Testing Board for Quadcoptor Parameters Visualization and Stability Analysis using Ultrasonic and Gyroscope Detection

Supervised by

Dr. Yassine SALIH-ALJ

Edited by

Idriss Sefrioui
Nadime Lhassani
Houssam Saber

Spring 2023
Al Akhawayn University in Ifrane

Contents

List of Figures

1 Introduction

Quadcopter and Aircraft applications in general require a lot of precision to ensure that the desired position and velocity are met. For even with very advanced implementations of control systems and accounting for nonlinear dyanamics and noise through various control approaches such as estimations through Linear or Gaussian Quadratic Estimations. Which is why it is necessary to include an independent testing mechanism where we can analyze the various parameters that a portable aircraft designer is interested in, such as:

- Positioning
- Velocity
- Acceleration
- Orientation

To test some of these parameters, a testing bench capable of measuring the parameters using sensors and displaying them in a user-friendly interface,for the aircraft designer would be a good starting point.

2 System Components

The system will be composed of the following:

- **Microcontroller:** The microcontroller will be responsible for taking the measurements from the sensors and performing the necessary calculations and sending the results to the user interface.
- **Ultrasound Sensor:** The ultrasound sensor will be used to measure the distance between the sensor and the Various objects in the environment.
- **A Bluetooth Module:** The Bluetooth module will be used to send the data from the microcontroller to the user interface.
- **User Interface:** A Linux-based system capable of displaying the data received from the microcontroller and displaying it in a user-friendly interface via a programmed GUI.
- **Power Supply:** A power supply capable of providing the necessary power to the system.
- **Gyroscope:** A gyroscope will be used to measure the orientation of the system.

The following Table will specify the choice of each component.

Table 1: System Components

Component	Choice	Purchase Link
Microcontroller	STM32F401RE	Here
Ultrasound Sensor	VL53L1X Time-of-Flight	Here
Bluetooth Module	HM-10	Here
Power Supply	5V 2A Power Supply	Already Available
Gyroscope	MPU-6050	Here

3 System Architecture

The system bench will be composed as follows:

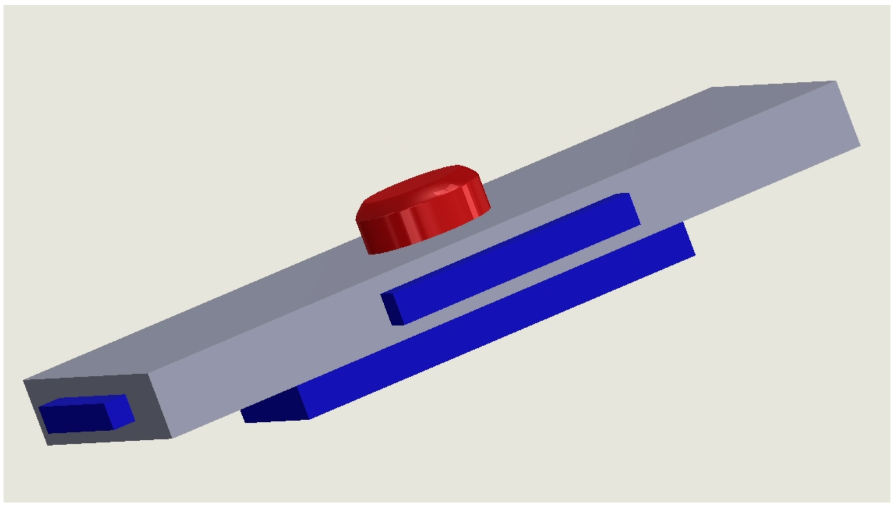

Figure 3.1: System Architecture

Where the microcontroller will be stored inside the box of the system and linked to the bluetooth module and the ultrasound sensors, which appear as blue boxes in Figure 3.2, with the Gyroscope appearing with a Red Color.
The mounting of the circuit will be done as follows:

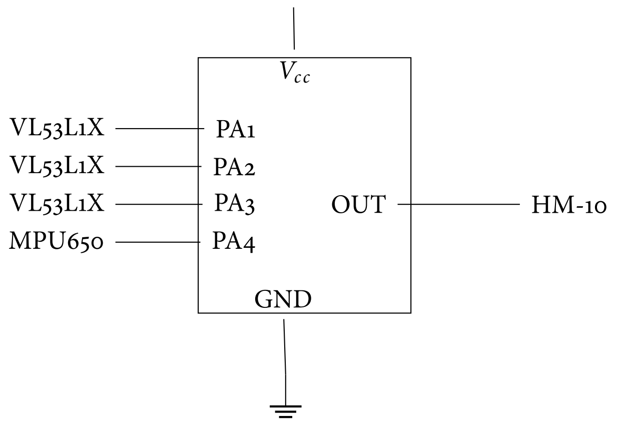

Figure 3.2: Microcontroller Circuit

Only the pins that are used in the circuit have been shown in the Figure.

4 System Implementation

4.1 User Interface

The user interface will be implemented using Python and the PyQt5 Library, which is a Python binding for the Qt5 GUI Library. The user interface will be composed of the following:

- **Main Window:** The main window will be the first window that the user will see when the program is launched. It will contain the following:
  - **Connect Button:** The connect button will be used to connect to the microcontroller via Bluetooth.
  - **Disconnect Button:** The disconnect button will be used to disconnect from the microcontroller.
  - **Start Button:** The start button will be used to start the data acquisition from the microcontroller.
  - **Stop Button:** The stop button will be used to stop the data acquisition from the microcontroller.
  - **Exit Button:** The exit button will be used to exit the program.
- **Settings Window:** The settings window will be used to set the parameters for the data acquisition from the microcontroller. It will contain the following:
  - **Start Button:** The start button will be used to start the data acquisition from the microcontroller.
  - **Stop Button:** The stop button will be used to stop the data acquisition from the microcontroller.
  - **Exit Button:** The exit button will be used to exit the program.

- **Data Acquisition Window:** The data acquisition window will be used to display the data acquired from the microcontroller. It will contain the following:
  - **Start Button:** The start button will be used to start the data acquisition from the microcontroller.
  - **Stop Button:** The stop button will be used to stop the data acquisition from the microcontroller.
  - **Exit Button:** The exit button will be used to exit the program.

The user will be able to input a range for each parameter, or chose to ignore some of them, or even all of them in case of simple visualization of the data. If any system computed variable during any instant t is not within the range specific by the user, it will trigger a response of LEDs and Sound Emitters which will signal that the quadcopter is not functioning as desired.

4.2 Calculations and Measurements

The microcontroller will be responsible for taking the measurements from the sensors and performing the necessary calculations and sending the results to the user interface. The microcontroller will be responsible for the following:

- **Distance Measurement:** The distance measurement will be straightforward, the microcontroller will be automatically given the distance between the sensor and the object in front of it with the time of flight of the ultrasound signal.

- **Velocity Measurement:** The velocity measurement will be done point by point to be able to draw a graph of the velocity across each axis. The velocity will be calculated straightforwadly by the taking the distance Δd at a time t_i between points $i-1$ and i and dividing it by the time interval between two sent iterations.

$$v_i = \frac{\Delta d}{\Delta t} \tag{4.1}$$

- **Acceleration Measurement:** The acceleration measurement will be done point by point to be able to draw a graph of the acceleration across each axis. The acceleration will be calculated straightforwadly by the taking the velocity Δv at a time $t_i$between points $i-1$ and i and dividing it by the time interval between two sent iterations.

$$a_i = \frac{\Delta v}{\Delta t} \tag{4.2}$$

- **Orientation Measurement:** The measurement of the orientation is a bit more complex, but by using a Gyroscope, we could be comparing the inclination of the quadcopter with a base value then calculate the error and send it to the user interface.

4.3 Control System

The emission of the hyper-sound waves will be controlled via a Discrete Proportional Controller, which is a simple controller that is used to control the frequency of the sound waves emitted by the sound emitters. The controller will be designed as follows:

$$u(n) = K_p e(n) \tag{4.3}$$

Where $u(n)$ is the control signal, K_p is the proportional gain, and $e(n)$ is the error signal [2]. The error signal will be calculated as follows:

$$e(n) = r(n) - y(n) \tag{4.4}$$

Where $r(n)$ is the reference signal, and $y(n)$ is the output signal. The reference signal will be the desired frequency of the sound waves, and the output signal will be the frequency of the sound waves emitted by the sound emitters. The controller will be implemented in the microcontroller, and the microcontroller will be responsible for sending the control signal to the sound emitters.

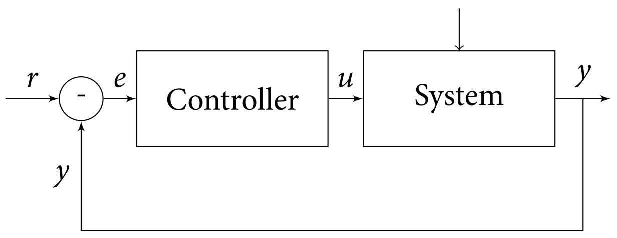

Figure 4.1: Block Diagram of the Control System

The controller is planned to be completed before the next phase of the project.

4.4 Flowchart

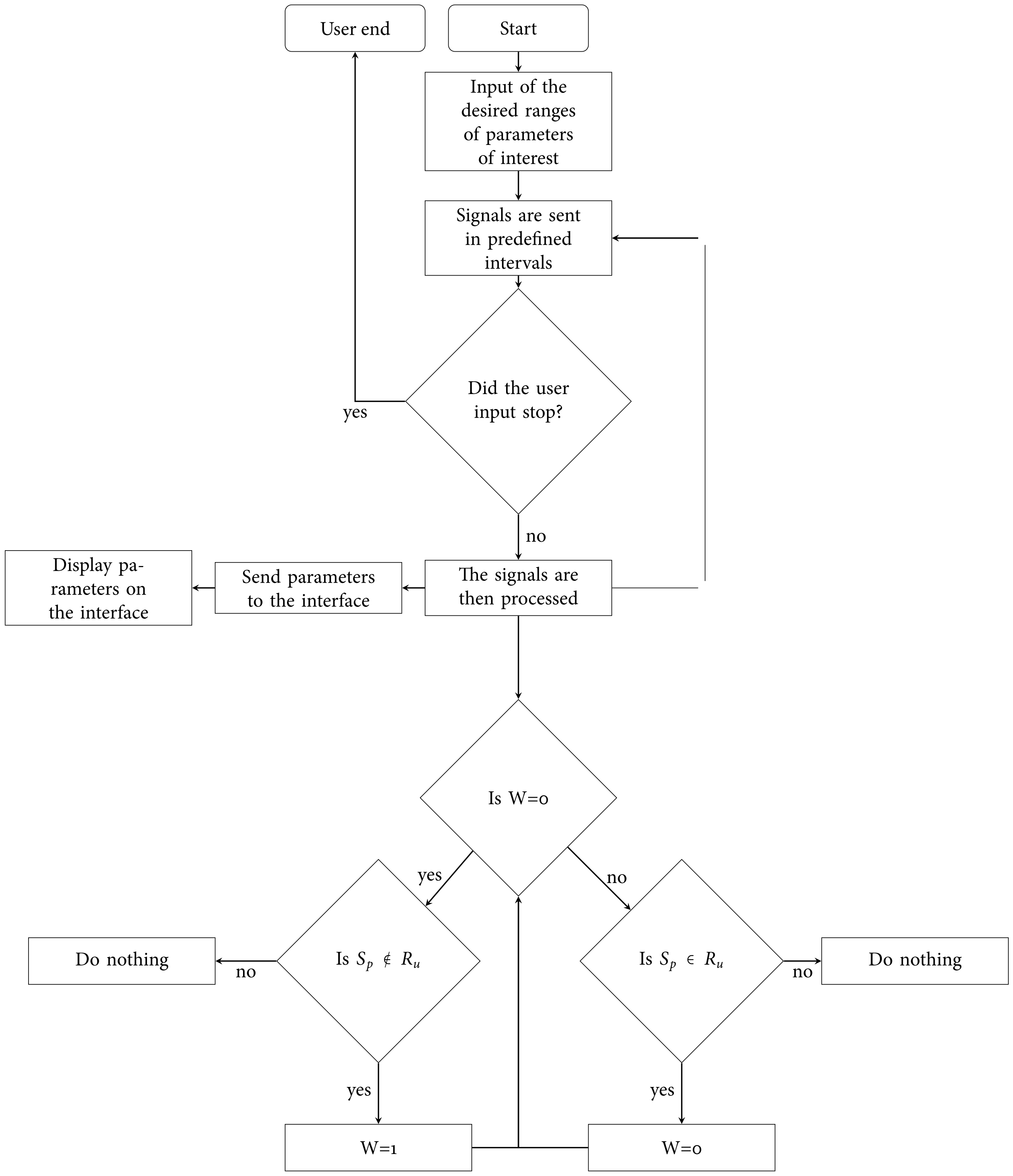
User end
Start
Input of the desired ranges of parameters of interest
Signals are sent in predefined intervals
Did the user input stop?
yes
no
Display parameters on the interface
Send parameters to the interface
The signals are then processed
Is W=0
yes
no
Do nothing
no
Is $S_p \notin R_u$
Is $S_p \in R_u$
no
Do nothing
yes
yes
W=1
W=0

Figure 4.2: Algorithm of the system

5 Kalman Filter

The Kalman Filter is a state estimation algorithm that is used to estimate the state of a system given a set of measurements. It is used in a wide variety of applications such as robotics, navigation, and control. The Kalman Filter is a recursive algorithm that uses the previous state and the current measurement to estimate the current state. The Kalman Filter is composed of two parts, the prediction step and the update step.[1]

5.1 Prediction Step

The prediction step is used to predict the state of the system at the next time step. It is composed of the following:

- **State Prediction:** The state prediction is used to predict the state of the system at the next time step. It is calculated by using the following equation:

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k \tag{5.1}$$

- **Covariance Prediction:** The covariance prediction is used to predict the covariance of the system at the next time step. It is calculated by using the following equation:

$$P_{k+1} = AP_kA^T + Q \tag{5.2}$$

5.2 Update Step

The update step is used to update the state of the system at the current time step. It is composed of the following:

- **Kalman Gain:** The Kalman Gain is used to calculate the optimal weighting between the prediction and the measurement. It is calculated by using the following equation:

$$K_k = P_{k+1}H^T(HP_{k+1}H^T + R)^{-1} \tag{5.3}$$

- **State Update:** The state update is used to update the state of the system at the current time step. It is calculated by using the following equation:

$$\hat{x}_k = \hat{x}_{k+1} + K_k(z_k - H\hat{x}_{k+1}) \tag{5.4}$$

- **Covariance Update:** The covariance update is used to update the covariance of the system at the current time step. It is calculated by using the following equation:

$$P_k = (I - K_kH)P_{k+1} \tag{5.5}$$

We will be using the Kalman Filter to minimize the error in the measurements of the sensors. The Kalman Filter will be used to estimate the state of the system at each time step. The state of the system will be the position of the quadcopter in the 3D space. The measurements will be the distance measurements from the sensors and the orientation of the quadcopter.

The Kalman Filter is planned to be completed before the end of the next Phase.

6 Conclusion and Future Work

The theoretical part of the project is still ongoing, but as it makes up the bulk of the project, and it will offer us a fruitful learning experience concerning both Kalman Filters and Discrete Time Control Systems, it is expected that the physical implementation of the project will start by the 10th of March, and will mostly be completed in a fraction of the time it takes to complete the theoretical study and algorithm of the system.
We are expecting to abide by the following schedule.

Dates	**Tasks**
27 Feb. - 2 March	Theoretical study of the Kalman Filter
3 March - 5 March	Theoretical study of Proportional Discrete time Control Systems
6 March - 9 March	Implementation of the various algorithms in C.
10 March - 13 March	Creation of the User Interface and system Integration.
13 March - 2 April	Testing and Debugging + Improvement of the System.

And it leaves us enough time to abide by the project deliverabes in the following weeks. Now all that is left is the bibliography section

References

[1] Phil Kim and Lynn Huh. *Kalman filter for beginners: With MATLAB examples*. CreateSpace, 2011. [Online; accessed 2023-02-26].

[2] Alan V. Oppenheim and Ronald W. Schafer. *Discrete-Time Signal Processing*. jul 23 2013. [Online; accessed 2023-02-26].