ISSN 2278-3091
Volume 9, No.1.2, 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse1391.22020.pdf
https://doi.org/10.30534/ijatcse/2020/1391.22020

Blimp Stabilization Controller Optimization using Fuzzy Logic

Mary Ann E. Telen[1], **Sherwin A. Guirnaldo**[2]
[1]University of Science and Technology of Southern Philippines, Philippines, maryann.telen@ustp.edu.ph
[2] Mindanao State University-Iligan Institute of Technology. Philippines, sherwin.guirnaldo@g.msuiit.edu.ph

ABSTRACT

Blimps are used for mission-based application due to its ability to hover at any direction the pilot would direct. Earlier studies implemented manual control and some feedback control system but still the blimp is susceptible to air interference, making it hard to compensate disturbance on its orientation. This study focus on self-stabilization control using fuzzy logic in its feedback control system to achieve basic maneuvering and balance of the blimp. By employing sensor fusion using complimentary filter provides smoother data when subject to air disturbances. The system now ensures stability in its mission flights via its feedback control system using a fuzzy logic algorithm with active braking mechanism and fast updating frequency sensor system.

Key words: Blimp, fuzzy logic controller, feedback control system, self-stabilized, sensor fusion

1. INTRODUCTION

Blimps are a type of lighter-than-air (LTA) craft called an airship which are non-rigid type of dirigible balloon or airship. Similar to a hot air balloon, blimps use a gas to generate lift. However, blimps can move forward through the air under their own power, like airplanes [1]. Blimps have streamlined shape and stabilizing tail fins.

Compared to the manned or semi-manned airship, unmanned blimp control involves providing independent and accurate flight operations with little human intervention. The control methods implemented on autonomous airship lie in two categories, traditional control methods, and advanced control methods. The traditional control methods achieve autonomous control goals via classical control algorithms, such as the PID control theorem. These control methods have the benefits of being implemented and delivering consistent control efficiency, while the disadvantages include the costs of computation to model the system and adjusting the control parameters. Advanced control methods are becoming more common for blimp autonomous operation, as these control methods are primarily designed to improve the control efficiency of an autonomous airship in a complicated and unpredictable flight [2].

Fuzzy Logic Controller model becomes a wide way used to resolve problem related to system control in the major area categories [3]. However, sufficient knowledge about exploiting rules and membership functions is required and can have an effect on the performance of Fuzzy results. Two inputs generally deployed to the Fuzzy Controller: the system transfer function (Input: I), and the error (ΔI). Author in [4] has modified the control rules of two-input FLC to three-input FLC. Simulation results show that the Three-input properties excel two-input FLC properties.

The current system suffers some stability issues such that the blimps can easily destabilize by air disturbances and the blimp could not recover from such disruption of its flight. The controller of the blimp needs to compensate disturbances to attain a stabilized orientation using feedback control system. This will help the controller to maneuver the blimp to stay or hover on the desired track.

This study focuses on the development of self-stabilized control using fuzzy logic which is considered as an advanced method of implementing feedback control system. It is considered as very robust, supports for multiple input and multiple output systems. It is economical and easier to implement.

Usage of all-in-one hardware sensor with accelerometer, gyroscope and magnetometer data as inputs to microcontroller, this can save cost and reduce additional hardware installation. Combine gyroscope and accelerometer sensor reading to achieve the most accurate data to make the airship more stable. A fuzzy controller is used to implement stability and developed an Arduino program to implement the self-stabilization control.

2. METHODOLOGY

The development phase of the study, namely analysis and planning, hardware installation, Fuzzy Controller implementation, testing of the system responses and closure. Some of the procedures used by [5], [6], and [7] were also used in the study. The methods and materials are anchored on the system framework as shown in Figure 1 where the blimp is controlled by a feedback mechanism using fuzzy logic to support the basic internal components for stabilization as shown in Figure 2.

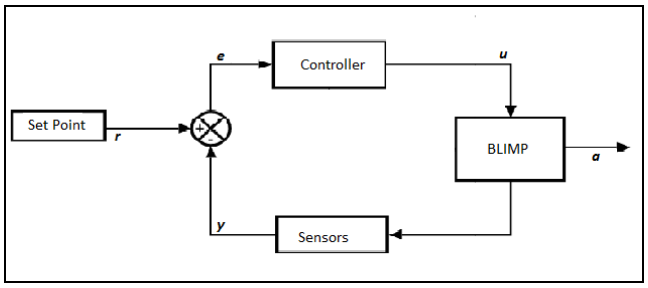


Figure 1: Controller System Framework

Legends:

r = [$r_{Øx}$, $r_{Øy}$, $r_{Øz}$]^T is the set point values. $r_{Øx}$ is the set point value along x-axis (fusion of magnetometer, accelerometer & gyroscope) (double). $r_{Øy}$ is the set point value along y-axis (fusion of magnetometer, accelerometer & gyroscope) (double). $r_{Øz}$ is the set point value along z-axis (fusion of magnetometer, accelerometer & gyroscope) (double).

u = [u_{w1}, u_{w2}, u_{w3}, u_{w4}, u_{s1}, u_{s2}, u_{s3}, u_{s4}]^T is the command for DC motors and command for Servo motors. u_{w1} is the command for DC motor1 (V) (double). u_{w2} is the command for DC motor2 (V) (double). u_{w3} is the command for DC motor3 (V) (double). u_{w4} is the command for DC motor4 (V) (double). u_{s1} is the command for Servo motor1 (V) (double). u_{s2} is the command for Servo motor2 (V) (double). u_{s3} is the command for Servo motor3 (V) (double). u_{s4} is the command for Servo motor4 (V) (double).

y = [$y_{Øx}$, $y_{Øy}$, $y_{Øz}$]^T is the output loop-feedback value from the sensor. $y_{Øx}$ along x-axis (fusion of magnetometer, accelerometer & gyroscope) (double), $y_{Øy}$ along y-axis (fusion of magnetometer, accelerometer & gyroscope) (double) and $y_{Øz}$ along z-axis (fusion of magnetometer, accelerometer & gyroscope) (double).

a = [$a_{Øx}$, $a_{Øy}$, $a_{Øz}$]^T is the actual value. $a_{Øx}$ actual value along x-axis (fusion of magnetometer, accelerometer & gyroscope) (double), $a_{Øy}$ actual value along y-axis (fusion of magnetometer, accelerometer & gyroscope) (double) and $a_{Øz}$ actual value along z-axis (fusion of magnetometer, accelerometer & gyroscope) (double) and **T**= sample time (int).

2.1 Input and Output Data

Set Points
The set point r is the desired value for the self-stability control for the blimp. It is composed of { $r_{Øx}$, $r_{Øy}$, $r_{Øz}$ }. It will be manually set by the pilot of the airship. Deciding the set point will depend on whether the controller can compensate or attain it without error. If it can't compensate or attain the set point, then will consider the error and adjust the set point. This will repeat in the trial and error state until such set point is recognized and considered.

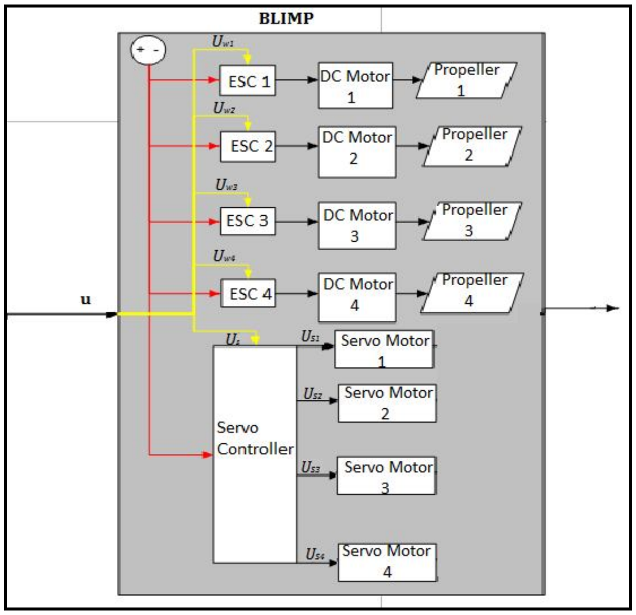


Figure 2: Blimp Internal Components for Stabilization

Process Values
The Processed Value y(t)' is the value from the sensor readings and will be compared to the set point value to know the error and identify its steady state. It is composed of { $r_{Øx}$, $r_{Øy}$, $r_{Øz}$ } values, the combined data from three-axis from an accelerometer, magnetometer, and gyroscope, for the accurate reading of orientation and heading three-dimensional axis. Sensor reading from the IMU will be sent via direct connection of the 9-DOF IMU to the Arduino microcontroller.

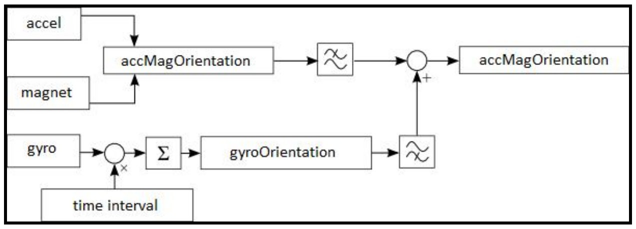


Figure 3: Complementary Filter

Sensor Fusion
Figure 3 illustrates how the complementary filter operates to generate a cleaner output. The researchers used the Adafruit's 9DOF (9 Degrees of Freedom) breakout board that allows capturing nine distinct types of motion or orientation-based data: 3 degrees each of acceleration, magnetic orientation, and angular velocity/ gyro. The two angles are based on the accelerometer and magnetometer output. However, both sensor outputs are inaccurate,

especially the output from the magnetic field sensor which includes a lot of noise [8] [9].

The gyroscope in the system is much more accurate and has a very fast response time. Its drawback is that of the gyro drift. The gyro sets the angular rotation speeds for all three axes. These speed values need to be averaged over time to get the true orientation. This is achieved by comparing the angular velocity with the time interval between the last and the latest sensor output. This results in an improvement in rotation. The sum of all rotation increments is the absolute orientation of the device. Minor errors are added in each iteration during this process. Such minor errors add up to the gyro drift, resulting in a continuous sluggish rotation of the measured direction

To bypass the gyro drift and noisy orientation, the gyroscope output is applied only for orientation adjustments in short time intervals, while the magnetometer/accelerometer data is used as backup information over long periods of time. It is similar to the low-pass filtering of the accelerometer and magnetic field sensor signals and the high-pass filtering of the gyroscope signals. The overall sensor fusion and filtering look like the figure shown in Figure 3.

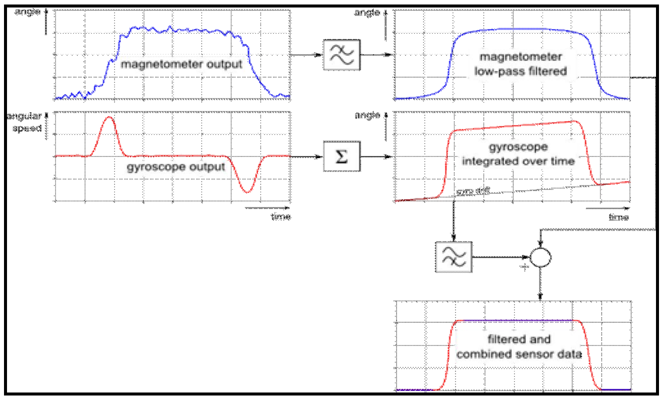


Figure 4: Result of the Sensor Fusion: filtered and combined sensor data

Figure 4 is an example that shows the application of sensor fusion of the 9 DOF sensors being turned 90 degrees in one direction and after a short time turned back to its initial position. In the said example, notice the gyroscope signal. These differences add up during integration and create an extra unwanted slow rotation of the gyroscope- orientation [10]. There were also few sensor fusion methodology available like Kalman Filter but it requires computational powers. The complementary filter requires less computationally intensive but gives an acceptable result.

2.2 Implementing Two Input FLC

Two-input FLCs in this study is based on the concepts of linear PD control schemes. The two fuzzy input variables used are the error e and the change-of-error ec and the fuzzy output variable used is Δu (incremental control input). Table 1 shows the PD-Type control rules of thirteen linguistic values for control input with rules for roll and pitch while in Table 2 for yaw.

After several tests and observations, the values of Δu stay at Z fuzzy set that ranges from [-3 to 3] since the value of the change-of-error ec does not exceed the range [-3 to 3]. Hence, not all rules are being used as an output.

Table 1: Control Rule Table of Two-Input FLC for the Roll and the Pitch

ΔU \ e / ec	PUltra	PMega	PSB	PB	PM	PS	Z	PS	PM	PB	PSB	PMega	PUltra
PUltra	Z	PS	PM	PB	PSB	PMega	PUltra	PUltra	PUltra	PUltra	PUltra	PUltra	PUltra
PMega	PS	Z	PS	PM	PB	PSB	PMega	PUltra	PUltra	PUltra	PUltra	PUltra	PUltra
PSB	PM	PS	Z	PS	PM	PB	PSB	PMega	PUltra	PUltra	PUltra	PUltra	PUltra
PB	PB	PM	PS	Z	PS	PM	PB	PSB	PMega	PUltra	PUltra	PUltra	PUltra
PM	PSB	PB	PM	PS	Z	PS	PM	PB	PSB	PMega	PUltra	PUltra	PUltra
PS	PMega	PSB	PB	PM	PS	Z	PS	PM	PB	PSB	PMega	PUltra	PUltra
Z	PUltra	PMega	PSB	PB	PM	PS	Z	PS	PM	PB	PSB	PMega	PUltra
PS	PUltra	PUltra	PMega	PSB	PB	PM	PS	Z	PS	PM	PB	PSB	PMega
PM	PUltra	PUltra	PUltra	PMega	PSB	PB	PM	PS	Z	PS	PM	PB	PSB
PB	PUltra	PUltra	PUltra	PUltra	PMega	PSB	PB	PM	PS	Z	PS	PM	PB
PSB	PUltra	PUltra	PUltra	PUltra	PUltra	PMega	PSB	PB	PM	PS	Z	PS	PM
PMega	PUltra	PUltra	PUltra	PUltra	PUltra	PUltra	PMega	PSB	PB	PM	PS	Z	PS
PUltra	PUltra	PUltra	PUltra	PUltra	PUltra	PUltra	PUltra	PMega	PSB	PB	PM	PS	Z

Table 2: Control Rule Table of Two-Input FLC for Yaw

Δ U \ e / ec	NUltra	NMega	NSB	NB	NM	NS	Z	PS	PM	PB	PSB	PMega	PUltra
PUltra	Z	PS	PM	PB	PSB	PMega	PUltra	PUltra	PUltra	PUltra	PUltra	PUltra	PUltra
PMega	NS	Z	PS	PM	PB	PSB	PMega	PUltra	PUltra	PUltra	PUltra	PUltra	PUltra
PSB	NM	NS	Z	PS	PM	PB	PSB	PMega	PUltra	PUltra	PUltra	PUltra	PUltra
PB	NB	NM	NS	Z	PS	PM	PB	PSB	PMega	PUltra	PUltra	PUltra	PUltra
PM	NSB	NB	NM	NS	Z	PS	PM	PB	PSB	PMega	PUltra	PUltra	PUltra
PS	NMega	NSB	NB	NM	NS	Z	PS	PM	PB	PSB	PMega	PUltra	PUltra
Z	NUltra	NMega	NSB	NB	NM	NS	Z	PS	PM	PB	PSB	PMega	PUltra
NS	NUltra	NUltra	NMega	NSB	NB	NM	NS	Z	PS	PM	PB	PSB	PMega
NM	NUltra	NUltra	NUltra	NMega	NSB	NB	NM	NS	Z	PS	PM	PB	PSB
NB	NUltra	NUltra	NUltra	NUltra	NMega	NSB	NB	NM	NS	Z	PS	PM	PB
NSB	NUltra	NUltra	NUltra	NUltra	NUltra	NMega	NSB	NB	NM	NS	Z	PS	PM
NMega	NUltra	NUltra	NUltra	NUltra	NUltra	NUltra	NMega	NSB	NB	NM	NS	Z	PS
NUltra	NUltra	NUltra	NUltra	NUltra	NUltra	NUltra	NUltra	NMega	NSB	NB	NM	NS	Z

Legends:

The control rules above represents the following fuzzy sets: **PUltra** for Positive Ultra; **PMega** for Positive Mega ; **PSB** for Positive Super Big; **PB** for Positive Big; **PM** for Positive Medium; **PS** for Positive Small; **Z** for Zero ; **NUltra** for Negative Ultra; **NMega** for Negative Mega ; **NSB** for Negative Super Big; **NB** for Negative Big; **NM** for Negative Medium.

The 13 Linguistic Values are the following:

1. PUltra = Fuzzy Set (14, 17, 17, 80)
2. PMega = Fuzzy Set (11, 14, 14, 17)
3. PSB = Fuzzy Set (8, 11, 11, 14)
4. PB = Fuzzy Set (5, 8, 8, 11)
5. PM = Fuzzy Set (2, 5, 5, 8)
6. PS = Fuzzy Set (0, 2, 2, 5)
7. Z = Fuzzy Set (-2, 0, 0, 2)
8. NS = Fuzzy Set (-5, -2, -2, 0)
9. NM = Fuzzy Set (-8, -5, -5, -2)
10. NB = Fuzzy Set (-11, -8, -8, -5)
11. NSB = Fuzzy Set (-14, -11, -11, -8)
12. NMega = Fuzzy Set (-17, -14, -14, -11)
13. NUltra = Fuzzy Set (-80, -17, -17, -14)

2.3 Software Implementation

This study used a fuzzy logic library to implement the controller. The library was validated through MATLAB and the result was closed enough between simulation (In MATLAB) and actual through Arduino Due microcontroller using Arduino IDE. For actual testing, the researchers use an open-source firmware to the ESC for a reverse mechanism of the brushless motor. To implement rotational control for the brushless motors to change its behavior by changing signal pulses called active braking or damped light. When active braking is not applied, as soon as the controller reduces the throttle to its minimum frequency of rotation in any direction, the motor does not stop immediately but it will continuously spin until friction from the propeller hits the air to slow it down until it stops.

2.4 Hardware: Stability Test

To test the fuzzy logic stability code, the researchers designed and made a prototype of a gimbal proportional to the gondola. The prototype gimbal is 3 dimensional and can simulate the roll, pitch, and yaw of the gondola as shown in Figure 5. Another arrangement used to test the stability code is by hanging the gondola five feet above the ground with rope to suspend the setup and manually tilting the gondola to a certain degree or angle. For its instrumentation, Adafruit AHRS (Altitude and Heading Reference System) is used to provide faster and precise orientation data. The AHRS takes this data further, converting it into heading or direction in degrees.

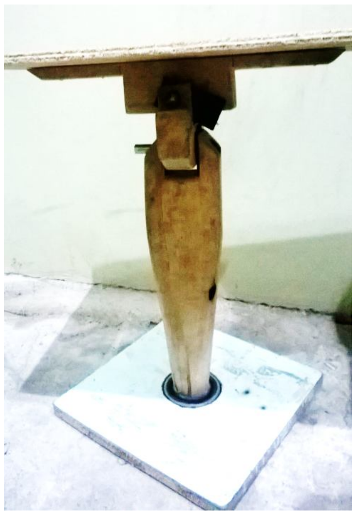


Figure 5: Gimbal Prototype

3. RESULTS AND DISCUSSIONS

A. Sensor Fusion Using Complementary Filter

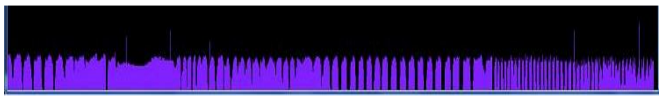


Figure 6: Raw Data from Sensor (Shake)

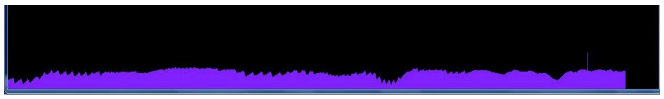


Figure 7: Filtered Data using Fusion (Shake)

Figure 6 shows that raw data when shook can cause undesirable output. It jumps from high to low value. On the other hand, Figure 7 shows the Sensor fusion gives smoother data even if shaken up and down.

B. Fuzzy Logic

Fuzzy Logic systems may be defined as a non-linear mapping of input data collection to scalar output data. The framework consists of four key components, such as fuzzifier, the fuzzy law, the inference engine, and the defuzzifier. In the initialization stage, define linguistic variables and terms and construct the membership functions and rule base. Then fuzzification, the conversion of crisp input data to fuzzy values using membership functions. Using the inference engine to evaluate the rules in the rule base and combine the results of each rule. To finish by defuzzification, conversion of the output data to non-fuzzy values [11].

C. Fuzzification

The Figure 8 to Figure 12 shows the six (6) input membership functions from the 3 dimensional axis (pitch, roll and yaw):

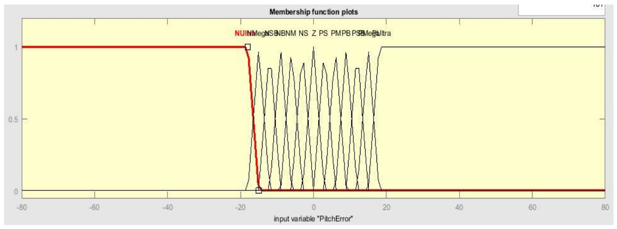


Figure 8: Pitch Error Raw Data from Sensor (Shake)

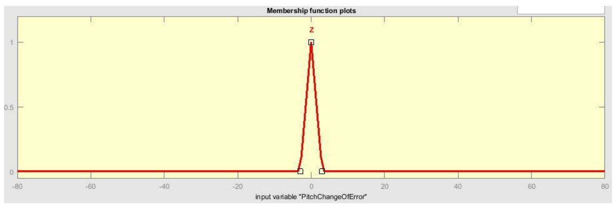


Figure 9: Pitch Change of Error

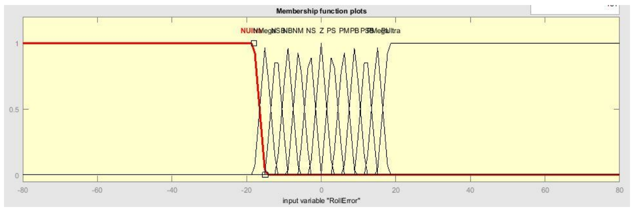


Figure 10: Roll Error

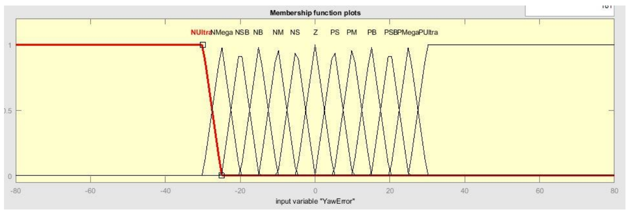


Figure 11: Yaw Error

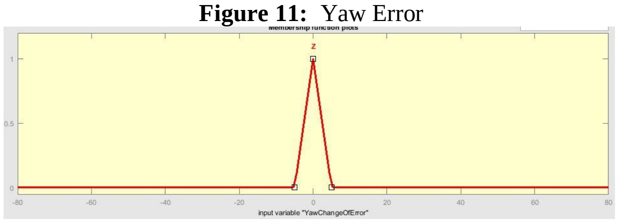


Figure 12: Yaw Change of Error

D. Defuzzification

The Figure 13 to Figure 15 shows three (3) output membership functions from 3 dimensional axis (pitch, roll and yaw): [Please refer to Table 1]

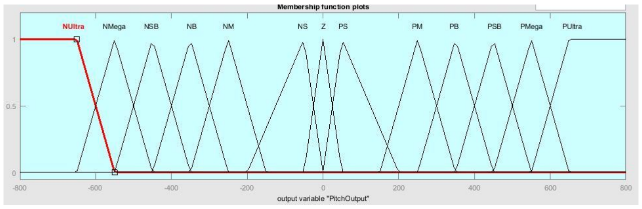


Figure 13: Pitch Output

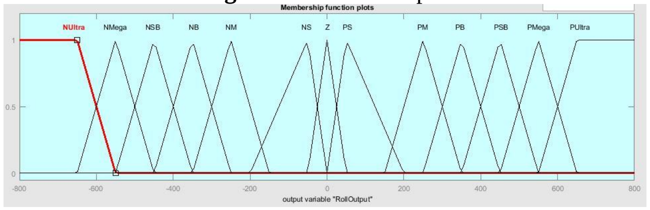


Figure 14: Roll Output

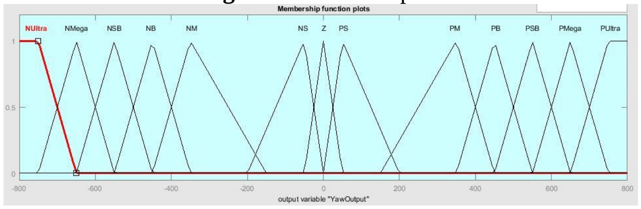


Figure 15: Yaw Output

E. Sample Input/ Output Result

The Sample Input and Output Results found on Figures 16 to 18 are from the input and output variables from 3 dimensional axis (pitch, roll and yaw):

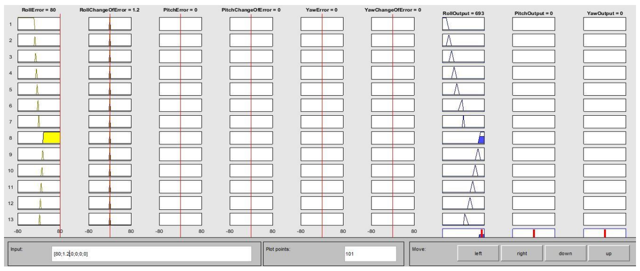


Figure 16: Roll Error =80, Roll Change of Error = 1.2, Roll Output =693

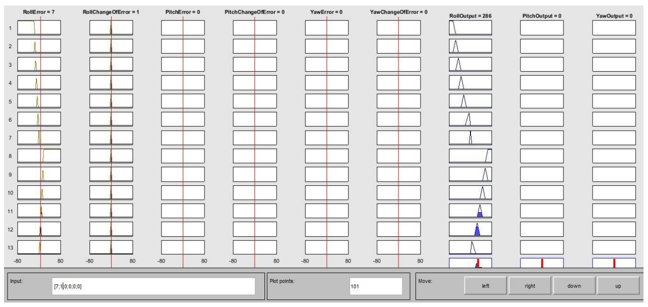


Figure 17: Roll Error =7, Roll Change of Error = 1, Roll Output =286

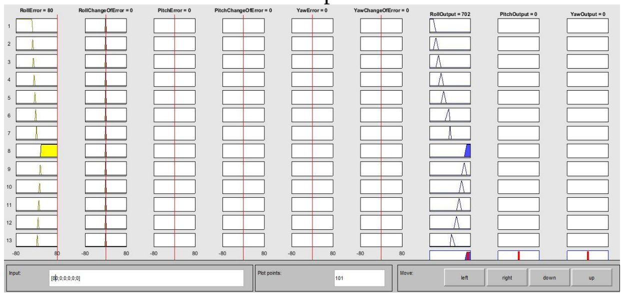


Figure 18: Roll Error =80, Roll Change of Error = 0, Roll Output =702

Table 3 shows the corresponding output for specific input from error (degrees) when comparing set point and sensor data. The output results with the unit of milliseconds (ms) are used to drive an electronic speed controller (ESC).

Table 3: Sample Input and Output from Actual Test

Roll Error	Pitch Error	Yaw Error	ESC1	ESC2	ESC3	ESC4
-27	1	4	1101	1101	1748	1748
-22	1	5	1127	1127	1731	1731
-15	1	6	1199	1199	1646	1646
-6	11	63	2000	2000	1223	1223
-4	11	67	2000	2000	1168	1168
1	2	-71	1788	1788	1048	1048
1	2	-70	1790	1790	1059	1059
0	3	-61	1794	1794	1023	1023
9	1	8	1230	1230	1332	1332
14	5	16	1544	1544	1536	1536

3.6 Actual Testing

The actual testing for the self-stabilization controller was done indoor, mounting the gondola to the gimbal to simulate air disturbances as shown in Figure 19. When the gondola tilts to any of the 3-axis directions, the controller provides signals to the motors to implement stabilization control based on the readings of its sensors. The feedback mechanism is based on fuzzy logic.

Response Time

Table 4 shows the response time using software implementation of damped light or active breaking in the ESC firmware versus the plain relay implementation. From the data gathered, the calculated percentage of the response time was 44.7% faster response from active braking.

Table 4. Response Time of Relay and Software Implementation (Damped Light/Active Breaking)

	Relay	Active Braking
Response Time	993 ms	**549 ms**

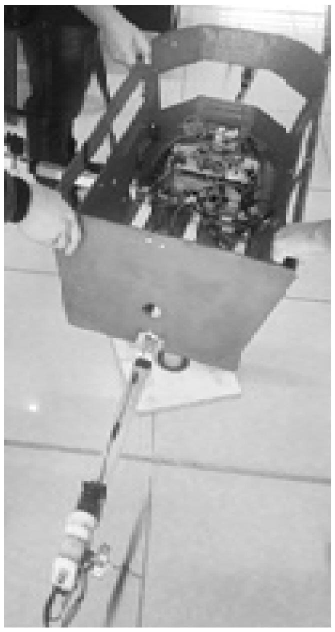


Figure 19: The Actual Set-up of Gondola mounted on the Gimbal

In the case of where relay was used and the brushless motor spins at its top speed, the rotation of the brushless motors fail to reverse as the motor was still spinning freewheeling. When active braking or damped light was enabled, the ESC actively stops the motor before spinning the brushless motor on the opposite direction. This allows faster response when there was a sudden change in the positioning as well as the rotation of the brushless motor.

CPU Utilization

Table 5 shows the CPU utilization of the Arduino Due by using the Adafruit's LSM9DS0 and Adafruit's BNO055 as well as its corresponding update frequencies. The update frequency determines how fast it can give orientation sensor data to the fuzzy logic algorithm. This was recorded using an Arduino built-in function called *millis()* which returns the time since the Arduino board began running the current program or part of the current program.

The result shows that Arduino Due has lesser CPU utilization on Adafruit's BNO055 compare to Adafruit's LSM9DS0. A lesser CPU utilization means that the lesser time of a specific task was executed in a given time was the lesser time for other task such as the fuzzy logic algorithm to wait before it is executed. Both sensors were tested using the same fuzzy logic algorithm code when testing the CPU Utilization and update frequency.

Table 5. CPU Utilization and Update Frequency of Adafruit's LSM9DS0 and Adafruit's BNO055

	Adafruit's LSM9DS0	Adafruit's BNO055
CPU Utilization Average	58.6%	**12.51%**
Update Frequency	6.55 ms	**1.376 ms**

Table 6 shows data that were gathered through testing, the ESC (E1, E2, E3, and E4) that uses active braking has a faster reaction time in terms of correcting errors.

Table 6. Data gathered with Active Braking

R	P	Y	Re	Pe	Ye	E1	E2	E3	E4
27	-1	-3	-27	1	3	402	402	744	744
23	-1	-4	-23	1	4	413	413	634	634
15	-1	-7	-15	1	7	425	425	620	620
6	-10	-64	-6	10	64	513	513	268	268
4	-11	-67	-4	11	67	501	501	223	223
-1	-2	72	1	2	-72	412	412	326	326
-2	-2	70	2	2	-70	416	416	338	338
0	-3	60	0	3	-60	409	409	311	311
-8	-1	-7	8	1	7	483	483	346	346
-14	-5	-16	14	5	16	499	499	596	596

Comparing data obtained and gathered from Table 3 and Table 6, significant increase in response time by **71.3%** for E1, **71.3%** for E2, **67.4%** for E3, and **67.4%** for E4 has been observed and calculated, as in (1).

$$\% difference = \left| \frac{newvalue - oldvalue}{oldvalue} \right| x100 \qquad (1)$$

Stability Testing Result

Tilting the gondola by certain degree or angle, the stability of the fuzzy logic algorithm of the flight controller is also calculated. This shows oscillations before it reaches its set points. In Figure 20, 21, and 22 shows that it took more than 600 samples before it reaches its set point and oscillations occurs. The default speed of the fuzzy logic controller ranges from 1490 as the minimum speed and 1880 as the maximum speed for the downward thrust and 1470 as minimum speed and 1070 as maximum speed for upward thrust. These speeds were based on signals that were accepted by the ESC range for its motor control for either a clockwise or counter clockwise rotation.

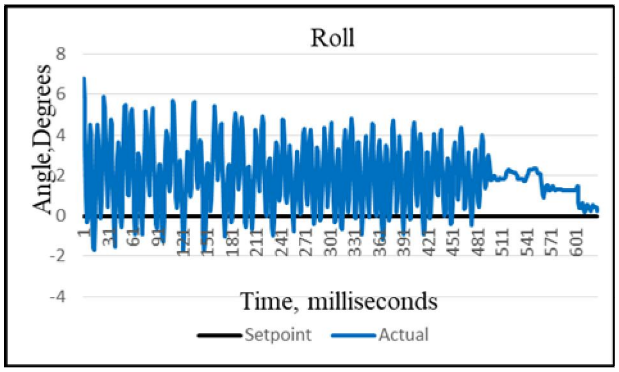


Figure 20: Roll Data Gathered When Tilting the Gondola to -3.5 degrees in Roll with a Minimum Speed of 1490 and Maximum Speed of 1880 for downward thrust and Minimum Speed of 1470 and Maximum Speed of 1070 for upward thrust

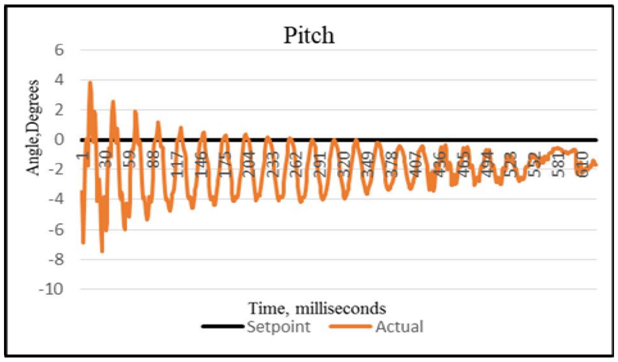


Figure 21: Pitch Data Gathered When Tilting the Gondola to 6 degrees in Pitch with a Minimum Speed of 1490 and Maximum Speed of 1880 for downward thrust and Minimum Speed of 1470 and Maximum Speed of 1070 for upward thrust

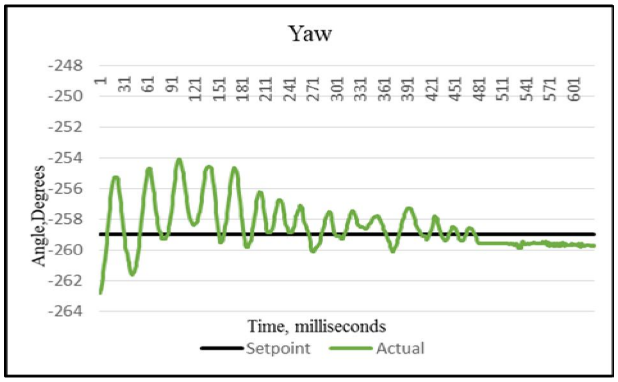


Figure 22: Yaw Data Gathered When Tilting the Gondola from -263 degrees in Yaw with a Minimum Speed of 1490 and Maximum Speed of 1880 for downward thrust and Minimum Speed of 1470 and Maximum Speed of 1070 for upward thrust to the set point of -259 degrees

There is a strong oscillation in the data obtained by gradually raising the minimum speed of the ESC by 100 such that it will produce further thrust than if the oscillations were to be reduced. Six (6) samples were measured, increasing the minimum speed by 100 for each sample before the average speed for the ESC of 1880 was achieved. As noted, and it reduced the oscillation and time to achieve the desired set points at each speed change. In Figure 20, 21, and 22, the data obtained was that the minimum and maximum speed of 1880 for the downward thrust and 1070 for the upward thrust were added to each ESC, maxing out the speed that the motor could provide for both clockwise and counter clockwise rotation.

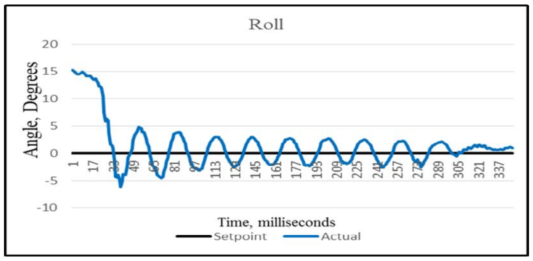


Figure 23: Roll Data Gathered When Tilting the Gondola to 15 degrees Roll, 5 degrees Pitch, and 14 degrees Yaw with a Minimum Speed of 1880 and Maximum Speed of 1880 for downward thrust and Minimum Speed of 1070 and Maximum Speed of 1070 for upward thrust.

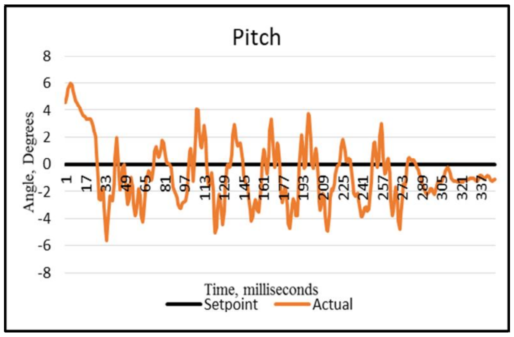
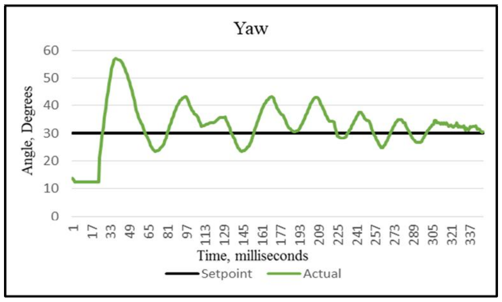


Figure 24: Pitch Data Gathered When Tilting the Gondola to 15 degrees Roll, 5 degrees Pitch, and 14 degrees Yaw with a Minimum Speed of 1880 and Maximum Speed of 1880 for downward thrust and Minimum Speed of 1070 and Maximum Speed of 1070 for upward thrust.



.**Figure 25:** Yaw Data Gathered When Tilting the Gondola to 15 degrees Roll, 5 degrees Pitch, and 14 degrees Yaw with a Minimum Speed of 1880 and Maximum Speed of 1880 for downward thrust and Minimum Speed of 1070 and Maximum Speed of 1070 for upward thrust

In the data samples obtained, gradually increasing the speed by 100 shows that the faster the motor speed and the higher the thrust, the lower the oscillation and the time for the stability controller to achieve the desired set points. In Figure 23, 24, and 25 were the data samples that have lesser oscillation and time among all the data gathered. Using percent error formula in (2) the result of 19.99% error in the stability from the data in Figure 21, thus the stability percentage of the fuzzy logic controller was 81.1%.

$$\% difference = \left| \frac{newvalue - oldvalue}{oldvalue} \right| x100 \qquad (2)$$

4. CONCLUSION

Fuzzy logic indicates better efficiency in the execution of control systems as it mimics human decision-making. In real-time systems such as blimp stabilization control, more precision is required, particularly when collecting data from sensors. When input data becomes noisy, the value of the error derivative adds to the system's unreliable value. With the error-correcting mechanism using fuzzy logic improves its performance. The implementation of active braking in the ESC firmware has resulted in a faster response time while using a unified sensor system, which has a faster update frequency, enhances the stabilization controller. This results lead to other flight stabilization controls using fuzzy logic technology on take–off control systems, landing control systems, and on course flight control systems [12].

ACKNOWLEDGEMENT

The authors wish to thank the leadership of the Computer Engineering Department of the University of Science and Technology of Southern Philippines (USTSP) for the support of this research undertaking. This study was made possible through the efforts of the following computer engineering students who conducted the investigative study and gathered data and to the faculty of CpE/ECE Department.

REFERENCES

1. Freudenrich, C., Ph.D. (2001). **"How Blimps Work"** February 26, 2001. HowStuffWorks.com. *Retrieved from http://science.howstuffworks.com /transport/flight /modern/blimp2*
2. Liu, Stirling, Pan, Naghd. (2009). **Control of Autonomous Airship**, in *Proc. 2009 IEEE International Conference on Robotics and Biomimetics*, Guilin, China, 2009, pp. 2457 – 2462. https://doi.org/10.1109/ROBIO.2009.5420403
3. Bhar, J., Barouni, S., & Bouazzi, I. (2016). **Multi-Level Fuzzy Logic Controller over different parameters variations.** *International Conference on Automation, Control. Engineering and Computer Science,* Tunisia Hammamet, 2016, pp.
4. Fuyin, D., & Weifeng, D. (2009). **Design of a Three-Input Fuzzy Logic Controller and the Method of its Rules Reduction**, in *Proc. 2009 International Symposium on Information Processing (ISIP 2009).* Huangshan, P. R China, August 21-23, 2009, pp. 51-53.
5. Alqudah, A. & Ashour,A. (2020**). "Controlling of Wind Turbine Generator System based on Genetic Fuzzy-PID Controller".** *International Journal of Advanced Trends in Computer Science and Engineering.* Volume 9, No. 1 February 2020 pp. 409 – 425. https://doi.org/10.30534/ijatcse/2020/58912020
6. Kumar, A. & Yoon, S. (2020) **"Development of Fast and Soft Landing System for Quadcopter Drone using Fuzzy Logic Technology".** *International Journal of Advanced Trends in Computer Science and Engineering,* Volume 9, No.1 February 2020 pp. 624 – 629. https://doi.org/10.30534/ijatcse/2020/87912020
7. Telen , M. E. & Guirnaldo, S. A.. (2017). **"Design Study of Gyro-stabilized, Remote-controlled Weapon Station".** *Mindanao Journal of Science and Technology, Volume 15 No.1. p.103-112.*
8. Choi, W. S., Hoang, N. M., Jung, J. H., & Lee, J. M. (2014). **Navigation System Development of the Underwater Vehicles using the GPS/INS Sensor Fusion.** In *International Conference on Intelligent Robotics and Applications* (pp. 491-497). Springer, Cham.
9. Patel, G. (2016). **Sensor Fusion** *Retrieved from: https://gunjanpatel.wordpress.com/2016/07/06/ sensor-fusion-via-complementary-filter-and-remove-gyro-drift-error/)*
10. Lawitzki, P. (February 18, 2014). **Android Sensor Fusion Tutorial.** Retrieved from http://www.codeproject.com/Articles/729759/Android-Sensor-Fusion-Tutorial
11. Mendel, J. M. (1995) **"Fuzzy Logic Systems for Engineering: A Tutorial,"** in *Proceedings of the IEEE,* vol. 83, no. 3, pp. 345-377, March 1995. https://doi.org/10.1109/5.364485
12. Africa, Aaron. (2019). **"Fuzzy Logic Control System with Gaussian Membership Functions"** *International Journal of Emerging Trends in Engineering Research.* Vol.7 No. 9 September 2019 pp. 328-332 https://doi.org/10.30534/ijeter/2019/16792019