A. R. Drone Parrot Controller

A. Herrera, A01323353. Student of ITESM, Puebla, Mexico; J. R. García, A01325155. Student of ITESM, Puebla, Mexico; K.A. Díaz, A01098497. Student of ITESM, Puebla, Mexico; L. E. Romero, A01097941. Student of ITESM, Puebla, Mexico and V. De Alva, A01324546 Student of ITESM, Puebla, Mexico.

***Abstract*— The following document describes the develop of a control system for an UAV parrot with the finality of flight in a stable mode. To reach this goal is necessary to obtain one model and the transfer function closer to the real system, propose the best controller and design a path flight. This system is composed by three parts: Fuzzy control, Vision system and the implementation of a Leap Motion Cotroller.**

Keywords— A. R. Drone, Parrot, Fuzzy Control, Vision system, Leap Motion, Gesture controller.

I. Introduction

Nowadays the application of unities like drones are basic, not only because the innovation of these, also because you can use it in almost every field. Controlling one of them from its core it's not an easy task, getting a faster time of response and functional stability for specific purposes causes problem during the development of more useful applications.

II. Objectives

The implementation of the following tasks.

VISION SYSTEM

The drone can follow some reference on the ground using the down facing camera doing real time tracking.

FUZZY CONTROL

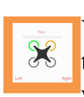
We controlled th behavior of the drone using Fuzzy logic techniques.

LEAP MOTION SENSOR

LEAP MOTION With the help of some gestures we can send a couple of specific instructions.

TAP PHOTOGRAPHY

Using the tap gesture we can take a Figure in the moment.



YAW AND HIGH CONTROL The drone is able the turn over the yaw axis also can go upwards and downwards and come back to the reference point.

III. Body

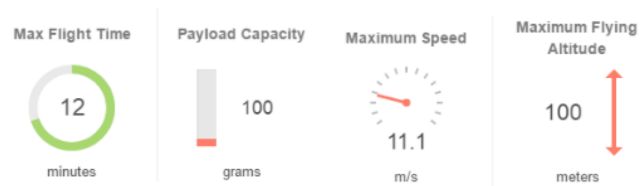
SPECIFICATIONS

HARDWARE

Drone Specifications

With a wingspan of 731mm, or 28.8 inches, this drone is on the larger end. The AR Drone 2.0 is above average in weight, weighing in at 420 grams or 0.9 pounds.

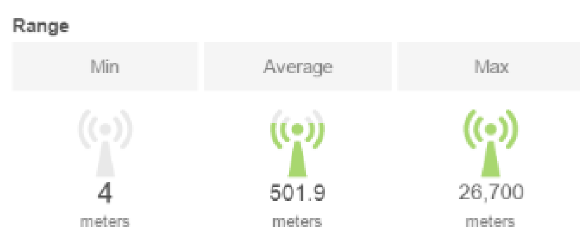
Dimensions	517 x 517 x 127 mm
Body Material	Carbon Fiber
Rotor Material	Plastic



Photography Details

Photo Formats	JPEG
Max Image Size	1280 x 720

The maximum distance away from the control system that the drone can fly.



SOFTWARE

LabVIEW Specifications

The drone was controlled using LabVIEW 2014 for the code. For the communication between the drone and labVIEW the AR Drone Toolkit version 0.1.0.36 was needed.

This toolkit includes the communication protocols, built in controller and camera manipulation.

For the Leap Motion communication the Makerhub Interface For LeapMotion Controller version 2.0.0.62 is needed. This toolkit provides communication protocols for the Leap Motion, motion and gesture detection for the hands

Leap Motion Specifications

The Leap Motion drivers must be downloaded from the official website when purchasing the Leap Motion Device. For this project we used the 2.3.1+31549 version of the software.

TECHNOLOGY USED IN THIS PROJECT.

IV. CONTROL

Why fuzzy control.

Tho find the right type of controller for a system you must know some parameters, once you have that value you can find the optimum plant model, So the advantages of fuzzy controller is not required the models and can be used in nonlinear process also it has high performance than PI controller. That why we decide to use this control technique.

Control Methodology:

- **Yaw Control**

The control of 3 degrees of freedom of the parrot was accomplished with fuzzy technique control. In order to manage the Yaw control it was necessary make a hover control. This control establish the Drone in a stable position with this set. The yaw control is easier to implement. In this paper, it was used the hover position that was preloaded on the board of the Parrot Ar Drone 2.0 by default.

In the Labview Code, it is a function called "automatic" that enable the hover position and controls the 4 speed rotors in order to reach the reference which is located on the Front Panel how can be seen in Figure 3. The fuzzy conditions are shown in the image 1, the output control receives values between -1 to 1 this parameter is set by the manufacturer. The sensor which gives the current yaw angle has a characteristic behavior: if the drone turns to the left the yaw angle moves from 0 to -180 if the drone turns to the right the angle moves from 0 to 180. It is notable that one point has two values: the point which has 180 degree value is the same which has -180 degrees. For this reason it was necessary mapping the yaw degree, to move between 0 to 360 degrees. To have more reference about the movement of the drone see figure 2, all parameter shown in image are set by the manufacturer.

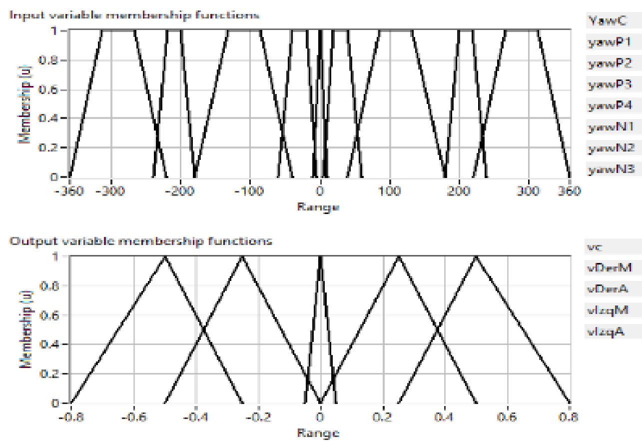


Figure 1. The first chart shows the values of the sensor, the second one shows the control values available for Drone.

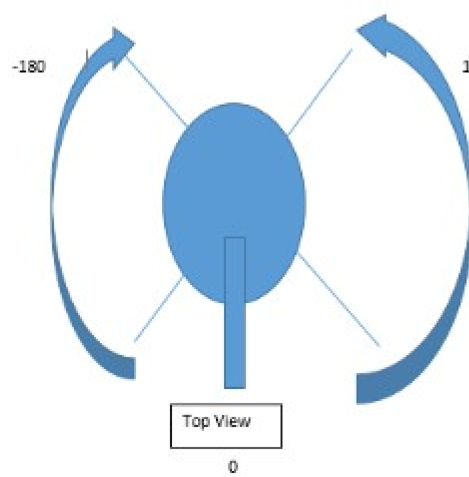


Figure 2. Top view of the drone.

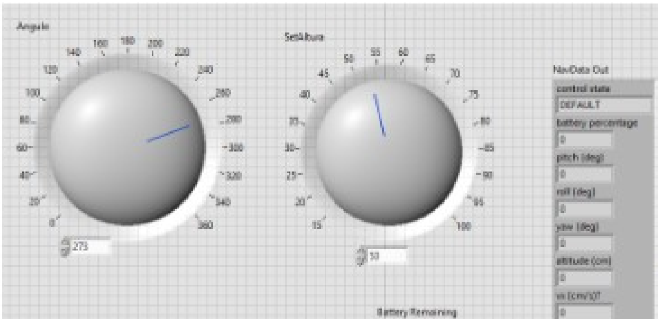


Figure 3. Labview controllers and indicators.

- **Height Control**

The height control was programed on fuzzy control technique too, like it was made to yaw control, the hover control must be set in order to handle successfully the Drone position.

In the function called "automatic" is enabled the hover position and it controls the 4 speed rotors in order to reach the reference which is located on the Front Panel how can be seen in Figure 3. The height sentences are shown in figure 4 and the testing of the control, the output control receives values between -1 to 1 this parameter is set by the manufacturer. The yaw and height control are separated by SubVIs but both of them controls the drone simultaneously how can be seen en figure 5.

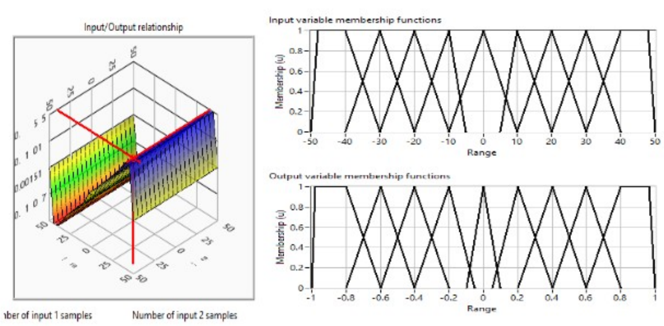


Figure 4. Control Rules.

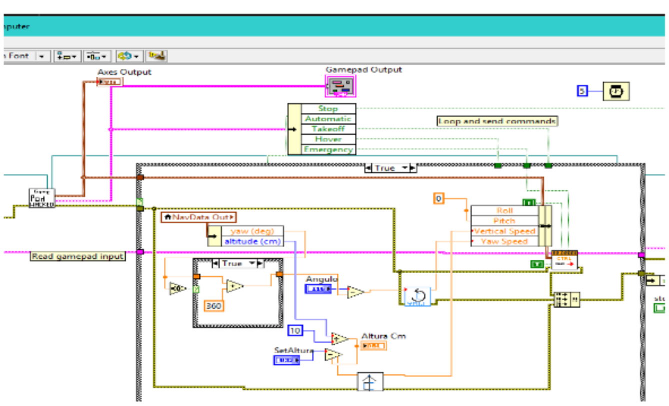


Figure 5. Labview Block Diagram.

V. LEAP MOTION CONTROL

With the yaw and height control programmed and working with both LabVIEW and an Xbox360 controller we proceeded to begin the Leap Motion configuration. In order to begin the implementation of the code we previously studied and worked with a number of examples from the MakerHub toolkit.

First we learned how to get the coordinates from thee hand using using the example for position and velocity as seen in figure 6. From this example we were able to obtain the xyz coordinates of the hand which were useful for controlling the movement in those directions.

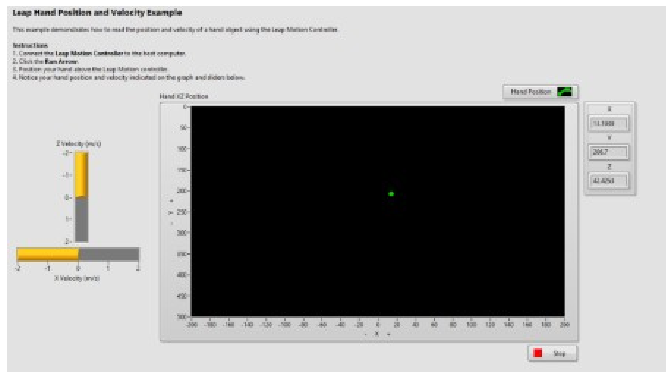


Figure 6. Leap hand position.

For the gesture control we analysed the swipe gestures and circle motion examples. In the swipe gesture we were able to read said motion and determine in which direction it was done. Likewise, the circle motion allowed us to tell if the circular motion was done in clockwise or counterclockwise sense. Figures 7 and 8 show the interface of these examples.

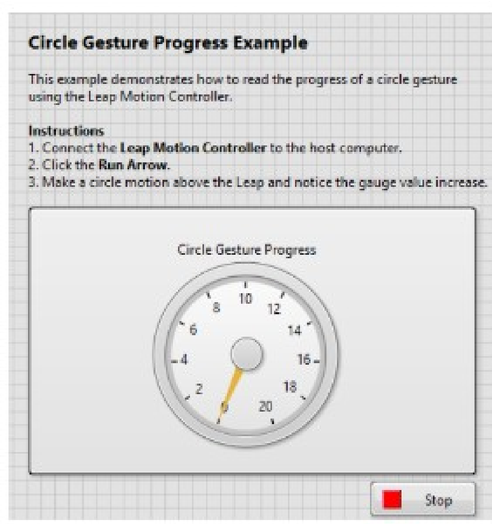


Figure 7. Leap gesture progress.

- **Control**

Take off/landing

To start the drone an upwards sweep motion was programed, the user needs to place the hand above the Leap Motion and perform the upward sweep to take off. In a similar fashion to land the drone one must perform a downwards sweep to safely land the drone in any moment.

The implementation of the control interface is the bridge between the drone and the Leap controller in which we had been working. The interface is divided in two forms of controls depending on the number of fingers detected on the hand: manual flight and yaw/height control.

Manual flight

The manual flight mode allows the user to pilot the drone with their hands using the position of the hand in the xyz axis to adjust the speed of the drone. The position of the hand is measured and a neutral set point is created to define a 'still' position. When the hand is in this coordinates the speed increment/decrement is null and the drone will keep its position on whatever axis the null coordinate is on.

When outside of these regions the drone will either increase or decrease its velocity on the axis providing movement in said axis. For example, if the hand is in the neutral y-axis position the drone will remain at its current height but, if the hand is risen the drone will begin to gain altitude until the hand is returned to the neutral set point and will remain at the new altitude.

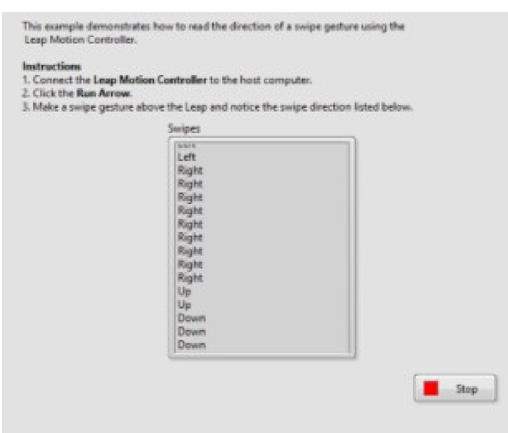


Figure 8. Swipe detections.

In order to enter manual flight, the hand must be completely open with the five fingers spread as shown in figure 9.

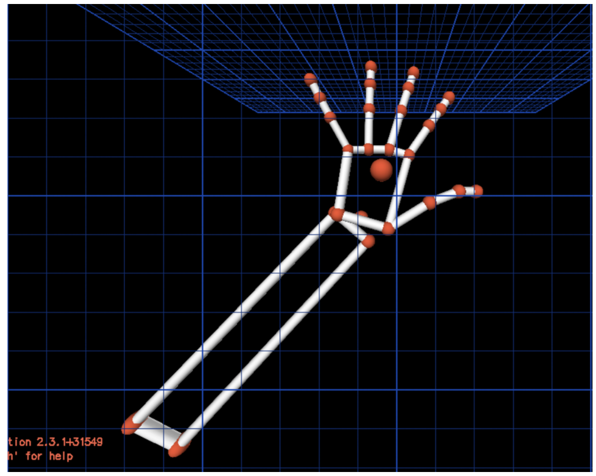


Figure 9. Hand projection.

Hover mode

The second mode, referred as 'Hover mode', allows the user to stop the movement of the drone, hover, modify the height and rotate the drone. This mode communicates directly with the fuzzy designed control that we previously discussed. Hover mode allows the control of two variables of the drone, the height and the rotation of the yaw. While in control of these movements the drone will remain in a hover state which means that there will be no control for moving forward, backwards or diagonally. To enter hover mode the user needs to place the hand above the leap motion in any way except the open configuration for manual flight.

Yaw control

The yaw control works with the circle motion. To enter yaw control the hand must create circular movements in either clockwise or counterclockwise depending of the direction of turn. The hand must be placed as shown in figure 10.

Thiscontrol allows for a complete 360° turn in any direction.

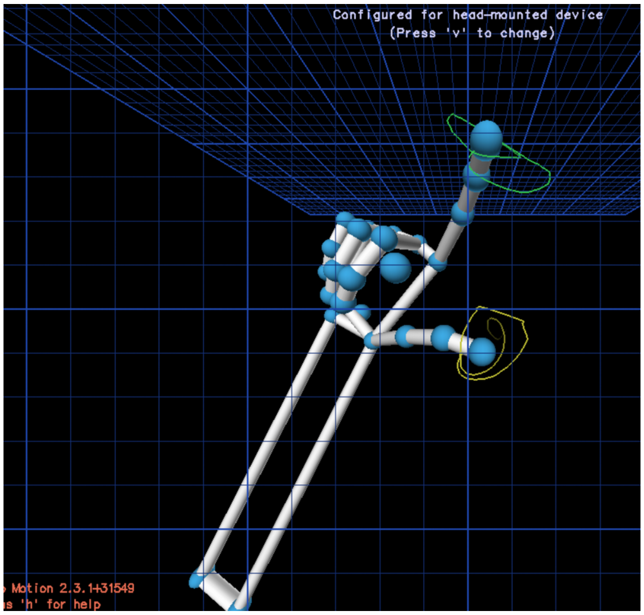


Figure 10. Swipe projection.

The control reads two values from the Leap Motion when translating the circular movement, the progress and normal. The progress variable is a counter of how many loops have been done since the circular motion was activated. The normal variable is a number which indicates the direction of the loops.

When processing these variables the progress variable is limited to a range of 0 to 360. The direction provided by normal controls if the progress variable will be increasing or decreasing. When the progress variable reaches the limit of 360°, any increment will lead to 0°, likewise reaching 0° and decreasing leads to 360°. Figure 11 shows the block diagram of these instructions.

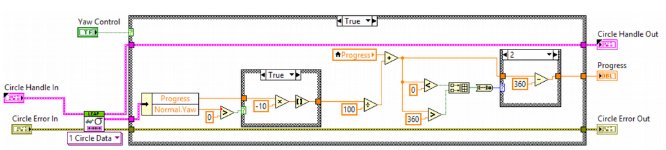


Figure 11.Labview Block Diagram.

The progress variable works as a set point for the fuzzy controller that was developed. By decreasing or increasing the progress setpoint the drone performs the turn to adjust to the new setpoint.

Height control

The height control works by mapping the vertical position (y-axis) of the hand. Unlike manual flight, when reading the position of the hand the drone doesn't adjust it's speed indefinitely, instead the height serves as a setpoint for the current height of the drone. To enter height control the hand must be placed as shown in figure 12.

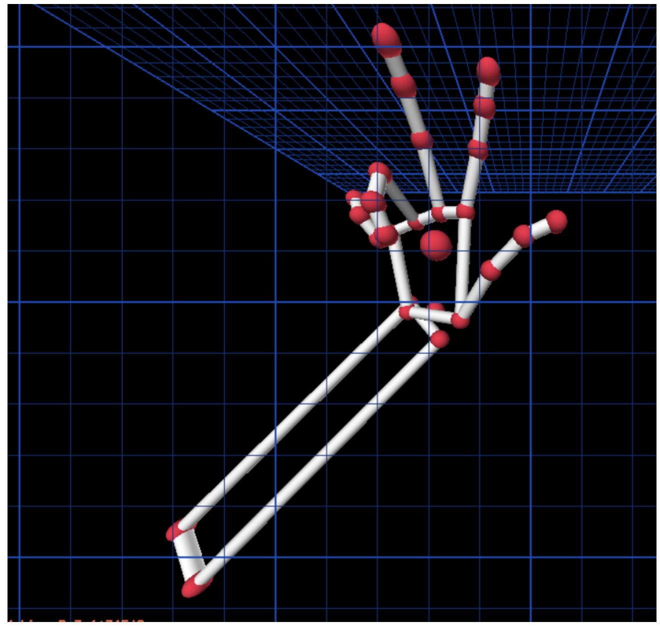


Figure 12. High gesture projection

The height code becomes similar to the yaw control, the y-axis position is read from the hand position and is mapped from 15 to 100, being able to achieve a maximum height of 1 meter.

The movement of the hand mustn't be too rash or the controller will read the movement as a take off/landing directive.

Final interface

At the end when creating the Leap Motion interface several indicators were placed to help the user keep track of the drone movement as seen in figure 13.

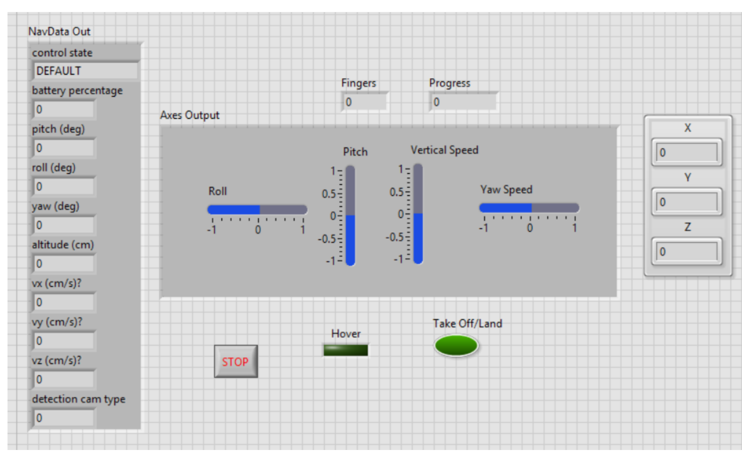


Figure 13. Labview Front panel.

The main indicators show the speed from the different movements of the drone to measure the response of the drone in manual mode. To the right we have the hand coordinates in the 3 axis. This indicator help to correct the position of the hand when in manual flight. In the top we've got to numerical indicators, the number of fingers for the different control modes and the progress variable for the yaw control setpoint. In the bottom the indicators for the hover mode and take off/land display are shown. Finally in the left we have the numerical values of the different moves that the drone performs. This helps to keep track of the variables for yaw control and height control.

VI. VISION CONTROL

The vision control is implemented using the Simple Video Vi provided by the Ar Drone Toolbox. For this goal it is necessary to access to the bottom camera, with this we will ensure that the Drone will keep tracking a target in order for the hover state to no to move around. The Drone once identifies the target, it will remain in the same position unless the target is moved, if that happens, the Drone will follow it until the end.

Two simultaneous while loops are running controlled by one stop button. One while loop will perform the Control Flight Sequence, and the other will do the object tracking.

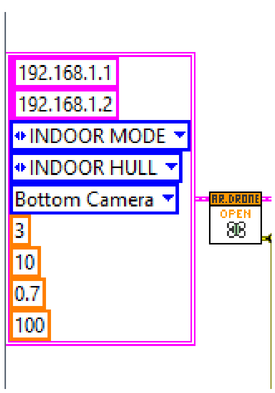


Figure 14. Labview Code.

A constant is created in the Setup Parameters in order to access to the bottom camera every time the VI its initialized.

Upper while Loop

The flight control sequence is commanded by the tools provided by the AR drone Toolbox. A button is created in order to access to the tracking control. If the button remains in off the default control is commanded by the recommended parameters provided by the Toolbox.

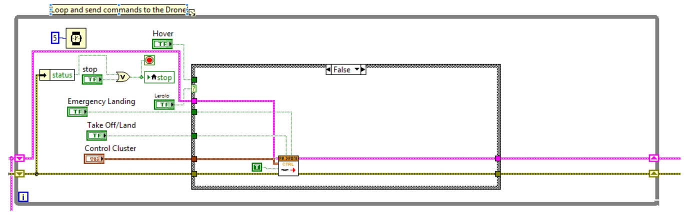


Figure 15.Labview Block Diagram.

Otherwise, when the Control Button is in the ON state, the sequence is commanded

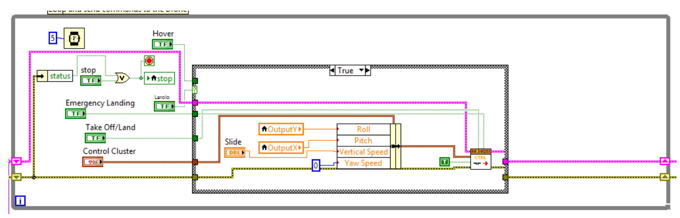


Figure 16.Labview Block Diagram.

Two local variable control the speed of the Pitch and the Roll, while the vertical speed is configurated by a control in the front panel by a sliding control. The yaw speed remains always in Zero. For this purpose of control the Drone will always be in Hover state, wheter with or without the button previously defined.

These two local variable OutputY and OutputX are continuously updated by the lower while loop.

The first sequence profrmed by the lower while loop is de video data acquisition. The image obtained by the button camera is processed by the following algorithm.

Figure 17. Algorithm

The binarization is essential for this purpose in order to detect only the bright objects, since the bottom camera keeps focusing to the floor, it's easier to recognize certain patterns. Then the shape matching tool help us to identify the desired pattern, for this goal we define an unique target in order for the control to not get easily confused by similar figures. The template created to identify is the next one.

Figure 18. Template.

Using this unusual figure instead a rectangle, a circle or any basic object, helps the Drone to remain in the same place, since depending on the floor or the objects around, it could be easily confused.

Using the process once the Algorithm identifies the object, the shape matching tools provides data that we use to determine the position of the Drone.

Knowing the resolution of the camera (360x180) a subtraction is executed with the coordinates given to know where the Drone is for the X axis as for the Y axis.

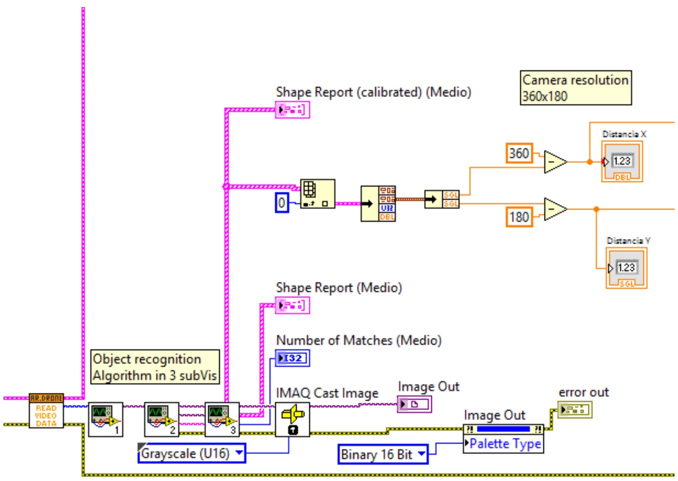


Figure 19.Labview Block Diagram.

Finally with the result obtained from the operation we define the area where the Drone. Using simple Fuzzy logic combined with the Formula Node element, we can control where the Drone should move in real time.

The established range goes from -150 to 150. 6 locations are defined.

Position (Result)	Location logic
-150 to -90	Far left
-90 to -45	Near left
-45 to 0	Center Left
0 to 45	Center right
45 to 90	Near right
90 to 150	Far right

This situations are evaluated using formula node in nested if statements.

If the location of the drone is far, the velocity impulse given to drone should be high, if near a middle impulse is required, otherwise, if center a slight impulse is given.

Location (Position)	Impulse to Drone	Drone translation
Far	High	+-0.5
Near	Middle	+-0.3
Center	Slight	+-0.2

This situation is evaluated for both axis, X and Y. this give us the specified position to control the Drone in the first while loop.

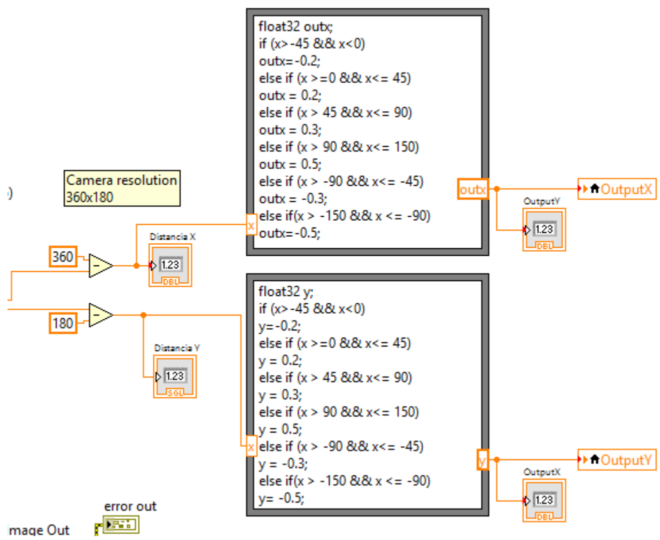


Figure 20. Labview Block Diagram.

In order to update in real time these values for the input control of the Drone, the output result of the operations in this fuzzy logic is sent to a local variable providing us this needed feature in the upper while loop.

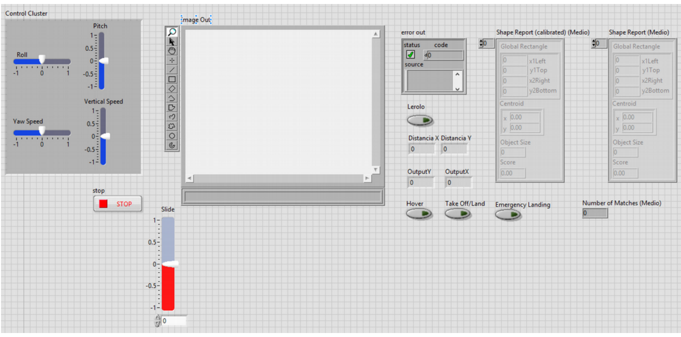


Figure 21. Labview front panel.

The Front panel shows the coordinates of the object recognized by the shape matching tool in the right.
The Image Out indicator helps the user to see what the Drone captures with the bottom camera, and the "Lerolo" button, in this case is the Control button that executes the sequence previously described.

V. Conclusion

The implementation of these Control approaches was handled by perfoming it in three VIs It is not possible to simultaneously run the three algorithm or group them all in the same VI because of the hardware limitations. Therefore to implement each algorithm was necessary two personal computer with this characteristics: 8 GB RAM memory, a processor Intel inside core i7-45100u @2.00 GHz 2.6 GHz The environment conditions are crucial in order to work properly for the vision algorithm. The final results were positive since each method work properly, controlling the AR Drone 2.0 Parrot in perfect way.

Finally we reach the objectives and controlled the behavior of the drone. Also we probe that is easier, faster and precise to apply a fuzzy control technique. We apply all the knowledge acquired in several courses, also we have to do some research by ourself letting us to acquired more information about several fields. The realization of this project is our first contact with a complete application I mean this project isn't a prototype is 100% finished.

VI. REFERENCES

National Instruments. Getting Started With the Leap Motion Controller. Retrieved from MarkerHub, https://www.labviewmakerhub.com/doku.php?id=learn:tutorials:libraries:leap:getting_started

Sakar, A. (2016, March 15). Gesture control of drone using a motion controller. IEEE

Parrot. (2016). Retrieved August 20, 2016, from http://www.parrot.com/usa/products/ardrone-2/

Niku, Saeed B., Introduction to robotics analysis, systems, applications, , Upper Saddle River, N.J. : Prentice Hall, New Jersey, 2001, eng, [0130613096]

Hvizdoš, J., & Sincák, P. (2015, January 24). Control library for AR.Drone 2.0. IEEE Xplore Digital Library.

A. M., & M. M. (2015). A hardware setup for formation flight of UAVs using motion tracking system. IEEE Conference Publications, 1-6.

Childers, B. (2014, May). Hacking the Parrot A.R. drone. Linux Journal, 2014(241).

F. S., Garrantt, M. A., & Anavatti, S. G. (2015). Fuzzy logic-based self-tuning autopilots for trajectory tracking of a low-cost quadcopter: A comparative study. IEEE Conference Publications,64-69. doi:10.1109/ICAMIMIA.2015.7508004

A. C., R. M., & X. B. (2016). Autonomous indoor object tracking with the Parrot AR.Drone. IEEE Xplore,25-30. doi:10.1109/ICUAS.2016.7502612

Awwalur, A., Imam, A., & Bharata, T. (2014, June 6). Path planning and formation control via potential function for UAV Quadrotor. Advanced Robotics and Intelligent Systems (ARIS), 2014, 165-170. Retrieved from IEEE Xplore.

Mary C, Totu L C, Koldbaek S K. Modeling and Control of Autonomous Quad-Rotor. Dept of Electronic Systems University of Aalborg Denmark. Project Report. 2010.

Koszewnik, A. (2014). The parrot UAV controlled by PID controllers. acta mechanica et automatica, 8(2), 65-69. http://www.degruyter.com/downloadpdf/j/ama.2014.8.issue-2/ama-2014-0011/ama-2014-0011.xml

Santana, L. V., & Brand, A. S. (2015, November). An automatic flight control system for the AR. Drone quadrotor in outdoor environments. In 2015 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS) (pp. 401-410). IEEE.

Indrawati, V., Prayitno, A., & Utomo, G. (2015, October). Comparison of two fuzzy logic controller schemes for position control of AR. Drone. In 2015 7th International Conference on Information Technology and Electrical Engineering (ICITEE) (pp. 360-363). IEEE.

SAKAR. A. (2016). GESTURE CONTROL OF DRONE USING A MOTION CONTROLLER. 2016 INTERNATIONAL CONFERENCE ON INDUSTRIAL INFORMATICS AND COMPUTER SYSTEMS(CIICS). IEEE.

GONZALEZ, H. G. (2016). SISTEMAS DE CONTROL EN TIEMPO CONTINUO Y DISCRETO.