Arduino Based Low-Cost Experimental Unmanned Aerial Flight System for Attitude Determination in Autonomous Flights

A project present to
The Faculty of the Department of Aerospace Engineering
San Jose State University

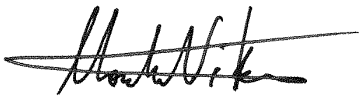
in partial fulfillment of the requirements for the degree
Master of Science in Aerospace Engineering

By

Jimmy Rico

2016

approved by



Dr. Kamran Turkoglu
Faculty Advisor

Arduino Based Low-Cost Experimental Unmanned Aerial Flight System for Attitude Determination in Autonomous Flights

Jimmy Rico and Kamran Turkoglu[y]
San Jose State University, San Jose, CA 95192, USA

This paper presents a low-cost experimental test bed of an Unmanned Aerial System (UAS) which is dedicated for autonomous ight testing and control system design. An in-depth look into the integration of the hardware is presented along with the design of the software in Arduino IDE and Matlab. The ability to compare and contrast between hardware and sofware in the loop simulation is developed to demonstrate safe ight char-acteristics for given ight control law design.

Nomenclature

	Roll Angle
	Pitch Angle
	Yaw angle
p	Roll Rate
q	Pitch Rate
r	Yaw Rate
m	Mass
b	Wing Span
S	Wing Area
c	Mean Aerodynamic Center
I_x	Moment of Inertia about the x-axis
I_y	Moment of Inertia about the y-axis
I_z	Moment of Inertia about the z-axis
	PWM Signal
	Density
P	Pressure
R_s	Speci c Gas Constant
T	Temperature
P_s	Static Pressure
P_s	Total Pressure
w	Gyro Measurement
	Correction Factor
t	Time di erence
g_y	Acceleration in the y-axis
g_z	Acceleration in the z-axis
	Elevator Command
	Aileron Command
	Rudder Command

Graduate Student, Aerospace Engineering, Control Science & Dynamical Systems (CSDy) Laboratory, jimmy.rico@sjsu.edu.

[y]Assistant Professor, Aerospace Engineering, kamran.turkoglu@sjsu.edu

I. Introduction

Low-cost avionics set-ups are becoming a commodity in the engineering industry, with vast application areas ranging from agriculture to monitoring and many more. Recent work has demonstrated the capabilities of low-cost ight computers. A great example is Dorbantu's[7] avionic system and its ability to demonstrate control law design and frequency domain analysis. Quite a few groups have also shown the capabilities of the ArduPilot Mega by designing closed loop control laws around the micro-controller through Matlab Simulink (c)[2,9,1]. Further, these same articles were able to demonstrate both hardware in the loop and software in the loop simulation. In this paper, as complementary to the existing literature, we present a low-cost based experimental research platform of an autonomous Unmanned Aerial System (UAS) which is based solely on easily accessible o -the-shelf products and open source Arduino software. Here, we aim to demonstrate a low cost-set up in terms of a blank canvas will be discussed in detail through the design of the avionics system, ight computer, to the integration of hardware, and software design to the nal results of the system.

The paper is organized as follows: In Section-II, xed-wing aircraft body frame, avionics system, IMU device, GPS system, data logging units, air-speed measurement units, is introduced, while in Section-III Arduino based software is explained in further details. In Section-IV we provide a brief background on the ltering techniques applied to the dynamics, and we explain autonomous control system design in Section-V.In Section-VI we provide the plan of our approach towards system identi cation of the paper and with the Section-VII, we conclude the paper.

II. Autonomous Unmanned Aerial System (UAS) Test Bed: Ultrastick 25e - Bare Framework

The Ultrastick 25e framework is a xed-wing aircraft body which comes with a suitable compartment, as shown in Fig. 1(b) for any add-on mechanism, device and/or ight computers. This model has the option between aperons or having aileron and aps. For this case, the latter is considered and most analysis is performed under this con guration. The following, Table 1, de nes the the physical properties of the aircraft at hand.

In the integration phase of the study, the Ultrastick 25e is utilized as the main frame of the UAS, and is used as the test bed along with an 10 DOF IMU, GPS, accelerometer, datalogger, pitot tube and Arduino microcontroller. To maintain a low-cost system, all products are based on o -the-shelf and easily available products.

(a) RC Model

(b) Avionics Compartment

Figure 1. Ultrastick 25e

A. Arduino Mega

For processing all the information and coding essentials, the Arduino Mega 2560[14] is used as the main avionics system for the model. The reason being that the Arduino Mega 2560 was chosen as the controller for the project is due to its low-cost, ease of use, online (and freely available) documentation support and functionality. The ArduPilot Mega, another Arduino Mega based micro-controller software system, was

Table 1. Physical Properties of the Aircraft

Property	Symbol	Value	Units
Mass	m	1.814	kg
Wing Span	b	1.27	m
Wing Area	S	0.31	m^2
Mean AC	c	0.25	m
Moment of Inertia	I_x	0.0653	kg-m^2
Moment of Inertia	I_y	0.0819	kg-m^2
Moment of Inertia	I_z	0.1054	kg-m^2

examined, but due to the fact that it lacks the ability to integrate various control law designs (such as optimal, adaptive, robust, nonlinear ... etc.), many of the libraries were re-written, modi ed and coded. This unique perspective provided us the chance to implement any type of control algorithm with the easy use (and functionality) of Arduino set-up. The software update portion of the project is iterated further in the following sections.

One of the unique aspects of the Arduino Mega board is that it has the ability to communicate with other hardware components through UART (hardware serial), I^2C (two wires), and SPI (4 wire). Among these, I^2C is easiest to work with because it only requires minimum amount of wires to interface with the micro-controller and it is suitable with the Arduino Libraries. Serial Peripheral Interface (SPI) is commonly used to send data between micro-controllers and SD card, along with other sensors that require 4 wires to interface with the Arudino board. It performs this by using a real time clock and data lines. Due to open source nature, most of the Arduino libraries already exist for many o -the-shelf products and allows to develop practically anything the user desires, considering the limitations of the micro-controller as shown in the following speci cations.

There are several iterations of the device, to name a few: the Leonardo, the Nano, the Uno, the Duemi-lanove, and the Micro, but the Arduino Mega 2560 provides the necessary memory and the number of input/outputs for the ight control system and hardware integration. A summary of the features embedded in the Arduino Mega 2560 system are as follows:

ATmega2560 Microcontroller

5V Operating Voltage

7-12v Recommended Input Voltage

54 Digital I/O (15 provide PWM ouput)

16 Analog Inputs

DC Current per I/O Pin 40mA

DC Current for 3.3V Pin 50 mA

4 UARTs (hardware serial ports)

256 KB Flash Memory with 8 KB Memory of RAM

16 Mhz Clock speed

USB connection

To allow for the integration of all devices and eliminating unecessary wires, a prototyping shield, as shown in Fig. 2, is mounted directly over the Arduino Mega to make installation and harware placement much easier for the user. This allows for the integration of the IMU, GPS, SD, including the servo output pins and receiver input pins to be easily installed. By doing so, this eliminates excessive use of wiring and connecting the bridge between hardware and micro-controller.

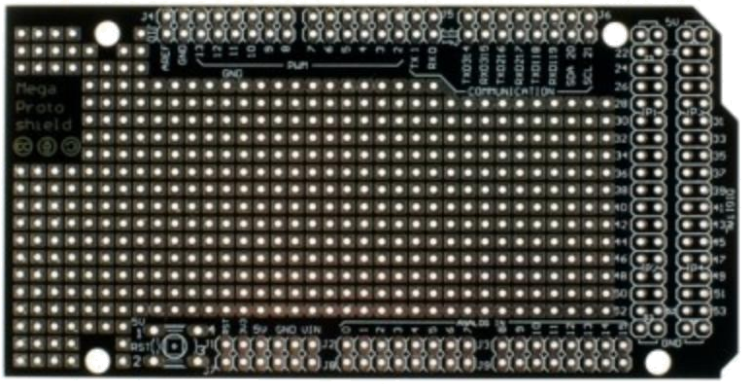


Figure 2. Arduino Mega Shield Top View

B. Inertial Measurement Unit (IMU)

The Inertial Measurement unit of interest for this low-cost setup is the Adafruit 10 degree-of-freedom Break-out board that combines three sensors to provide 3 axes of accelerometer data, 3 axes of gyroscopic data, and 3 axes of magnetic data along with temperature and pressure/altitude readings.[15] The bene t of working with these pieces of equipment is that it interfaces with the Arduino Mega through I^2C, once again, this device only requires two wires to interface with the micro-controller through SDA and SCL ports. Table 2 illustrates three sensors embedded into a single breakout board. The reason this product was chosen over three separate devices was to reduce the cost, weight and optimize space. With each of these devices, separate information can be obtained independently from the other. For example, the L3GD20H provides 3-axes of gyro data where Euler angles can be extracted from. Whereas the LSM303, provides both 3-axis of accelerometer and 3-axis compass data where Euler rates can be obtained from. The BMP180 is a barometric and temperature sensing device, where altitude, density, and temperature can be extracted from. With all these devices, an aircraft heading reference system can be established along the axis of the aircraft.

Table 2. IMU Breakout Board Speci cations

Sensor	Functionality	Speci cations
L3GD20H	3-axis gyroscope	250, 500, 2000 [deg/s]
LSM303	3-axis compass	1.3 to 8.1 gauss magnetic eld scale
LSM303	3-axis accelerometer	2g to 16g selectable scale
BMP180	barometric pressure/temperature	-40 to 85 , 300-1100 hPa range , 0.17m resolution

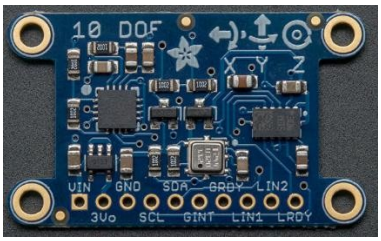


Figure 3. Adafruit 10 DOF IMU

As previously mentioned, by working with o -the-shelf products, the libraries for the Adafruit 10-DOF Breakout board are readily available to the user, where data can be obtained by referring to corresponding orientation, magnetic, and acceleration libraries.

C. Global Positioning System (GPS) Unit

The GPS, namely Adafruit Ultimate GPS (as shown in Fig.-4), is another breakout board that has the capabilities of an update rate of 1-10 Hz, where the update rate can be de ned by the user in the software, with a velocity accuracy reading within 0.1 m/s and position accuracy reading of <3 m.[16] This device is based o the MTK339 chipset, a GPS module that can track 22 satellites on 66 channels with a high sensitivity receiver. Two unique features that this GPS o ers are the data-logging capabilities and the external antenna functionality.[16] With the external antenna, adding a GPS antenna provides an additional 28 dB of gain and allows for the antenna to be mounted along the surface of the wing. Despite its internal data-logging, an external data logger is used to capture all the necessary information of the aircraft performance.

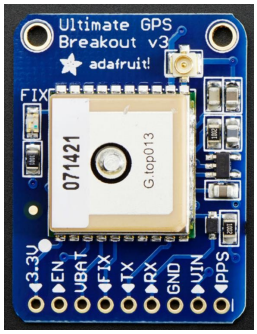


Figure 4. Adafruit Ultimate GPS

From this device, the longitudinal and lateral position, along with the velocity of the aircraft is obtained through National Marine Electronics Association (NMEA) protocol. Under this method, the user requests a string of information from the GPS and then interprets the data to a more useful value. In this case, the longitudinal and lateral coordinates as well as the current velocity of the model.

D. Data Logger

Since the data to be stored can become quite large, a class 10 32gb MicroSD card with an estimated 65 MB/s write speed and a SD breakout board was considered for the project over real-time data logging through an XBee communication device. The reason behind this is that with the XBee, data packets would be lost in transition during testing and the entire system would begin to lag, where the servo command inputs would take about a second to respond. To improve the rate at which data would be stored, an embedded data-logger seemed t and adequate to capture all the necessary data. The current frequency at which the data is stored is 72 Hz, where a le is initially created for data to be written and closed when the system shuts o . This method was implemented to save time from closing and opening the le after every iteration, and thus increasing the sampling frequency at which the data can be read and stored. This device interfaces with the Arduino Mega over the SPI network, which requires 4 wires for connection. These wires connect directly to the digital pins 50, 51, and 52. Additionally, pin 53 is required to select the SD card known as the hardware SS pin. Without this last pin, the SD library functionality is not guaranteed.

E. Pitot Tube and Airspeed Measurements

To measure the airspeed of the aircraft model, a di erential pressure sensor with a pitot static tube is required. The pitot tube of interest is the MPXV7002DP (Airspeed Sensor Kit), which is shown in Fig. 5. This model is designed for micro-processors or micro-controller-based systems and allows for easy integration through an analog input.[19] The pitot tube has the ability to measure positive and negative pressure, but accurate results are obtained when density is also taken into account. With the assistance of the BMP180, temperature and density values can be used to adjust for changes in pressure at various altitudes, thus providing an accurate airspeed reading. For this system, two airspeed sensors are attached on both sides of the wing to estimate an average airspeed over the course of ight.

Figure 5. Airspeed Sensor Kit

F. Battery Eliminator Circuit (BEC) and Electronic Speed Controller (ESC)

To supply the Arduino Mega with an adequate voltage (which is 7-12V range), a Castle Creastions 10 Amp adjustable electronic battery eliminator circuit (BEC) is used to supply a constant voltage of 7V. The voltage setting is adjusted via the Castle Link program, that is available online.[17] This device helps to eliminate the receiver and Arduino battery pack.Therefore, the avionics system is powered by a single 3300mAh 3-Cell/3S 11.1V battery pack. However, a separate battery pack is required to supply voltage to the servos independlty from the BEC. The current servo mechanisms requires a voltage of 4-6V, which are powered by four 1.5V/2400mAh NiMH batteries.

Since the model requires the use of brushless motor, an E-Flite 40 Amp Brushles electronic speed controller (ESC) is necessary to convert the pulse-width signal produced by the receiver into a signal that the motor can understand and use. The ESC will also rely on the main battery pack to power the brushless motor and the avionics system. The ESC depends on the motor itself, since the motor operates at a continuous current of 32 amps with a maximum burst current of 44 Amps, a 40-45 Amps ESC is required for the system to operate within safe limits to prevent the ESC from overheating and frying during ight.[18] Fig. 6 illustrate the wiring setup between the ESC and BEC.

Figure 6. ESC and BEC

G. Cost

In this section, the details of the total cost of the project is provided. Table 3 illustrates the breakdown cost of each piece of hardware.

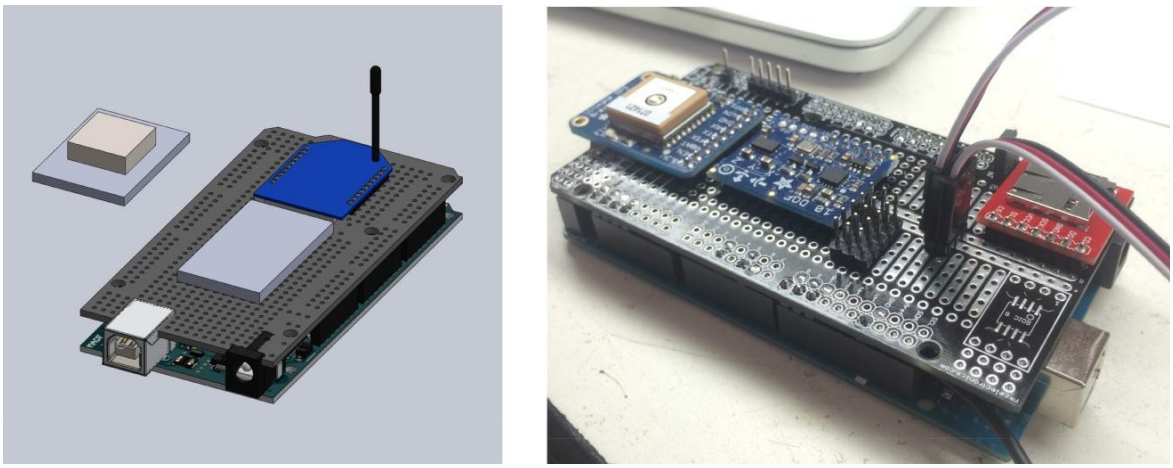
H. Flight Computer Design and Integration

To develop a proper and fully functional avionics system, the Arduino Mega 2560 micro-controller along with all the necessary hardware is developed and coded to provide a fully functional ight computer. Fig. 7(a) depicts a preliminary model of the avionics system designed in Solidworks, while Fig. 7(b) demonstrates the current and nal iteration of the avionics hardware. Before embarking on the nal iteration, a simple model was developed to illustrate the location of the GPS, IMU, and SD breakout board for neatness and simplicity before soldering onto the Arduino Shield. All the corresponding pins from each component are soldered to their respective pin inputs to the Arduino Mega. The 10-DOF requires that SCL ! digital Pin 21 and SDA

Table 3. Total cost of the UAS set-up integration

Hardware	Supplier	Product	Cost [USD]
RC Model	Amazon	Ultrastick 25e	169.99
Motor	Amazon	Power25	69.99
Servos	Amazon	6 JR Sport M-48	167.94
Electronic Speed Controller	Amazon	40A Brushless ESC	54.99
Reciever	Amazon	AR7010 7 Channel DSMX	89.99
Propeller	Amazon	Thin Elec. Prop 12x6E	3.95
Microcontroller	Arduino	Mega 2560	45.95
IMU	Adaftuit	10 DOF IMU	29.95
GPS	Adafruit	Ultimate GPS	39.95
Data Logger	Sparkfun	Sparkfun Micro SD	9.99
Pitot Tube	3DRobotics	Airspeed Sensor Kit	24.95
		Total	672.70

! Digital Pin 20. The GPS requires TX ! Pin RX, RX ! Pin TX. SD breakout board requires CS ! Digital Pin 53 , DI ! Digital Pin 51 , DO ! Digital Pin 50, SCK ! Digital Pin 52. Fig. 8 illustrates the given receiver input along the Analog Pin to Digital Pin outputs for the control surfaces and throttle. Inputs are as follows: Flaps ! A13, Aileron ! A12, Elevator ! A10, Throttle ! A9, and Rudder ! A8. The servo and motor commands are as follows: D11 ! Elevator, D9 ! Aileron, D8 ! Flaps, D5 ! Rudder, and D3 ! Throttle. Finally, all the components are connected directly to ground and 5V supplied by the Arduino. On the other hand, the servos and the Arduino are powered directly by servo battery pack and the BEC, respectively.



(a) CAD model

(b) Flight computer prototype

Figure 7. Flight computer

III. Software

To demonstrate the capabilities of a low-cost unmanned aerial system, the software development consists of Matlab, Simulink(c) and Arduino IDE. Within the Arduino IDE, all hardware is controlled by the algorithm written in Arduino. This includes the software written for the GPS, the IMU, the data-logger, and the control and interrupt sequence for servo actuation and throttle command. By doing so, control laws are designed and implemented through hardware in the loop simulation and then analyzed in Matlab and compared to software in the loop-simulations.

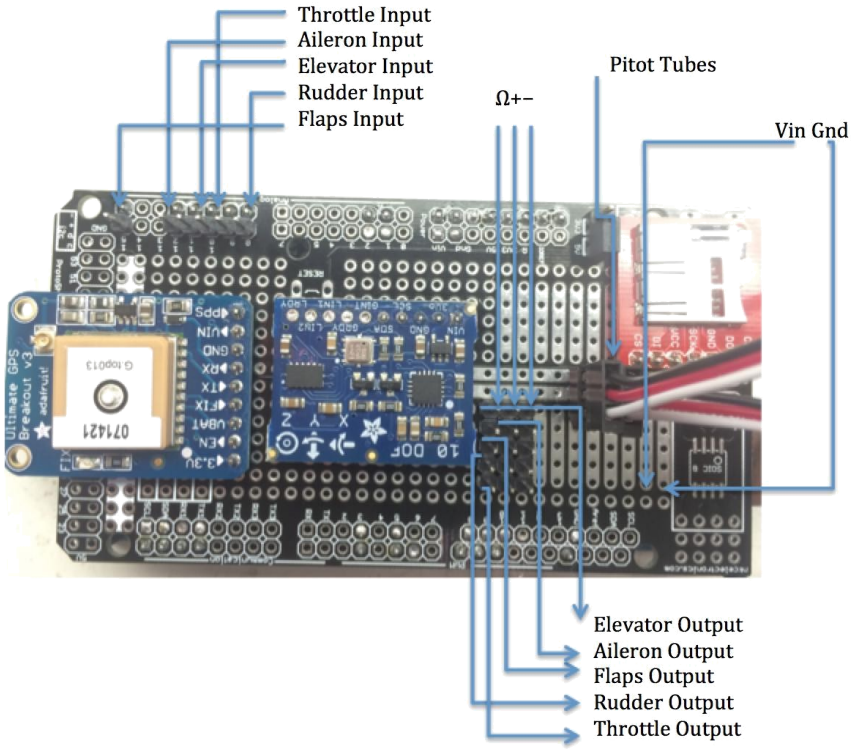


Figure 8. Pin Layout

A. Aircraft Heading Reference System

This section of the code is responsible for reading angular rates, acceleration, and compass heading to develop the Euler angles and Euler rates, in terms of a body and earth xed axes. The main libraries are provided by Adafruit for each corresponding sensor Embedded in the 10 DOF breakout board. Through every iteration, all of these parameters are updated and recorded at 72 Hz.

B. Controls & Receiver

The control surfaces of the aircraft are controlled through pulse width modulation (PWM) signals sent from the Arduino Mega to the control surfaces. In the current architecture, the transmitter sends the command to the receiver, where each command is transferred into the Arduino Mega through several analog pins, each corresponding to the throttle, ailerons, aps, elevator, and rudder. Once the signal has been received, the signal is read by interrupts system embedded in several digital pins on the Arduino Mega board, and when a new command has been placed, a ag is raised by the interrupt and tells the system to update the control surface or adjust the rotations of the motor. When the iteration has gone through, the ag is lowered and the process begins again. This process is conducted to eliminate undesired control surface commands by rst processing incoming commands before relaying out the response. At this phase, control laws can be implemented by simply switching one of the predetermined knobs/switch on the transmitter. By doing so, the pilot has the capability of switching between automated sequence and pilot input commands on the y. This is bene cial in terms of emergency landing in case the automated sequence fails.

C. Airspeed, Temperature, Pressure and Density

As previously noted, the pitot static system along with the BMP180 are coded to work in unity. The pressure measurement requires a simple conversion code that translates a voltage reading to a pressure value. With a known temperature and pressure value, density is then determined via Eq.(1). Then, Eq.(2) is applied to determine the airspeed of the aircraft from known static pressure and dynamic pressure values.

$$= \frac{P}{R_s T} \tag{1}$$

Here, R_s denotes the speci c gas constant for air, P_t and P_s are the total pressure and static pressure, respectively.

$$u = \sqrt{\frac{2(P_t - P_s)}{}} \tag{2}$$

D. GPS

The GPS program allows for the retrieval of latitude, longitude, and velocity of the aircraft through the NMEA command module. The library used for this system is known as TinyGPS. This library was chosen because it does not require the use of Software Serial as most other GPS libraries do. Since the transmitter requires the use of Software Serial, having both devices operate on the same Serial would lead to an error and the code would not upload to the micro-controller. Therefore, TinyGPS is the most suitable library for the task at hand.

E. Data Logging

The collection of all valuable data begins with the data logger. This piece of software operates by initializing a le once the system has been turned on, it then converts all data into a string and stores all information as an array. The data is recorded in terms of micros(), a command in Arduino IDE that allows for data to be analyzed and stored in microseconds. As previously mentioned, all the data is being stored at 72 Hz per iteration.

The data-logging process involves saving the data into a string value. Most of the data is translated from degree to a string with the dtostrf() command in Arduino IDE while operating the SdFat library. By doing this, there is no need for the program to continually open and close the SD le to write the data, which actually slows down the system. Thus, to increase the rate at which data is saved, a 510 byte block is initially created and then the string data could be stored at a much faster rate. When trying to store oating point values, the system would operate at 20 Hz, but this can only be achieved by opening and closing the SD le after every iteration. Therefore, to take advantage of the 72 Hz update rate, all data must be translated into a string and then stored into the SD card as a 510 byte block. The system can handle a 512 byte block, and a 510 block byte would su ce since most information was less than the speci ed value. However, if anything exceeds that value, no data could be stored past that block. The remaining portion of data processing is handled through Excel and Matlab, respectively.

F. Control Law Design

At this point, additional control law design can be easily implemented through the current avionics system. This can be achieved via the Arduino IDE or through Matlab Simulink(c). The bene t of using Simuilnk(c) to design control laws is its ease of use and the fact that real-time software in the loop simulation (SIL) allows for the analysis of the current model and make changes accordingly, whereas the HIL would require to manually transfer the data onto Matlab, analyze the results and then update the information. Further discussion on control law design is provided in Section-V.

IV. Filtering Techniques and Estimation

When relying on sensor measurements for optimal performance, noise can lead to undesired results. To compensate for that error, certain ltering techniques have to be approached. If left un xed, noise may amplify within the feedback system and lead to undesired results. Therefore, to eradicate the possibility of failure, ltering is a must in these type of schemes where aircraft are highly dependent on synthetic data. The following sections will iterate which type of schemes were chosen to eliminate or minimize this noise.

A. Complementary Filter

To properly adjust the gyro drift measurements and incorrect accelerometer readings during vibrations, both matters are taken into account in the low pass and high pass frequencies. By taking advantage of these two parameters, estimated data becomes more precise as compared to the raw data[11].[8] Analogy provided in Eq.(3) is applied to aid and correct this issue.

$$= (_{k\ 1} + w_k\ t) + (1\quad) _{accel;k} \tag{3}$$

Here $_{k\ 1}$ is the previous pitch angle value, w_k is the angular velocity measured from the gyro, and $_{accel;k}$ is the pitch angle at that instant given by equation 4 from accelerometer readings.

$$_{accl} = \tan^{1} \left(\frac{g_y}{g_z}\right)_{0} \tag{4}$$

Here, $_0$ de nes the correction factor. It was found that = 0.98 provides a clean signal that does not deviate over time and estimates the angles fairly well, as shown in Fig. 9

B. Kalman Filter

The Kalman lter is an algorithm that uses a series of past measurements and present information over time to estimate a precise value in respect to the noisy raw value. This technique operates in a recursive matter on noisy data to predict a clean response. After predicting a value, the estimates are yet updated using a weighted average, where more weight is given to those with higher certainty. Here Patra (2013)[3] , demonstrates the advantages of using a Kalman lter for a missile subjected to two noise inputs, along with the derivation of the covariance matrix and weighted matrix and LQR application. The Eq.'s(5-9) demonstrate the Kalman ltering scheme, where more in depth analysis is provided in Patra(2013)[3] and Liu (2008),[12] as well as many other valuable resources.

$$x_{k+1} = F_k x_k + G_{u;k} u_k + G_{v;k} v_k \tag{5}$$

$$y_k = H_k x_k + D_k u_k + e_k \tag{6}$$

$$Q_k = Cov(v_k) \tag{7}$$

$$R_k = Cov(e_k) \tag{8}$$

In this formulation, x are the state and y are the measurements, e is the error of the system, de ned as

$$e = x\ \ \hat{x} \tag{9}$$

C. Extended Kalman Filter

Under this method, a similar approach (as seen in the Kalman lter) is applied. However, linear and non{ linear systems are taken into account, through Eq.(10) and Eq.(11).[21] Similarly, the concept is also used on GPS and IMU data estimation through an adaptive extended Kalman lter to fuse the information together.[20]

$$x_{k+1} = f(x_k; u_k; v_k) \tag{10}$$

$$y_k = h(x_k; u_k; e_k) \tag{11}$$

Then, the solution is determined by taking the Jacobian of $f(x_k; u_k; v_k)$ and $h(x_k; u_k; e_k)$ and assuming Gaussian noise, as shown in Eq.(12) and Eq.(13). Further detail on this process is given in Kallapur (2006).[21]

$$x_{k+1} = f(x_k; u_k) + v_k \tag{12}$$

$$y_k = h(x_k; u_k) + e_k \tag{13}$$

The following Covariance values for the angle and rate are used for implementation: Q_{angle} = 0:01, Q_{gyro} = 0:0003 with R_{angle} = 0:01. These values were used with Kalman lter and EKF ltering routines, respectively. However, in experimental validations, EKF provided results which tend to lag dramatically, and requires further investigation. Here, Extended Kalman Filter (EKF) results are not provided due to the latency issues related to the calculation time within given sampling period, and only Kalman Filter results together with the Complementary Filter results are provided (as shown in Fig.-9), for comparison.

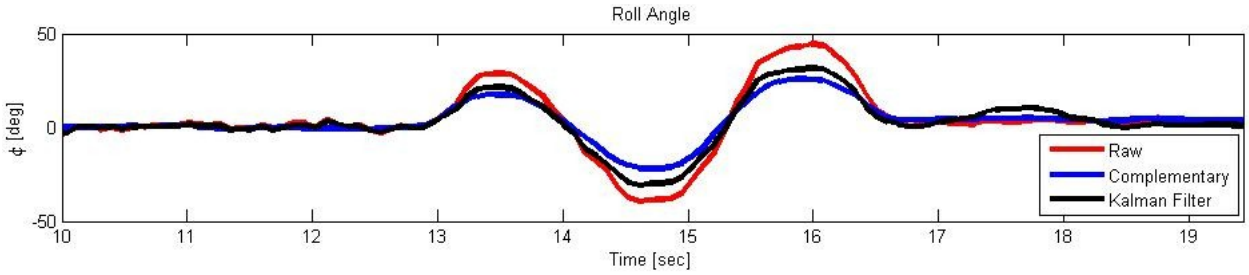


Figure 9. Roll angle measurements through various lters

V. Autonomous Control System Design

The ability to reach a desired response with minimal cost is bene cial to a pilot, or in this case, an RC model aircraft. To properly reach this goal, certain control methodologies have to be accounted for.

A. PID

A PID controller is designed to minimize the control e ort for each of the control surfaces while reaching a desired response. This article[9] illustrates the use of all three parameters of the proportional-integral-derivative PID controller, shown in equation 14.[10] At times, it may be best to only use one or two of the terms (such as PI, PD), and it is not uncommon to use all three parameters to reach an optimal response. As well known from literature, each term addresses di erent issues in the system.

$$P\,ID(s) = K_p\left(1 + \frac{1}{T_i s} + T_d s\right) \tag{14}$$

After having developed the proper transfer functions, as shown in Section-VI, a feedback system can be designed to control the response of the aircraft with respect to a given command. In this case, a pitch and roll controller were developed using P and PD type control algorithms respectively. The results are shown in Fig.-10. The given values are shown in Table 4

Table 4. PID Controller Design Values

Controller	P	I	D
Pitch	-1	0	0
Roll	1	0	0.1

VI. System Identi cation

To investigate longitudinal and lateral system dynamics presented and developed in this article are based on the fundamental ight dynamics theory presented in.[13] In addition, a frequency sweep to excite the natural longitudinal and lateral modes of the aircraft will be performed and then analyzed to produce the proper transfer functions for validation using CIFER[23] and Matlab.

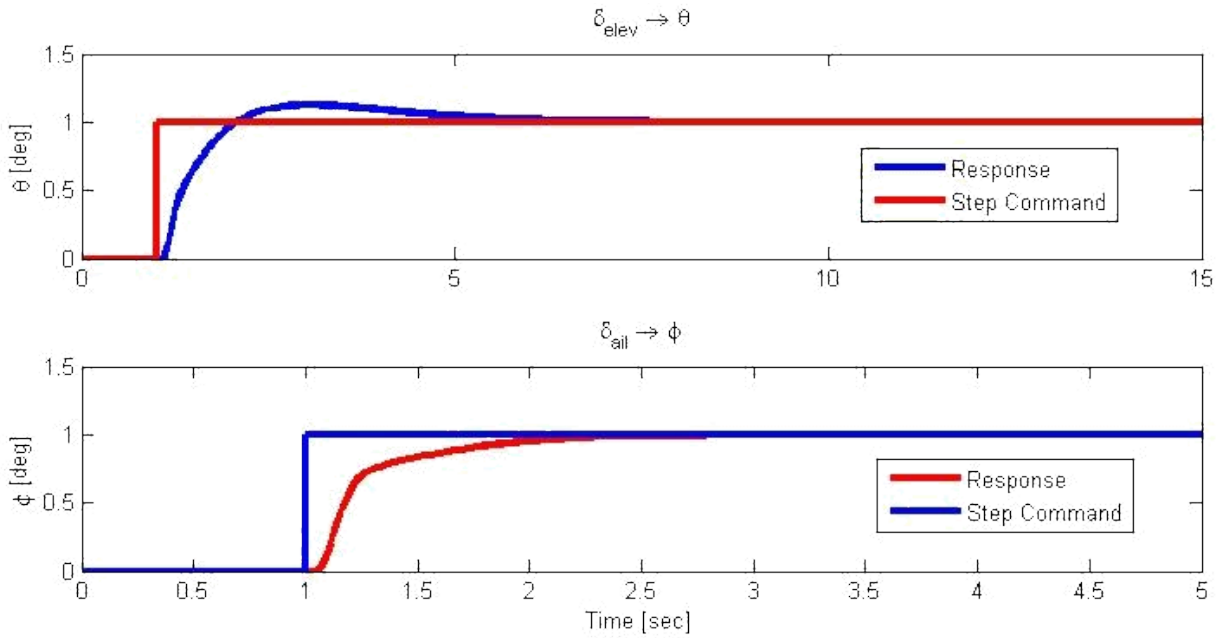


Figure 10. Closed loop response for 1^{o} step input.

A. Open Loop Analysis

Considering the fact that the xed wing aircraft is naturally stable, the frequency sweep analysis was performed under open loop analysis. Control law design was implemented for self-stabilization, however, the choice to proceed without a feedback design seemed the best option to capture the true dynamics of the aircraft. Experience tells that investigation of chirp signal response in closed loop dynamics has the potential to mask (and/or compensate) for mode dynamics, which is not desired.[23]

To identify a model, a frequency sweep via a chirp signal is taken into e ect. For the case of an aircraft, both longitudinal and lateral modes have to be excited via actuator commands, where the actuator will follow an automated sinusoidal command at a certain magnitude from 0.01 Hz to 15 Hz for the duration of 10 seconds. To develop the lateral system, both rudder command and aileron command have to be taken into account. When one command is active, the other must remain at trim, and vice-versa. The following Fig. 11 illustrates a chirp command on the rudder, while the ailerons are maintained at trim. From this data, the roll rate and yaw rate response are then extracted into CIFER for system identi cation. The same procedure is prevalent for the longitudinal dynamics analysis.

B. SISO Identi cation Analysis

For system identi cation, the student version of CIFER[23] is used. This software has the capabilities of producing state space models as well as transfer functions by analyzing data in the frequency domain. The following, Eq.(15), depicts a transfer function developed , as shown in Fig.-14.

$$\frac{q(s)}{(s)} = e^{\ _1 s} \quad \frac{51{:}415s(s + 10{:}4)(s + 0{:}1)}{(s^2 + 0{:}384s + 0{:}16)(s^2 + 27{:}24s + 282{:}6)} \tag{15}$$

where the associated time delay-( ) component is $_1 = 0{:}07$.

C. MIMO Identi cation Analysis

In terms of multi-input-multi-output analysis, there were issues in extracting proper data from the student version of CIFER as the coherence for several parameters would fall below desired values and not enough information could be extracted. This provides the need for better, cleaner and longer chirp data sets for further analysis. However, Fig. 12 illustrates an adequate coherence plot for yaw rate to a rudder sweep.

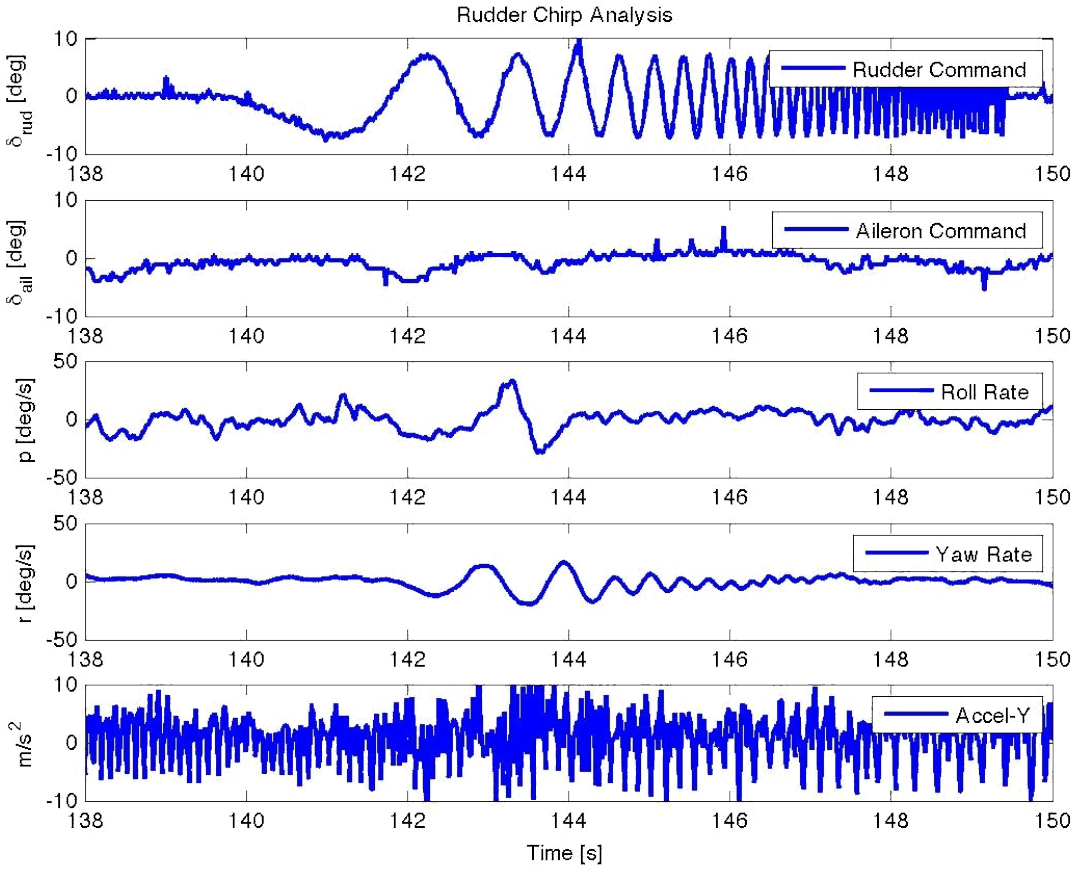


Figure 11. Rudder automated chirp analysis

Therefore, proper information could be extracted from this plots and the transfer functions for rudder to yaw rate response, as shown in Eq.(17), and aileron to roll rate, as shown in Eq.(16).

$$\frac{p(s)}{(s)} = e^{\ _2 s} \quad \frac{115{:}73s(s^2 + 6{:}262s + 19{:}45)}{(s + 12{:}34)(s + 0{:}019)(s^2 + 6{:}906s + 35{:}32)} \tag{16}$$

$$\frac{r(s)}{(s)} = e^{\ _3 s} \quad \frac{37{:}733(s + 7{:}706)(s^2 + 0{:}54s + 0{:}81)}{(s + 12{:}34)(s + 0{:}019)(s^2 + 6{:}906s + 35{:}32)} \tag{17}$$

where the associated time delay-( ) values are $_2 = 0{:}08$ and $_3 = 0{:}06$, respectively.

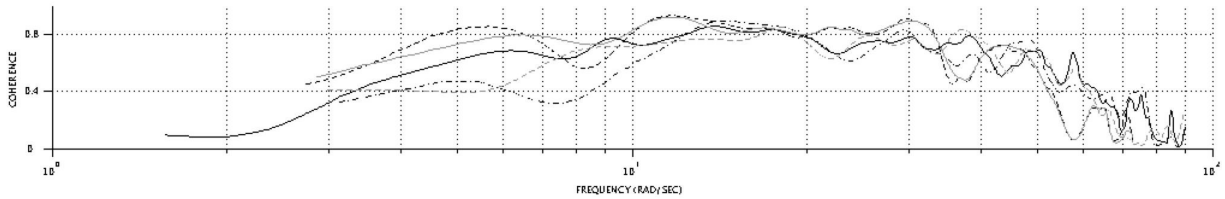


Figure 12. Yaw Rate Coherence in CIFER

D. Validation

To verify the accuracy of the model, an automated doublet command at 7 degree of actuator response is applied to the aircraft and compared to the results using the identi ed transfer function with the same input magnitude. Fig. 13-15 represents a doublet command input on each control surface to their respective rates. It can be seen that the transfer functions t the actual aircraft response really well and thus validating the transfer functions as a good t for the system dynamics.

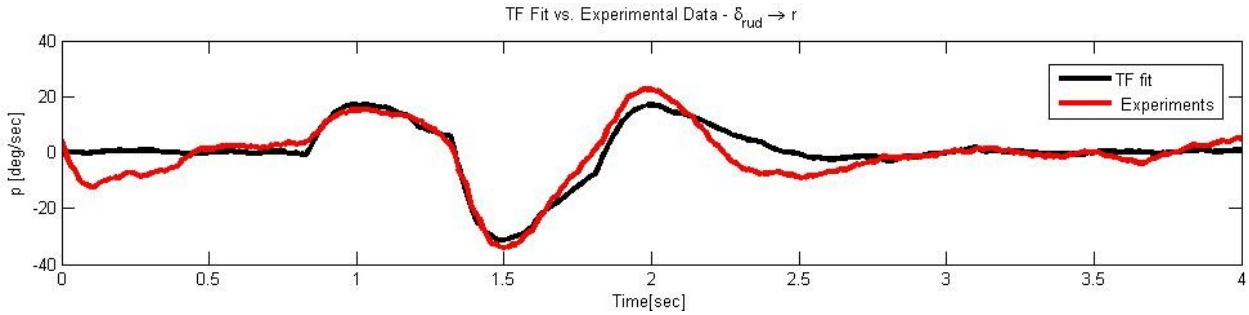


Figure 13. Yaw Rate Response to Doublet Command of 7 degrees of rudder de ection

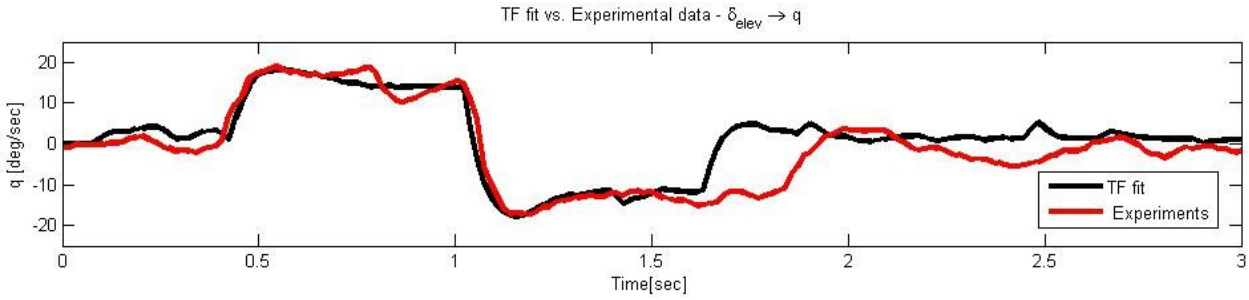


Figure 14. Pitch Rate Response to Doublet Command of 7 degrees of elevator de ection

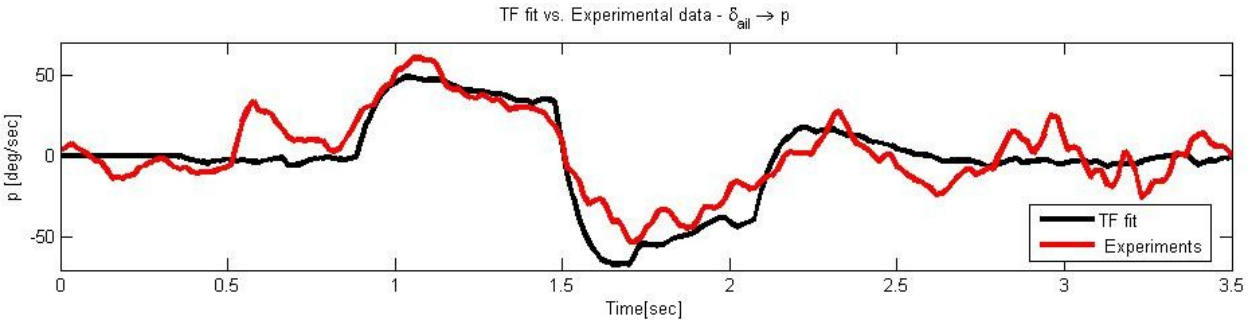


Figure 15. Roll Rate Response to Doublet Command of 7 degrees of aileron de ection

VII. Conclusion

In this paper, successful integration and validation of the hardware and software through the Arduino IDE environment and veri ed the results in MATLAB are demonstrated. An in-depth analysis of ltering schemes and ight data analysis was performed to produce acceptable results through the use of the Arduino Mega and low-cost components. System identi cation produced an adequate t for the angular rates with their respective control input, where better and improved results are available through more precise chirp measurements, and is included in the future work. With the presented system identi cation procedure,

control law design becomes achievable and could be easily integrated into the Arduino code to improve the ight characteristics of the aircraft, which is seen in ight in Fig.16.

Further e orts will be driven to x the important issues between the coherence plots of the remaining states of the aircraft to produce a full blown state space representation of the system dynamics and allow for the more advanced (such as Adaptive, Optimal, Robust, ... etc.) control methodologies to be applied and tested.

Figure 16. Flight testing computer design

VIII. Acknowledgements

The authors would like to thank the George Mateer, a retired Aerospace Engineer and RC hobbyist who took the liberty and time to y the aircraft during the testing phase of the project and sharing insights.

References

[1]Coombes, M., McAree, O., Wen-Hua Chen, "Development of an autopilot system for rapid prototyping of high level control algorithms," 2012 UKACC International Conference on Control (CONTROL), IEEE, Piscataway, NJ, USA, 2012, pp. 292-7.

[2]Esteves, D.J., Moutinho, A., and Azinheira, J.R., "Stabilization and Altitude Control of an Indoor Low-Cost Quadro-tor: Design and Experimental Results," Autonomous Robot Systems and Competitions (ICARSC), 2015 IEEE International Conference on, 2015, pp. 150-155.

[3]Patra, N., Halder, K., Routh, A., Mukherjee, A., and Das, S. (2013). Kalman lter based optimal control approach for attitude control of a missile. Paper presented at the Computer Communication and Informatics (ICCCI), 2013 International Conference on, 1-4. doi:10.1109/ICCCI.2013.6466158

[4]Lie, F. A., & Gebre-Egziabher, D. (2013). Synthetic air data system. Journal of Aircraft, 50(4), 1234-1249. doi:10.2514/1.C032177

[5]Shiau, J., & Wang, I. (2013). Unscented kalman ltering for attitude determination using MEMS sensors. Journal of Applied Science and Engineering, 16(2), 165-176. doi:10.6180/jase.2013.16.2.08

[6]Soken, H. E., & Hajiyev, C. (2009). Adaptive unscented kalman lter with multiple fading factors for pico satellite attitude estimation. Paper presented at the 2009 4th International Conference on Recent Advances in Space Technologies (RAST 2009), 541-6. doi:10.1109/RAST.2009.5158254

[7]Dorobantu, A., Murch, A., Mettler, B., & Balas, G. (2013). System identi cation for small, low-cost, xed-wing unmanned aircraft. Journal of Aircraft, 50(4), 1117-30. doi:10.2514/1.C032065

[8]Euston, M., Coote, P., Mahony, R., "A complementary lter for attitude estimation of a xed-wing UAV," Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on, 2008, pp. 340-345.

[9]Sufendi, Trilaksono, B.R., Nasution, S.H., "Design and implementation of hardware-in-the-loop-simulation for uav using pid control method," Instrumentation, Communications, Information Technology, and Biomedical Engineering (ICICI-BME), 2013 3rd International Conference on, 2013, pp. 124-130.

[10]Zhang Peng, and Liu Jikai, "On new UAV ight control system based on Kalman & PID," Intelligent Control and Information Processing (ICICIP), 2011 2nd International Conference on, Vol. 2, 2011, pp. 819-823.

[11] Coopmans, C., Jensen, A.M., and YangQuan Chen, "Fractional-order complementary Iters for small unmanned aerial system navigation," Unmanned Aircraft Systems (ICUAS), 2013 International Conference on, 2013, pp. 754-760.

[12] Liu, C., Zhou, Z., and Fu, X., "Attitude determination for MAVs using a Kalman Iter," Tsinghua Science and Technology, Vol. 13, No. 5, 2008, pp. 593-597.

[13] Cook, M.,Flight Dynamics Principles:A linear system approach to aircraft stability and control, 3rd ed., Elsevier, Burling-ton,MA, 2013

[14] "Arduino 2015, available http://www.arduino.cc/en/Main/ArduinoBoardMega2560"

[15] "Adafruit 2015, available http://www.adafruit.com/product/1604"

[16] "Adafruit 2015, available http://www.adafruit.com/products/746"

[17] "CastleCreations, available http://www.castlecreations.com/"

[18] "E ite 2015, available http://www.horizonhobby.com/ultra-stick-25e-arf-e 4025"

[19] "Robotshotp 2015, available http://www.robotshop.com/en/airspeed-sensor-kit.html"

[20] Xiaogang Wang, Jifeng Guo, and Naigang Cui, "Adaptive extended Kalman Itering applied to low-cost MEMS IMU/GPS integration for UAV," Mechatronics and Automation, 2009. ICMA 2009. International Conference on, 2009, pp. 2214-2218.

[21] Kallapur, A.G., and Anavatti, S.G., "UAV Linear and Nonlinear Estimation Using Extended Kalman Filter," Compu-tational Intelligence for Modelling, Control and Automation, 2006 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on, 2006, pp. 250-250.

[22] Guoqiang Mao, Drake, S., and Anderson, B.D.O., "Design of an Extended Kalman Filter for UAV Localization," Infor-mation, Decision and Control, 2007. IDC '07, 2007, pp. 224-229.

[23] Tischler, M. B., and Remple, R. K., Aircraft and Rotorcraft System Identi cation: Engineering Methods and Flight Test Examples, 2nd ed., AIAA Education Series, AIAA, Reston, VA, 2012,