degrees of freedom fully suits our intentions and match with our desired task. In the following sections, we will present the mathematical model, the assumptions taken into consideration, the block diagram along with the developed controller and feedback and the performed simulations, as well as the trajectory planning.

## 2 Mathematical Model of DextAir

Newton-Euler's equations describing the dynamics of the Drone written in a fixed inertial reference frame  $R_f$  are the following:

$$\begin{bmatrix} mI_{3 \times 3} & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} & \mathbf{I}_f \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{p}} \\ \dot{\omega} \end{bmatrix} + \begin{bmatrix} \mathbf{0}_{3 \times 1} \\ \omega \times \mathbf{I}_f \omega \end{bmatrix} + \begin{bmatrix} mg\mathbf{z} \\ \mathbf{0}_{3 \times 1} \end{bmatrix} - \begin{bmatrix} \mathbf{F}_s \\ \mathbf{N}_s \end{bmatrix} = \begin{bmatrix} \mathbf{F} \\ \mathbf{N} \end{bmatrix} \quad (1)$$

Where:

- $m$  is the total mass of the Drone.
- $g$  is the gravitational acceleration.
- $\mathbf{P}$  the coordinates of the Drone center of gravity G.
- $\mathbf{I}_f$  the inertia matrix of the Drone expressed in the inertial frame.
- $\mathbf{z} = [0 \ 0 \ 1]^T$ .
- $\mathbf{F}_s$  and  $\mathbf{N}_s$  the force and the moment applied by the spring on the Drone.
- $\mathbf{F}$  and  $\mathbf{N}$  the moment applied by the spring on the AWG, F and N the force and the torque generated by the propellers on the Drone.

Let  $\mathbf{R}_{fb}$  be the rotation matrix from the inertial frame  $R_f$  to the body frame  $R_b$ , such that  $v^f = R_{fb}v^b$ . Then  $\mathbf{I}_f$  can be written as:

$$\mathbf{I}_f = \mathbf{R}_{fb} \mathbf{I}_b \mathbf{R}_{fb}^T \quad (2)$$

with  $I_b$  the constant diagonal inertia matrix of the Drone expressed in the body frame  $R_b$ .

### 2.1 Simplify of rotation matrix

#### 2.1.1 euler angles

Euler angles represent three consecutive rotations in the order of  $\psi, \phi, \theta$  so that one coordinate axes system is made to coincide with another system. We suppose these three rotation consequently:

- Angle  $\psi$  - Angle to be rotated about the current frame's Z-axis (about  $Z_f$ ) and we get frame  $R_1$ .
- Angle  $\theta$  - Angle to be rotated about the current frame's Y-axis (about  $Y_1$ ) and we get frame  $R_2$ .
- Angle  $\phi$  - Angle to be rotated about the current frame's X-axis (about  $X_2$ ) and we get frame  $R_b$ .



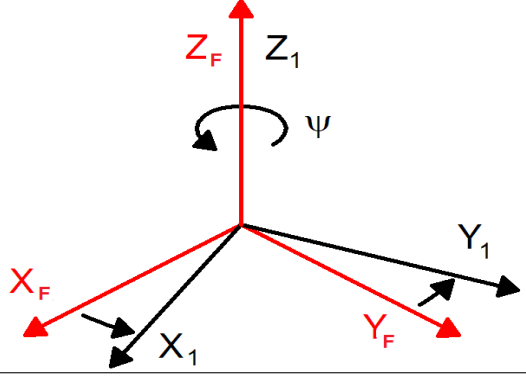
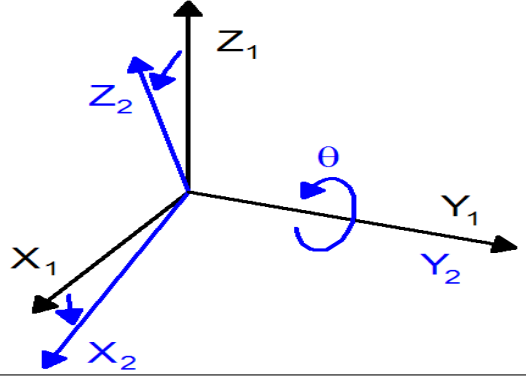
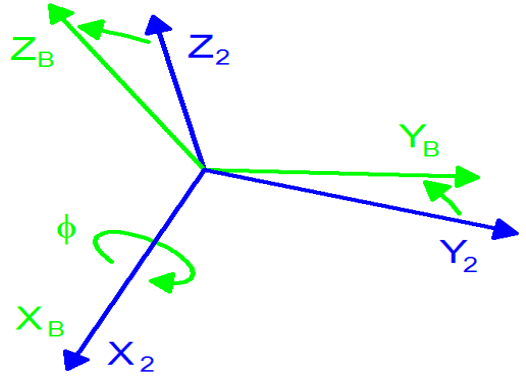
<p>rotation matrix about <math>Z_f</math>:</p> $R_z = \begin{pmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{pmatrix}$	
<p>rotation matrix about <math>Y_1</math>:</p> $R_y = \begin{pmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{pmatrix}$	
<p>rotation matrix about <math>X_2</math>:</p> $R_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{pmatrix}$	

Table 1: rotation matrices

We define rotation matrices as explained in table (1)

And the total rotation matrix is:

$$R_{fb} = R_x \times R_y \times R_z \quad (3)$$

To simplify writing, we put  $Cos := C$  and  $Sin := S$ ,

$$R = \begin{pmatrix} C(\theta)C(\psi) & C(\theta)S(\psi) & -S(\theta) \\ -C(\phi)S(\psi) + S(\phi)S(\theta)C(\psi) & C(\phi)C(\psi) + S(\phi)S(\theta)S(\psi) & S(\phi)C(\theta) \\ S(\phi)S(\psi) + C(\phi)S(\theta)C(\psi) & -S(\phi)C(\psi) + C(\phi)S(\theta)S(\psi) & C(\phi)C(\theta) \end{pmatrix} \quad (4)$$

### 2.1.2 relation between angular velocity and derivatives of euler angles

Frame  $R_b$  become after three consecutive rotations of Frame  $R_f$ , the angular velocity can be written as:

$$\vec{\omega}^B = \begin{pmatrix} \dot{\phi} \\ 0 \\ 0 \end{pmatrix} + R_x \begin{pmatrix} 0 \\ \dot{\theta} \\ 0 \end{pmatrix} + R_x R_y \begin{pmatrix} 0 \\ 0 \\ \dot{\psi} \end{pmatrix}$$

So:

$$\vec{\omega}^B = \begin{pmatrix} \dot{\phi} - \dot{\psi} \sin(\theta) \\ \dot{\psi} \sin(\phi) \cos(\theta) + \dot{\theta} \cos(\phi) \\ \dot{\psi} \cos(\phi) \cos(\theta) - \dot{\theta} \sin(\phi) \end{pmatrix} \quad (5)$$

### 2.1.3 Approximation to simplify rotation matrix

To simplify rotation matrix, we will assume for small angle  $\mu$  the following:

- $\sin(\mu) = 0$
- $\cos(\mu) = 1$

In our case, the control should guarantee zero euler angles all time, so (4) become:

$$R_{fb} = I_{3 \times 3} \quad (6)$$

And after doing the same staregy with the relation between angular velocity and derivatives of euler angles, 5 become:

$$\vec{\omega} = \begin{pmatrix} p \\ q \\ r \end{pmatrix} = \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} \quad (7)$$

## 2.2 Force and moment applied by the spring

The force applied by the spring on the Drone can be written as:

$$\mathbf{F}_s = -k(\mathbf{OP} - l_0 \frac{\mathbf{OP}}{\|\mathbf{OP}\|}) \quad (8)$$

Where:

- $k$  is the spring constant
- $l_0$  the unstretched length of the spring
- $O$  and  $P$  the attachment points of the spring respectively to the carrier and to the body.

Thus:

$$\mathbf{OP} = \mathbf{P} + \Delta \mathbf{R}_{fb} \hat{\mathbf{z}} \quad (9)$$

The moment applied by the spring on the Drone is given by:

$$\mathbf{N}_s = \Delta \mathbf{R}_{fb} \hat{\mathbf{z}} \times \mathbf{F}_s \quad (10)$$

### 2.2.1 Simplify force and moment applied by the spring

From (9) and (6) :

$$\begin{aligned}\mathbf{OP} &= \mathbf{P} + \Delta \mathbf{I}_{3 \times 3} \hat{\mathbf{z}} \\ \mathbf{OP} &= \mathbf{P} + \Delta \hat{\mathbf{z}} \\ \mathbf{OP} &= \begin{bmatrix} x \\ y \\ \hat{z} + \Delta \end{bmatrix}\end{aligned}\tag{11}$$

We use (11) in (8), so the force applied by spring is:

$$\mathbf{F}_s = \begin{bmatrix} f_{sx} \\ f_{sy} \\ f_{sz} \end{bmatrix} = -k \left( 1 - \frac{l_0}{\sqrt{x^2 + y^2 + (z + \Delta)^2}} \right) \begin{bmatrix} x \\ y \\ z + \Delta \end{bmatrix}\tag{12}$$

We use (11) in (9), so the moment applied by spring is:

$$\mathbf{N}_s = \begin{bmatrix} N_{sx} \\ N_{sy} \\ N_{sz} \end{bmatrix} = -k\Delta \left( 1 - \frac{l_0}{\sqrt{x^2 + y^2 + (z + \Delta)^2}} \right) \begin{bmatrix} -y \\ x \\ 0 \end{bmatrix}\tag{13}$$

### 2.3 Force and the torque generated by the propellers

Let  $\omega_i$  the rotational speed of the  $i$ th propulsion unit. We define vector  $\mathbf{w}_2$  of signed squared propeller rotational speeds as follow:

$$\mathbf{w}_2 = [\omega_1|\omega_1| \ \omega_2|\omega_2| \ \omega_3|\omega_3| \ \omega_4|\omega_4| \ \omega_5|\omega_5| \ \omega_6|\omega_6|]^T \quad (14)$$

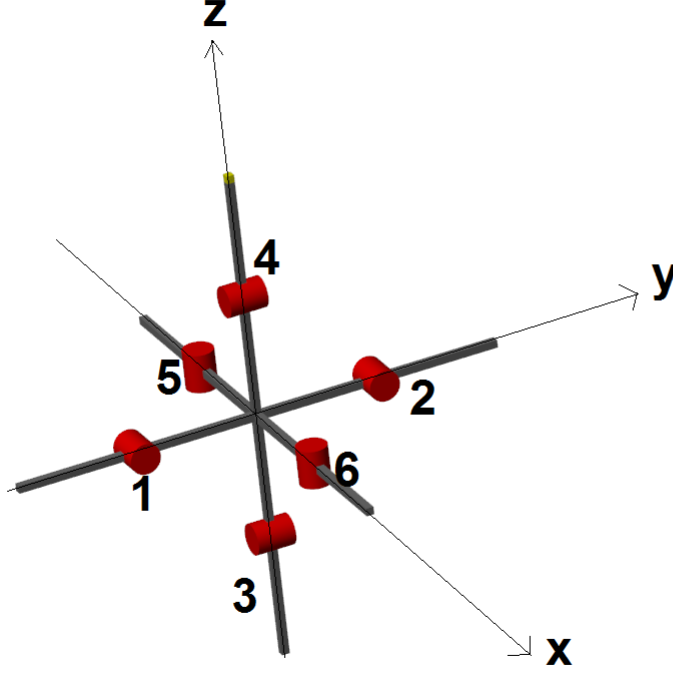


Figure 2: motors

Let  $\mathbf{A}_b$  be the matrix that maps the vector  $\mathbf{w}_2$  the wrench they apply to the Drone:

$$\begin{bmatrix} \mathbf{F}^b \\ \mathbf{N}^b \end{bmatrix} = \mathbf{A}_b \mathbf{w}_2 \quad (15)$$

$$\mathbf{A}_b = a \begin{bmatrix} \mathbf{u}_1^b & \mathbf{u}_2^b & . & . & . & \mathbf{u}_6^b \\ \mathbf{GB}_1^b \times \mathbf{u}_1^b & \mathbf{GB}_2^b \times \mathbf{u}_2^b & . & . & . & \mathbf{GB}_6^b \times \mathbf{u}_6^b \end{bmatrix}$$

Where

- $a$  the thrust coefficient
- $\mathbf{u}_i$  the axis of the  $i$ -th propeller
- $B_i$  the center of the  $i$ -th propeller

We have:

$$\mathbf{u}_1 = \mathbf{u}_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad \mathbf{u}_3 = \mathbf{u}_4 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \quad \mathbf{u}_5 = \mathbf{u}_6 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad (16)$$

$$\begin{aligned} \vec{G}_{b1} &= -L\vec{y} & \vec{G}_{b2} &= L\vec{y} \\ \vec{G}_{b3} &= -L\vec{z} & \vec{G}_{b4} &= L\vec{z} \\ \vec{G}_{b5} &= -L\vec{x} & \vec{G}_{b6} &= L\vec{x} \end{aligned} \quad (17)$$

From (16) and (17) we have:

$$\begin{aligned} \vec{G}_{b1} \times \vec{u}_1 &= -L\vec{y} \times \vec{x} = L\vec{z} \\ \vec{G}_{b2} \times \vec{u}_2 &= L\vec{y} \times \vec{x} = -L\vec{z} \\ \vec{G}_{b3} \times \vec{u}_3 &= -L\vec{z} \times \vec{y} = L\vec{x} \\ \vec{G}_{b4} \times \vec{u}_4 &= L\vec{z} \times \vec{y} = -L\vec{x} \\ \vec{G}_{b5} \times \vec{u}_5 &= -L\vec{x} \times \vec{z} = L\vec{y} \\ \vec{G}_{b6} \times \vec{u}_6 &= L\vec{x} \times \vec{z} = -L\vec{y} \end{aligned} \quad (18)$$

From (17) and (18):

$$\mathbf{A}_b = a \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & L & -L & 0 & 0 \\ 0 & 0 & 0 & 0 & L & -L \\ L & -L & 0 & 0 & 0 & 0 \end{bmatrix} \quad (19)$$

Let define  $f_i$  as follow:

$$f_i = \omega_i |\omega_i| \quad (20)$$

From (19), (15) and (20) :

$$\begin{bmatrix} \mathbf{F}^b \\ \mathbf{N}^b \end{bmatrix} = a \begin{bmatrix} f_1 + f_2 \\ f_3 + f_4 \\ f_5 + f_6 \\ L(f_3 - f_4) \\ L(f_5 - f_6) \\ L(f_1 - f_2) \end{bmatrix} \quad (21)$$

And we can write  $F$  and  $N$  in reference frame :

$$\begin{aligned} \mathbf{F} &= \mathbf{R}_{fb} \mathbf{F}^b \\ \mathbf{N} &= \mathbf{R}_{fb} \mathbf{N}^b \end{aligned}$$

From (21):

$$\begin{bmatrix} \mathbf{F} \\ \mathbf{N} \end{bmatrix} = a \mathbf{R}_{fb} \begin{bmatrix} f_1 + f_2 \\ f_3 + f_4 \\ f_5 + f_6 \\ L(f_3 - f_4) \\ L(f_5 - f_6) \\ L(f_1 - f_2) \end{bmatrix} \quad (22)$$

After simplification, we apply (6) on (22):

$$\begin{bmatrix} \mathbf{F} \\ \mathbf{N} \end{bmatrix} = a \begin{bmatrix} f_1 + f_2 \\ f_3 + f_4 \\ f_5 + f_6 \\ L(f_3 - f_4) \\ L(f_5 - f_6) \\ L(f_1 - f_2) \end{bmatrix} \quad (23)$$

## 2.4 Simplification of the model

We recall equation (1)

$$\begin{bmatrix} mI_{3 \times 3} & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} & \mathbf{I}_f \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{p}} \\ \dot{\omega} \end{bmatrix} + \begin{bmatrix} \mathbf{0}_{3 \times 1} \\ \omega \times \mathbf{I}_f \omega \end{bmatrix} + \begin{bmatrix} mg\mathbf{z} \\ \mathbf{0}_{3 \times 1} \end{bmatrix} - \begin{bmatrix} \mathbf{F}_s \\ \mathbf{N}_s \end{bmatrix} = \begin{bmatrix} \mathbf{F} \\ \mathbf{N} \end{bmatrix}$$

From (6) and (2) :

$$\mathbf{I}_f = \mathbf{R}_{fb} \mathbf{I}_b \mathbf{R}_{fb}^T \quad (24)$$

We put (7), (12), (23), and (24) in (1):

$$\begin{aligned} m\ddot{x} - f_{sx} &= a(f_1 + f_2) \\ m\ddot{y} - f_{sy} &= a(f_3 + f_4) \\ m\ddot{z} + mg - f_{sz} &= a(f_5 + f_6) \\ I_x\ddot{\phi} - N_{s\phi} &= aL(f_3 - f_4) \\ I_y\ddot{\theta} - N_{s\theta} &= aL(f_5 - f_6) \\ I_z\ddot{\psi} - N_s\psi &= aL(f_1 - f_2) \end{aligned} \quad (25)$$

We define new variables:

$$\begin{aligned} u_x &= a(f_1 + f_2) + f_{sx} \\ u_y &= a(f_3 + f_4) + f_{sy} \\ u_z &= a(f_5 + f_6) + f_{sz} - mg \\ u_\phi &= aL(f_3 - f_4) + N_{s\phi} \\ u_\theta &= aL(f_5 - f_6) + N_{s\theta} \\ u_\psi &= aL(f_1 - f_2) + N_s\psi \end{aligned} \quad (26)$$

So the system (27) :

$$\begin{aligned} m\ddot{x} &= u_x \\ m\ddot{y} &= u_y \\ m\ddot{z} &= u_z \\ I_x\ddot{\phi} &= u_\phi \\ I_y\ddot{\theta} &= u_\theta \\ I_z\ddot{\psi} &= u_\psi \end{aligned} \quad (27)$$

Thus, our linearized system is 6 single input single output systems.

### 3 Simulation and results

#### 3.1 Block diagrams

In Order to simplify the simulation and control procedural, we build block diagrams for every sigle part of the system.

Propeller rotational speeds are the input for Dextair system. Position and attitude are the output. External force and moment applied by the spring (Figure 3).

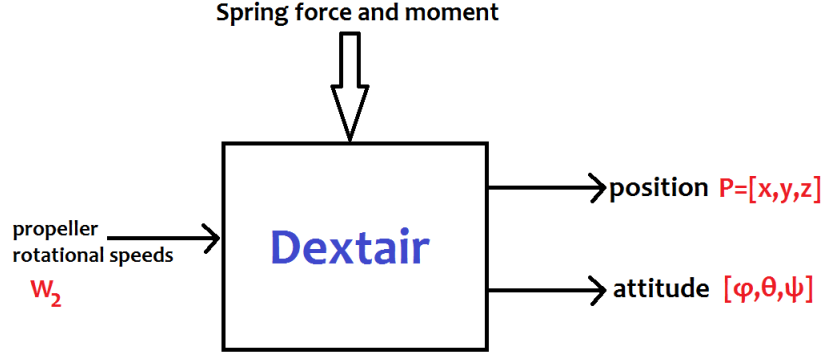


Figure 3: Dextair I/O

We recall equations (20) and (26):

$$\begin{aligned}
 f_i &= \omega_i |\omega_i| \\
 u_x &= a(f_1 + f_2) + f_{sx} \\
 u_y &= a(f_3 + f_4) + f_{sy} \\
 u_z &= a(f_5 + f_6) + f_{sz} - mg \\
 u_\phi &= aL(f_3 - f_4) + N_{s\phi} \\
 u_\theta &= aL(f_5 - f_6) + N_{s\theta} \\
 u_\psi &= aL(f_1 - f_2) + N_{s\psi}
 \end{aligned}$$

We define new variables:

$$\begin{aligned}
 u_{1x} &= a(f_1 + f_2) \\
 u_{1y} &= a(f_3 + f_4) \\
 u_{1z} &= a(f_5 + f_6) \\
 u_{1\phi} &= aL(f_3 - f_4) \\
 u_{1\theta} &= aL(f_5 - f_6) \\
 u_{1\psi} &= aL(f_1 - f_2)
 \end{aligned} \tag{28}$$

So:

$$\begin{aligned}
 u_x &= u_{1x} + f_{sx} \\
 u_y &= u_{1y} + f_{sy} \\
 u_z &= u_{1z} + f_{sz} - mg \\
 u_\phi &= u_{1\phi} + N_{s\phi} \\
 u_\theta &= u_{1\theta} + N_{s\theta} \\
 u_\psi &= u_{1\psi} + N_{s\psi}
 \end{aligned} \tag{29}$$

Equations (20) and (28), define decoupling between control input and propeller rotational speeds (Figure 4). Equations (29) define feedback linearization (Figure 5), where forces



Figure 4: Control input and propeller rotational speeds decoupling

and moments applied by the spring are estimated using system output (position and attitude).

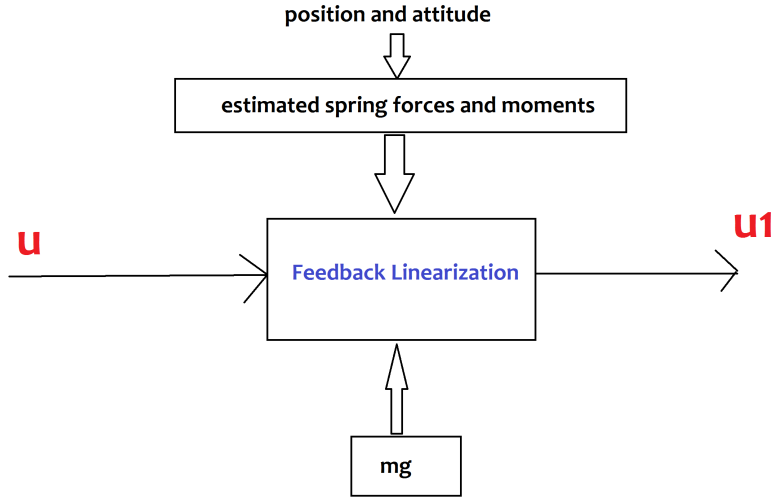


Figure 5: Feedback Linearization

And finally we obtain  $\mathbf{u} = [u_x \ u_y \ u_z \ u_\phi \ u_\theta \ u_\psi]$  from the controller of 6 SISO systems as equations (27), block diagram (Figure 6).



Figure 6: 6 SISO controllers



### 3.2 Simulink

Simulation was done using Simulink-Matlab, (Figure 7)

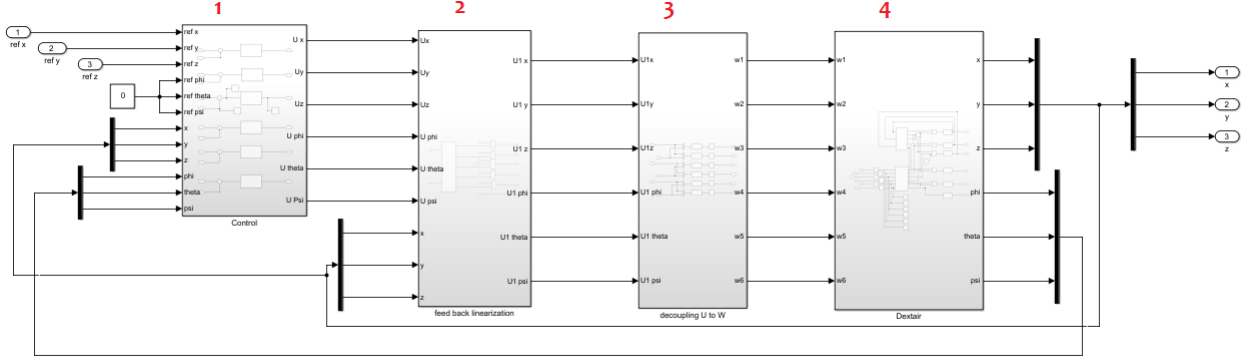


Figure 7: Simulink

1. Controllers: where we implement 6 separate controllers.
2. Feedback Linearization, where we implement equations (29).
3. Decoupling, where we implement equations (20) and (28).
4. Dextair System, this block define Dextair system.

To do simulations, we have to use numerical values for system parameters, we use numerical values as [14], as shown in (Table 2).

Parameter	Description	Value
$m$	Total mass	$2.07kg$
$I_x, I_y, I_z$	Moments of inertia	$5.04 \times 10^{-2}kgm^2$
$k$	Spring stiffness	$22.5kgm^{-1}$
$L$	Propeller axis to CoG distance	$0.16m$
$\Delta$	Spring to CoG distance	$0.32m$
$a$	Thrust coefficient	$1.7\mu Nrad^{-2}s^2$

Table 2: model parameters

### 3.3 Simscape multibody

For the purpose of getting a more realistic projection to reality, we will use simulation tool called Simscape multibody, which will allows us to design and test our control system on the actual shape of DextAir based on the suitable assumptions and configurations. Simscape multibody is the best option, since it's a tool integrated and compatible with

MATLAB, which is our desired simulation environment, and we will have the chance to visualize the influence of our control system on an actual shape rather than plots and graphs only. So we build Dextair system using 3D cad tool embedded with simmechanics, we build the model so that the model parameters are identical to values in (Table 2). The 3D model of dextair as shown in (Figure 8).

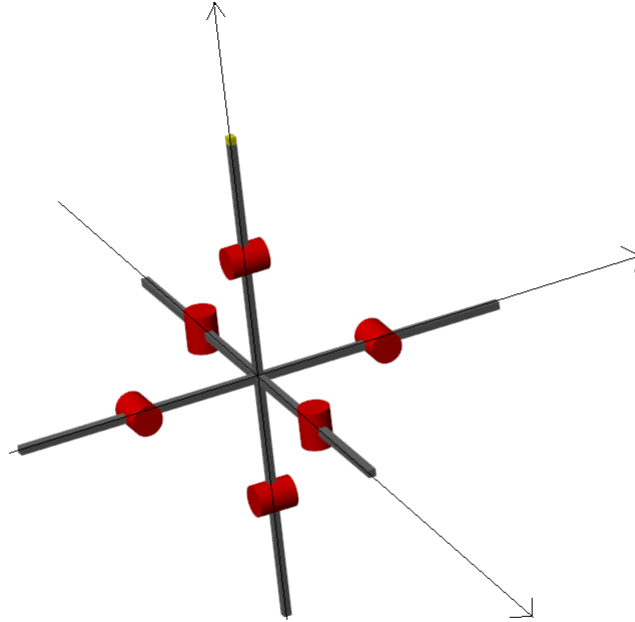


Figure 8: Simmechanic model

It was difficult to build spring model using simmechanics, so in this section we will simulate the system and control without spring (set  $k = 0$  when  $k$  appears in simulink blocks).

Simulation results for different trajectories:

- Circle (Figures 9, 10).
- Rectangle (Figures 11, 12).
- Heart (Figures 13, 14).

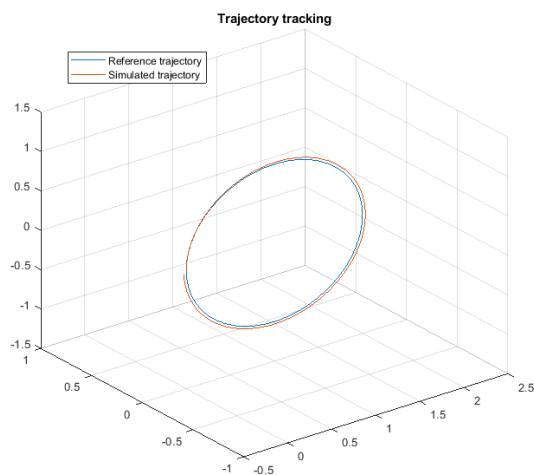


Figure 9: 3D view circle tracking

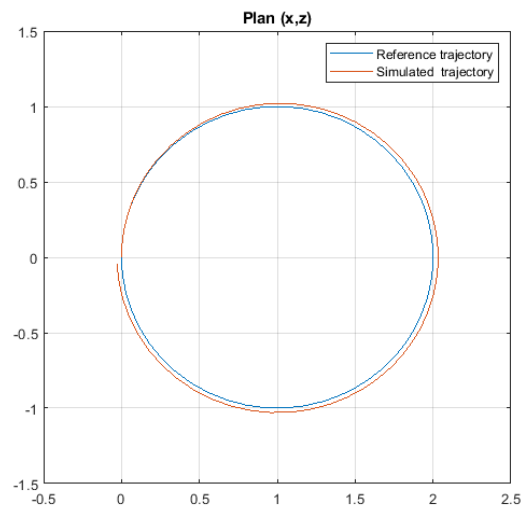


Figure 10: XZ-plan view circle tracking

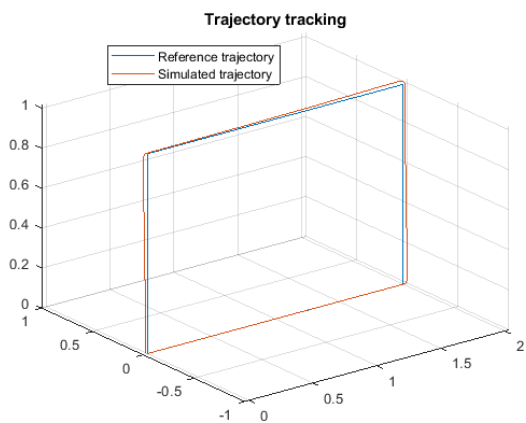


Figure 11: 3D view Rectangle tracking

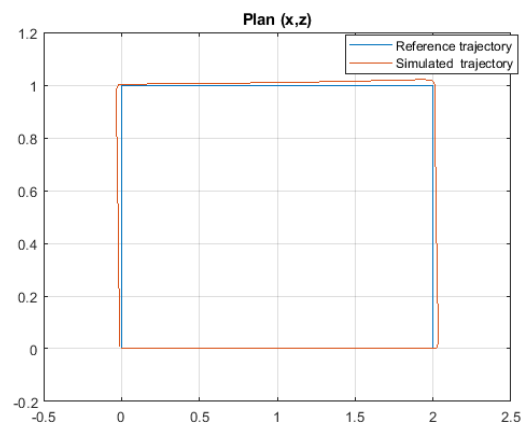


Figure 12: XZ-plan view Rectangle tracking

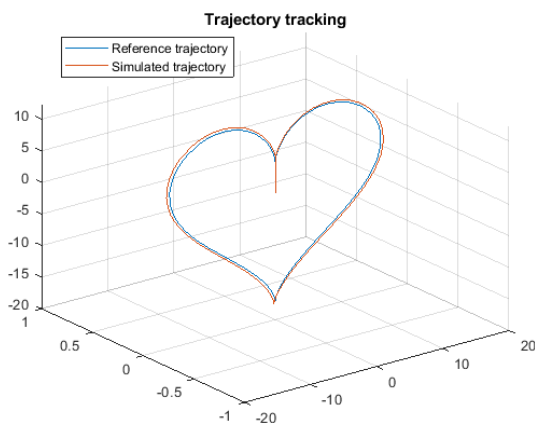


Figure 13: 3D view Heart tracking

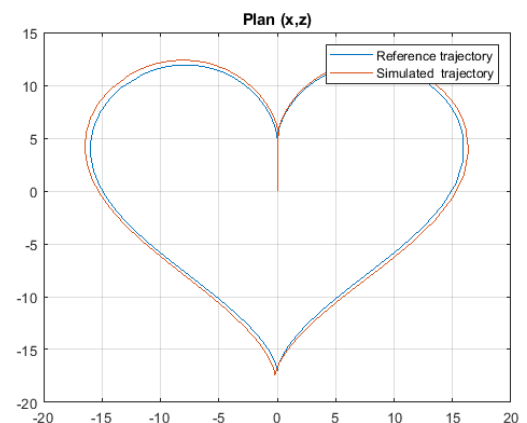


Figure 14: XZ-plan view Heart tracking

### 3.4 Improving the simulink model

In this section, we will do the simulation under spring effect (put the real value of  $k$  as table 2), and put simplified model of dextair in block 4 in simulink (refer to Figure 7).

#### 3.4.1 Adding saturation to actuators

We consider that we are using actuators as [14], so there are saturation on actuators  $\omega_{max} = 25000rpm = 2618rad.s^{-1}$ . The result for circle tracking after adding saturation on actuators is shown in (Figure15). This result was expected, because the high value of

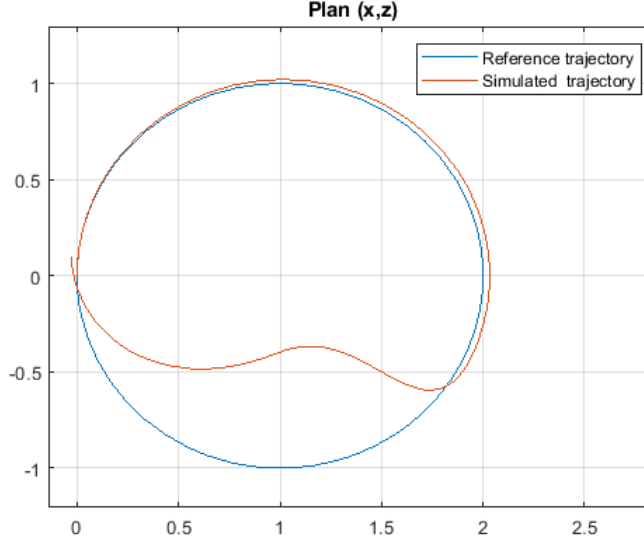


Figure 15: circle tracking after adding saturation

spring stiffness, so dextair should paint in relatively small area, and in order to paint in other area, we change the position of the carrier. It is important to test each trajectory on simulation before go to real dextair. If we decrease the vertical motion, for example  $z \in [-0.1, 0.1]$  trajectory tracking for ellipse, see figure 16. And for input signals (rotors rotational speeds) see Figure 17, we can see that the controller has respected the saturation constraints.

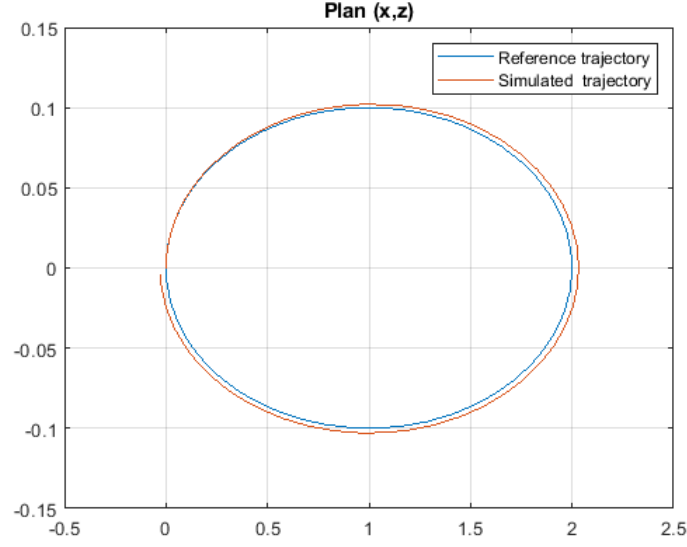


Figure 16: ellipse tracking

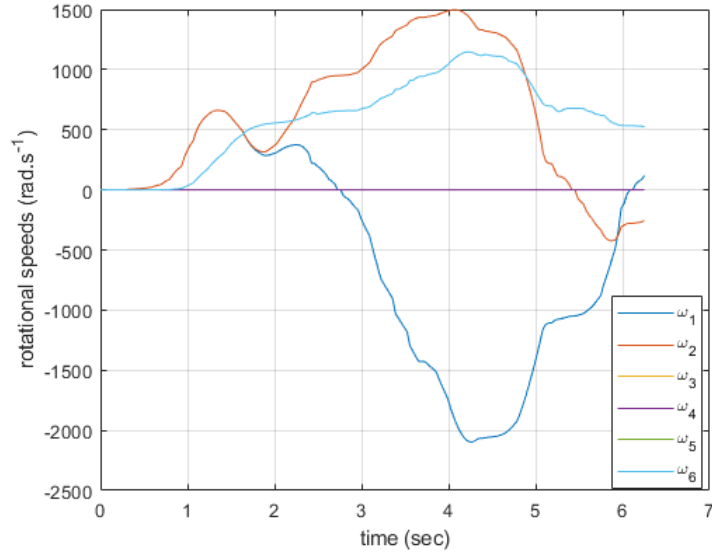


Figure 17: rotors rotational speeds using PID

### 3.4.2 Add delay to rotors

The 12 rotor velocity loops are modeled as a first-order system with a 60 ms time constant, so the transfer function is like  $H_{rotor} = \frac{1}{0.06s+1}$ . After applying this delay, we got the result shown in (Figure 18). We can see that adding a delay to the rotors has caused noticed error in tracking the desired shape. Based on this result, we started to consider a more robust control algorithm to apply on Dextair.

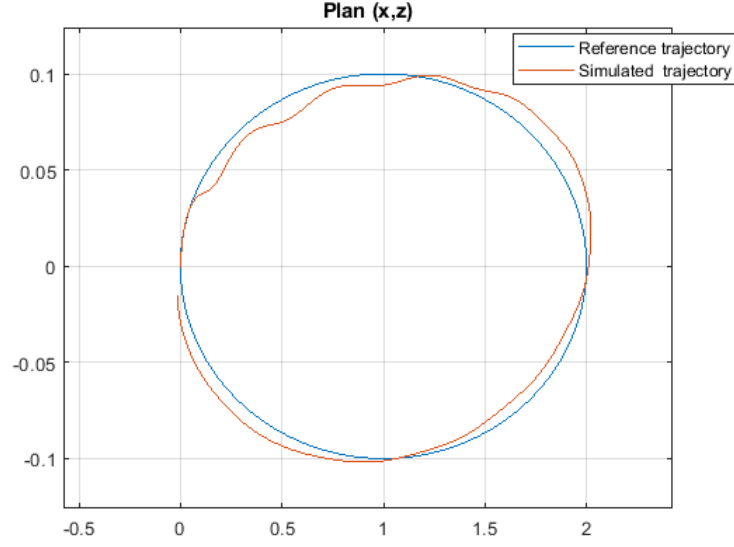


Figure 18: tracking a circle after imposing the delay to the rotors.

### 3.4.3 Add perturbation on spring

We assume to have a perturbation signal on the elastic suspension in  $f_x$  (X-axis direction) in the form of a sin wave, whose amplitude and frequency are  $10N, 10Hz$  respectively.

$$f_{x \text{ perturbed}} = f_x + 10\sin(20\pi t)$$

After applying this perturbation, we got the result shown in (Figure 19). We can see also that adding a perturbation to the spring has caused noticed error in tracking the desired shape. The solution is discussed in the next section.

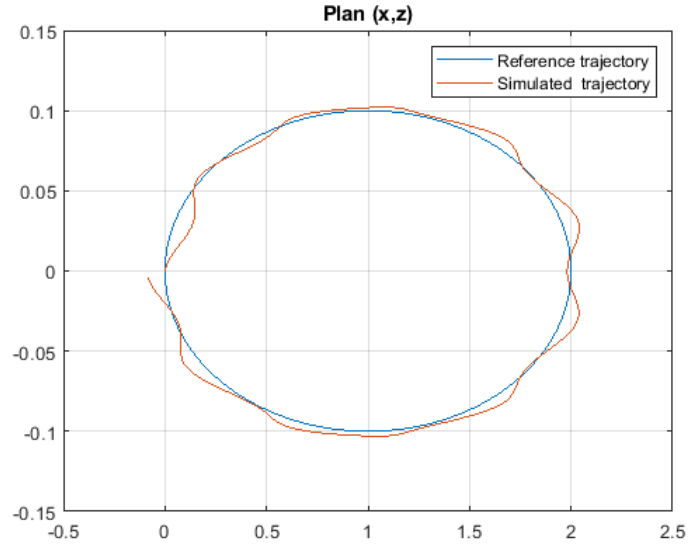


Figure 19: tracking a circle after imposing the perturbation on spring.

### 3.5 Sliding mode control

In control systems, sliding mode control (SMC) is a nonlinear control method that alters the dynamics of a nonlinear system by applying a discontinuous control signal (or more rigorously, a set-valued control signal) that forces the system to "slide" along a cross-section of the system's normal behavior.

To design SMC, there are two steps, first we define a sliding surface  $S$  based on the control objectives, second we design the discontinuous control signal that impose  $S$  to be equal to zero.

We define the sliding surface as follows:

$$S(x, t) = \dot{x}_e + \gamma x_e$$

Where:

- $S$ : the sliding surface.
- $x_e = x_{ref} - x$  : is the error.

When reaching  $S = 0$  so:

$$\dot{x}_e + \gamma x_e = 0$$

has a solution of the form:

$$x_e(t) = x_e(t_0) \exp(-\gamma(t - t_0))$$

where  $t_0$  the time when  $S$  reach zero. In this case time response is to be  $tr = \frac{3}{\gamma}$ . So the remaining is to reach  $S = 0$  in finite time. We define Lyapunov function  $V(S) = \frac{1}{2}S^2$ ,  $\dot{V}(S) = S\dot{S}$ . If we put  $\dot{S} = -\eta \text{sgn}(S)$  so :  $\dot{V}(S) = -\eta|S|$  where  $\eta$  is positive integer, so Lyapunov condition is satisfied. We return to equations (27), the six subsystems are SISO with the form  $\ddot{x} = \text{const}/s^2 u$ .

$$\dot{S} = -\eta \text{sgn}(S)$$

$$S = \dot{x}_e + \gamma x_e$$

$$S = -\dot{x} + \gamma(x_{ref} - x)$$

$$\dot{S} = -\ddot{x} + \gamma\dot{x}$$

but our systems are with the form  $\ddot{x} = \text{const}u$ , so

$$\dot{S} = -\text{const}u + \gamma\dot{x}$$

So the final control law:

$$u = \frac{1}{\text{const}}(\eta \text{sgn}(S) + \gamma\dot{x})$$

But this method is still subjected to chattering problem. It turns out that, there is a more efficient solution to avoid chattering is by using sliding mode control super twisting, where the control input became as follows [15]:

$$u = \lambda \text{sgn}(S) \sqrt{|S|} + W \int_0^t \text{sgn}(t) dt$$

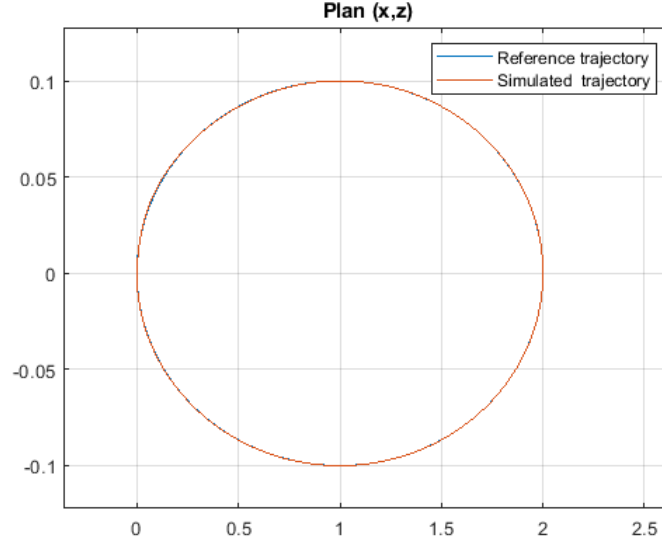


Figure 20: tracking a circle using SMC

Where  $\lambda > 4L$ ,  $W > \frac{1}{2}\sqrt{L}$ , and  $L$  constant to be tuned.

After tuning, we applied the following values:  $\gamma = 3$ ,  $L = 3$ ,  $\lambda = 12$ ,  $W = 0.886$ . In simulation, SMC is applied on  $X, Y, Z$  subsystems, and we remain using PID for  $\phi, \theta, \psi$ . The result is shown in (Figure 20). Clearly, using sliding mode control super twisting showed far more better performance than PID.

And for input signals (rotors rotational speeds) see Figure 21, we can see that the controller has respected the saturation constraints.

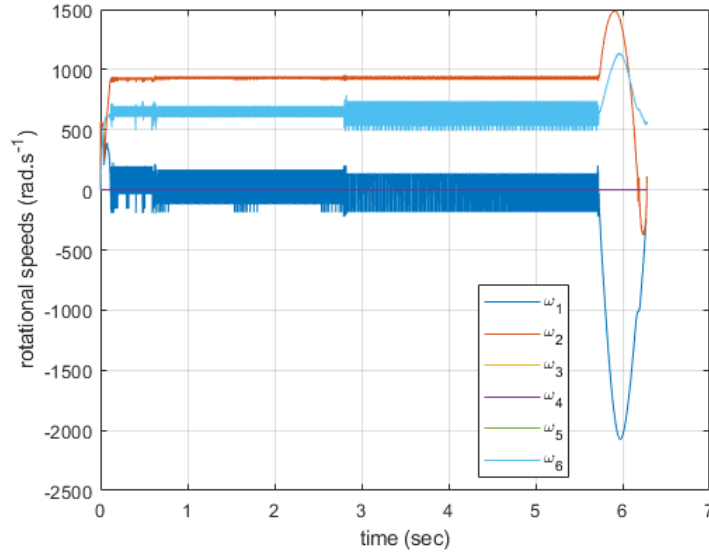


Figure 21: rotors rotational speeds using SMC



## 4 Trajectory planning

Now that we are aiming to improve the design and stability of our system, we need to develop a strategy for generating a trajectory to our DextAir. The actual aim beyond this project is that the DextAir be able to follow whatever path we want while it maintains a good performance, stability, smoothness, the public safety and its own safety as well. Since DextAir will be used for the painting, most of its movement will be mainly in the 2D plan parallel to wall desired to draw on. Our strategy consists of setting coordinates for multiple points in the space for the model to follow in order, and see what response it will give us. The trajectory planning will be also done in MATLAB. The trajectory planning for our DextAir concept was to input a set of points in the MATLAB called Waypoints. The best definition that could be assigned to waypoints is that they represent the position of DextAir at each time. These waypoints are considered to have 4 coordinates which are  $x, y, z$  and the time  $t$ . It was important to introduce time as a coordinate for two main reasons : the first is that we should prioritize our navigation with DextAir, as in the order of points we want to reach first. The second important thing is to keep in mind, if the distance that should be travelled from one point to the next with respect to time, if it's within the capability of our controller and our model.

We used a tool in matlab called **Grabit**, for the purpose of selecting the waypoints of the desired trajectory. Basicly the drone will draw the graffiti in a parallel plane to its movement. Thus the waypoints will be selected from the drawing. The Grabit tool will import the picture that we want to draw, and then it will allow us to select the trajectory points all along this picture. The following example (Figure 22) will illustrate the use of Grabit tool for the purpose of drawing a flower.

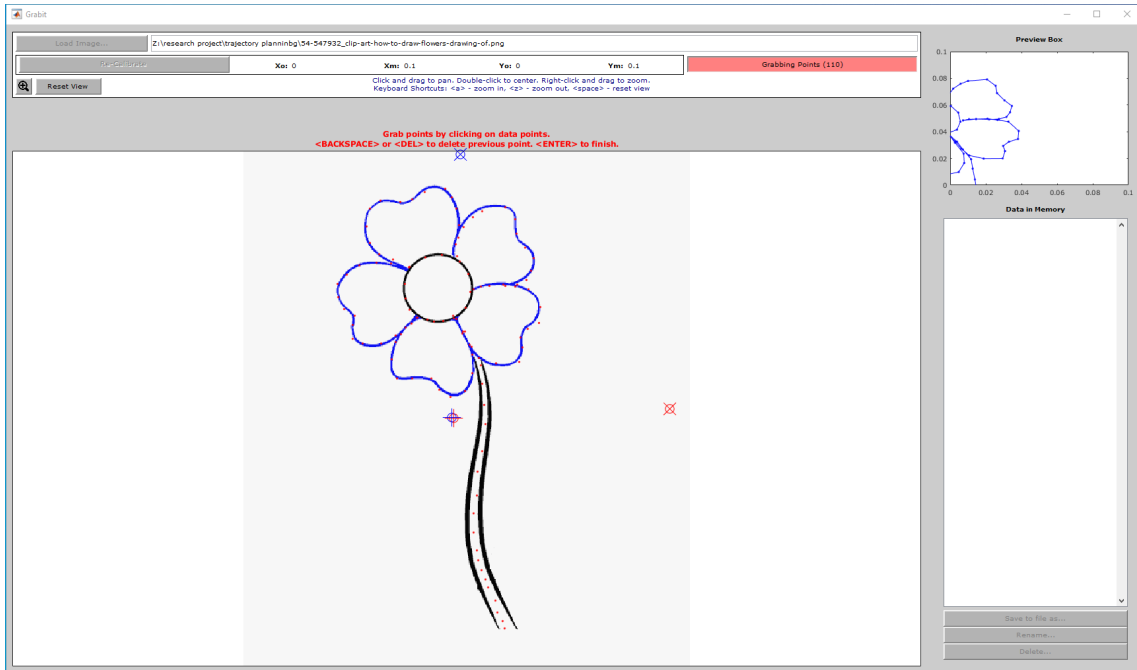


Figure 22: selecting waypoints via Grabit

After importing the waypoints to the workspace, we applied Dextair tracking to follow

these points, the results are shown in figures (23) (24).

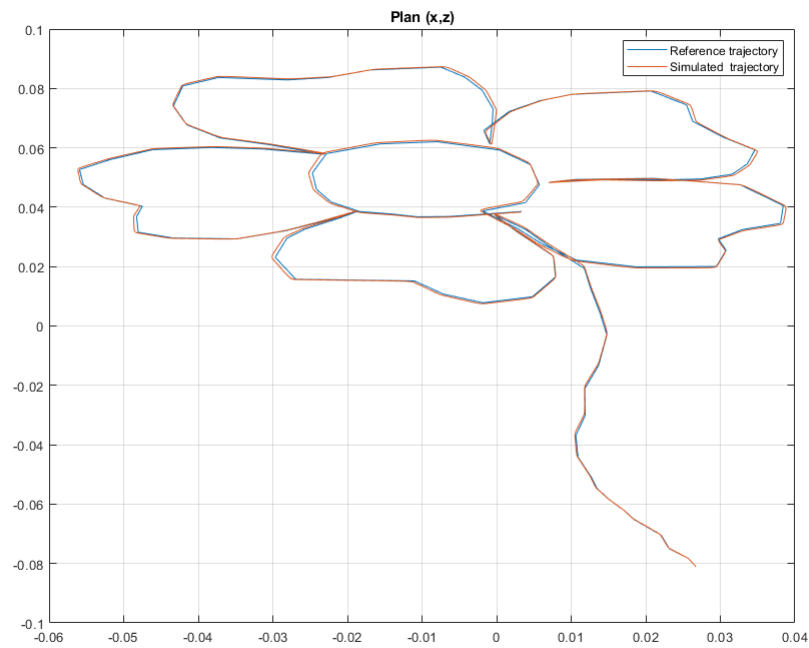


Figure 23: XZ-plan view flower tracking

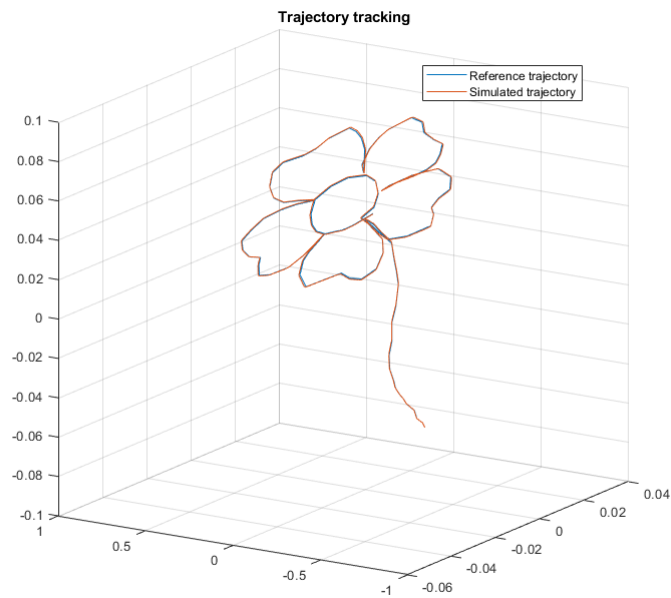


Figure 24: 3D view flower tracking

## 5 Conclusion and further studies

Finally, in this research project, we have modeled, controlled, and planned the trajectory of DextAir drone. We have also applied two types of control on our model (PID and Sliding Mode Control Supertwisting) and showed the the strengths and the weaknesses of each type with the respect to saturations constraints and external factors, and the results achieved were outstanding.

For further development of the overall DextAir project, we propose a further study project, which is to implement a computer vision or image processing software, whose purpose is to detect the lines and shapes in future graffitis to be drawn, and generate a set of waypoints automatically for DextAir to follow.

## References

- [1] Xiangdong Meng, Yuqing He, Feng Gu, Qi Li, and Jianda Han. Dynamics modeling and simulation analysis for rotorcraft aerial manipulator system. In *2017 36th Chinese Control Conference (CCC)*, pages 1156–1161, 2017.
- [2] Nipuna Wijayathunga, Thilina Dulantha Lalitharatne, Damith Suresh Chathuranga, and Buddhika Jayasekara. Development of a robotic manipulator to be used in multi-rotor aerial vehicle. In *2019 Moratuwa Engineering Research Conference (MERCon)*, pages 638–643, 2019.
- [3] Kresimir Turkovic, Marko Car, and Marko Orsag. End-effector force estimation method for an unmanned aerial manipulator. In *2019 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED UAS)*, pages 96–99, 2019.
- [4] Salua Hamaza, Ioannis Georgilas, and Thomas Richardson. 2d contour following with an unmanned aerial manipulator: Towards tactile-based aerial navigation. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3664–3669, 2019.
- [5] M. I. Sánchez, J. Á. Acosta, and A. Ollero. Integral action in first-order closed-loop inverse kinematics. application to aerial manipulators. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5297–5302, 2015.
- [6] Yuri S. Sarkisov, Min Jun Kim, Davide Bicego, Dzmitry Tsetserukou, Christian Ott, Antonio Franchi, and Konstantin Kondak. Development of sam: cable-suspended aerial manipulator. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 5323–5329, 2019.
- [7] A. Suarez, P. Sanchez-Cuevas, M. Fernandez, M. Perez, Guillermo Heredia, and Anibal Ollero. Lightweight and compliant long reach aerial manipulator for inspection operations. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6746–6752, 2018.
- [8] Arda Yiğit, Miguel Arpa Perozo, Loïc Cuvillon, Sylvain Durand, and Jacques Gangloff. Novel omnidirectional aerial manipulator with elastic suspension: Dynamic control and experimental performance assessment. *IEEE Robotics and Automation Letters*, 6(2):612–619, 2021.
- [9] Marco Tognon and Antonio Franchi. Omnidirectional aerial vehicles with unidirectional thrusters: Theory, optimal design, and control. *IEEE Robotics and Automation Letters*, 3(3):2277–2282, 2018.
- [10] Yidong Chen, Xiaofan Zhang, Chunhui Gu, Qingchun Tang, and Xiaole Li. Control for aerial transportation using a quadrotor with a multi-dof manipulator. In *2017 4th International Conference on Information Science and Control Engineering (ICISCE)*, pages 916–920, 2017.

- [11] D. Wuthier, D. Kominiak, C. Kanellakis, G. Andrikopoulos, M. Fumagalli, G. Schipper, and G. Nikolakopoulos. On the design, modeling and control of a novel compact aerial manipulator. In *2016 24th Mediterranean Conference on Control and Automation (MED)*, pages 665–670, 2016.
- [12] Guangyu Zhang, Yuqing He, Bo Dai, Feng Gu, Jianda Han, and Guangjun Liu. Robust control of an aerial manipulator based on a variable inertia parameters model. *IEEE Transactions on Industrial Electronics*, 67(11):9515–9525, 2020.
- [13] Adriana Guayasamín, Paulo Leica, Marco Herrera, and Oscar Camacho. Trajectory tracking control for aerial manipulator based on lyapunov and sliding mode control. In *2018 International Conference on Information Systems and Computer Science (INCISCOS)*, pages 36–41, 2018.
- [14] Arda Yiğit, Gustave Grappe, Loïc Cuvillon, Sylvain Durand, and Jacques Gangloff. Preliminary study of an aerial manipulator with elastic suspension. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4287–4293, 2020.
- [15] Dennis Driggers. A survey of some sliding mode control designs. *EE691*, 2006.