

Grado en Ingeniería Informática

Curso 2021-2022

Trabajo Fin de Grado

**Detección y seguimiento de  
un objeto mediante visión artificial  
desde un dron de bajo coste comercial**

Enrique Ortiz Ibáñez

Tutores

Juan Pedro Llerena Caña

Jesús García Herrero

Escuela Politécnica Superior, 2021

## **RESUMEN**

Este trabajo se ha realizado con la idea de conseguir que, mientras el vehículo aéreo no tripulado de ala rotatoria esté volando de forma autónoma, sea capaz de reconocer la pelota de fútbol y pueda, de esta forma, seguir el desarrollo del partido.

El presente trabajo tiene como finalidad la creación de dos modelos de detección de una pelota de fútbol. Para ello, haremos uso de la inteligencia artificial y, así, podremos posteriormente analizar los dos modelos que vamos a estudiar, compararlos e implementar el más eficaz en el dron DJI Tello y pudiendo, posteriormente, llevar a cabo la función de seguimiento del dron del objeto. Los dos modelos de detección que se van a usar son YOLOv3 y Haar Cascade.

En la función de seguimiento, con el fin de simplificar parte de la investigación, se van a analizar los algoritmos de seguimiento de la librería cv2, siendo estos Boosting, MIL, KFC, TLD, MEDIANFLOW, CRTS y MOSSE.

Además, se va a realizar una pequeña investigación de cómo funciona el módulo de control PID (proporcional, integral y derivativo) y cómo aplicarlo al dron para que el movimiento de este sea lo más fluido posible.

Por tanto, como se puede observar, el objetivo principal de este trabajo es la construcción de un sistema de visión artificial, que realice un estudio de las imágenes obtenidas a tiempo real por parte de la cámara del dron y que el movimiento de seguimiento se lleve a cabo de forma fluida.

### Palabras clave

Dron; Visión artificial; OpenCV; cv2 YOLOv3; Haar Cascade; Vuelo Autónomo

## ABSTRACT

This work has been carried out with the idea of achieving that, while the rotary wing UAV is flying autonomously, it is able to recognize the ball of football and can, in this way, follow the development of the match.

The aim of this work is to create two models for detecting a ball of football. To do this, we will make use of artificial intelligence and, thus, we can subsequently analyze the two models that we will study, compare them, and implement the most effective in the DJI Tello drone and can subsequently carry out the function of tracking the object. The two detection models to be used are YOLOv3 and Haar Cascade, both requiring neural networks.

In the tracking function, in order to simplify the part of the research, the tracking algorithms of the cv2 library that are going to be analyzed, are Boosting, MIL, KFC, TLD, MEDIANFLOW, CRTS and MOSSE.

In addition, an investigation of how the PID (proportional, integral and derivative) control module works is going to be carried out and how to apply it to the drone so that the movement of the drone is as smooth as possible.

Therefore, as can be seen, the main objective of this work is the construction of an artificial vision system, which performs a study of the images obtained in real time by the camera of the drone and that the tracking movement is carried out smoothly.

Currently, despite the number of restrictions on drones, there is a growing use of drones growing use of drones.

In recent years, the demand for drones has increased by 370 %[1]. All of them can be used in a wide variety of fields, such as search and rescue of people, cinema and in the cinema and entertainment industry, in agriculture, ... they are also part of military and police operations... In any of these activities there are some steps that a drone must perform always, and that is that a drone must take off, fly and land.

Of these three tasks, nowadays the landing and take-off processes are normally automated, so the most interesting part that remains to be automated is the flight part.

Even though performing an experiment in a real environment with drones involves some higher costs than performing it by simulation, in this case, it was decided to carry it out in the real world since the necessary elements were available.

Finally, I would like to emphasize that the fundamental reason that has led me to investigate in this field of computer vision and drone control is due to the problems in this field that I had working for Fly-Fut [3].

In this work I manually manage a process in which the drone must focus all the time on the soccer ball and record the match that is being played. Also, many times, two people record two 7-football matches on a 11-football field. If this process were automated, the

work could be optimized because with only one person we would be able to record both matches.

It is obvious that automation process is a breakthrough in the world of technology. It is also obvious that this allows a process that requires the knowledge of several experts to be performed autonomously once the system has been created without error.

With this approach and linking it to the main problem of the thesis, we find an interesting solution to the lack of qualified drone pilots in Spain. Currently, to use a drone, even if it is for recreational use, you must have a title that certifies you to use drones. Since the main problem is the lack of drone pilots in Spain, a possible solution is to automate the drone flight process.

The work of tracking a ball in any type of sport exists, but this type of work is currently done from a static camera. Among these systems we find companies specialized in the recording of matches of any type of sport, but we do not have too much data about them since they are very recent systems. Some of these companies are Veo Technologies [4] and Pixellot [5] that perform the process of recording the match in 180 degrees and process it to obtain a video following the areas where the ball is located.

Regarding similar studies of object tracking with drones, we can find systems created with OpenCV, MAVSDK and PX4 [7] using object detection by colour segmentation. The company DJI has developed an object tracking system, called Active Track [8], which consists of manually selecting the region of interest.

There are also other types of drone tracking. These cases don't require manual selection, or automatic detection, simply the drone is in charge of following a transmitter from a set distance. This system is known as Follow-Me [9], a utility that connects the drone to a GPS-enabled device that acts as a transmitter and allows the aerial vehicle to follow a transmitter and allows the UAV to track the device.

There are many projects on the autonomous control of a drone that can obtain a good performance with the movement of the drone without having a person operating it. One of them is the software of the company Ptrotecma[10], capable of alternating between different flight plans and performing optimal movements despite the obstacles that may exist in the environment. This is done thanks to the data received by the sensors that the drone possesses.

In this project all legal aspects have been complied. Among them we find the regulations for the use of UAV (Unmanned Aerial Vehicles).

The drone flight regulations that have been followed have been those established by the Reglamento de Ejecución (EU) 2019/947 and the Reglamento Delegado (EU) 2019/945 [13]. Regulations that oblige drone users to register with the Agencia Estatal de Seguridad Aérea (AESA) whenever the drone weighs more than 250 g and/or has a camera.

When flying a drone, the data protection law must also be considered, as the people are being recorded during the flight. Therefore, it must be ensured that the images taken are down the Reglamento General de Protección de Datos (RGDP), which regulates how information is handled in the European Union.

Next, we will try to clearly define all the vocabulary that will appear throughout the project and that are related to it. clearly defined all the vocabulary that will appear throughout the project and that are related to it.

First of all, an Unmanned Aerial Vehicle is an aircraft, which does not need any pilot inside to be controlled and that control could be performed by remote control or by software, making it act autonomously.

Therefore, based on this definition, we can say that there are two types of UAVs depending on the type of control. Those controlled manually or controlled automatically. That is, those that are managed by remote control or by software.

There are different types of drones that can be classified in different ways. For the classification we can consider their use, their dimensions, their autonomy, ... However, the main classification of drones is based on their lift in the air being able to be fixed wing or rotary wing as helicopters or multirotor.

A drone during flight can perform four different types of movements: roll, the drone performs a horizontal movement from left to right, pitch, the drone performs a forward and backward movement, yaw, the drone performs a movement around the central axis of the aircraft, and finally, throttle, the drone performs a vertical movement from bottom to top.

Since the system is going to be developed around a drone, it is important to know that in order to be able to communicate with the UAV while it is in the air, a ground station or GCS (Ground Control Station) must be available. Normally, this ground station is used to monitor the performance, current position and control the camera of the UAV, all this in real time [27].

A ground station communicates with the drone wirelessly and can configure new parameters depending on the data received and the purpose for which the running software was made [27].

There exists many software that enable the use of computers, cell phones, tablets, etc. as if it were a ground station. Among these software's we can find some like MAVProxy, QGroundControl, AdroPilot, MAVPilot, etc [27].

The control station is connected to the drone through a communication protocol, encryption, and data compression. The sender of the data sends the compressed and encrypted data, which is received by the receiver of the drone that delivers the received information to the flight controller.

There are the following types of communication protocols:

- Pulse Width Modulation (PWM): It is the oldest communication protocol. This protocol is mainly used in radio-controlled cars and airplanes. It allows using a digital signal provided by a microcontroller, through channels, to operate a servo. Finally, we must emphasize that many cables are required for this protocol to work.
- Pulse Position Modulation (PPM): It is a protocol like PWM, but with the advantage that in this case only one wire is needed compared to the large number

of wires in the previous one. This protocol is considered analogy since it works by measuring the time of the states.

- Pulse Code Modulation (PCM): It is a digital signal protocol, capable of detecting an erroneous signal, which it will try to correct, so that the received data is correct. PCM is a more reliable protocol and, therefore, receives less interference than the previously mentioned protocols.
- Serial Protocol: uses multiple channels to avoid interference by making use of three wires, one for signal, one for ground and one for receiver power
- UDP Protocol: UDP (User Datagram Protocol) is one of the most important Internet protocols in existence, since it allows a set of datagrams to be sent with guarantees without establishing a connection to lower layers such as TCP or IP. Data transmission is carried out without establishing a prior connection, in other words, it is sent directly without prior notice. This leads to unreliable data transmission between the sender and receiver of the data.

What is meant by Artificial Intelligence? Defining “Artificial Intelligence” has become a debate within scientific organizations. There is not really an exact and unique answer about it but taking some of the ideas we could define it as “the combination of algorithms designed with the purpose of creating machines that present the same capabilities as human beings” [33].

On the origins of artificial intelligence, we can mark Alan Turing as one of the first references. We are around the 1930s, although the main starting point is considered to be the early 1950s, with the publication of “Computing machinery and intelligence”[34].

Artificial intelligence had a breakthrough in the 1980s with the introduction by John Hopfield and David Rumelhart of deep learning, which allowed computers to learn from their computers to learn from their experience [34].

Nowadays, AI seems to have no limits as it is used in a large number of fields in our day-to-day life. In addition, new fields in which it can be used are also emerging daily [34]. For example, some fields could be computer vision, data mining, planning, robotics, etc...

With respect to computer vision, we can divide it into detection and follow. Humans can look at an image or a video and distinguish what objects are in it. This task for a computer is much more complex. To solve this task, computer vision is used, which consists of teaching the computer to see. To do this, it is necessary to create different algorithms, that search for real-world objects from images taken before, with convolutional neural networks. A problem that exists for detection and that we can highlight can be the labelling of the object [35].

Algorithms can be classified into low and high level. With respect to low-level algorithms, we find the following:

- Template matching: this is an algorithm that receives a template with an object, in other words, a small image of what we want to detect. We want to detect the object of that template in a larger image. This comparison is made by the proximity of the details of the template and the image according to the coordinates of both. This algorithm is often used for example to detect the letters on a car number plate. [36]

- Background subtraction: this algorithm is used with a static camera and where the background of the image cannot have noise, in other words, areas of the image that are not important or that may resemble the object to be detected. It is often used to detect objects quickly and simply[36].
- Haar Cascade: machine learning algorithm for object detection. This algorithm was proposed in 2001 by Paul Viola and Micheal Jones in the article Rapid Detection Using a Boosted Cascade of Simple Features [37]. This article deals with the ability to detect faces in images. However, this system also allows the detection of other types of objects. To train this algorithm correctly, numerous weak classifiers called stages must be inserted in a cascade function. Each of these stages analyses a different part of the image, with a large number of positive and negative images from which an XML file is finally obtained. Through this file, objects in any other multimedia content can be detected, regardless of the size and location of the object in the image [38].

With respect to high-level algorithms, we find the following:

- R-CNN: To try to avoid the problem of selecting regions of interest from an image, Ross Girshick proposed a method that consists of using this selective search algorithm that can extract two thousand regions from an image [41]. The selective search algorithm generates a segmentation of the image and combines the smaller similar regions into larger regions [42]. Algorithm that needs a long time to be trained and to detect the object [41].
- Fast R-CNN: This algorithm solves some of the problems of the R-CNN. To speed up the learning process, instead of introducing the proposed regions into the convolutional neural network, the image itself is introduced to produce a convolutional feature map. And once they have created the map, regions of interest are obtained from the selective search algorithm, which are fed into the connected layers of the neural network [41]. But being faster than the previous algorithm alters the reliability of the algorithm to some extent.
- Faster R-CNN: An algorithm that does not use the selective search algorithm and instead allows the neural network to learn from the proposed regions [41]. It uses a separate neural network used to predict the regions of interest. It takes on average 0.2 seconds to run.
- YOLOv3: The YOLO (You Only Look Once) algorithm is a system whose main objective is the detection of objects in real time. For this purpose, convolutional networks are used together with deep learning, so that the image is divided into different  $S \times S$  regions. YOLO is a more efficient algorithm than the algorithms discussed above. It is capable of processing about sixty frames in one second.

A tracking algorithm is used to locate and follow an object in successive frames. To do this, there are seven different techniques which will be analysed below. The advantage that tracking algorithms have over detection algorithms is that they are faster, they do not need to analyse the entire image [45].

- Boosting: Tracker based on the Adaboost algorithm. It needs negative and positive images. To obtain these images, the region of interest is selected as a positive image and everything around it is converted into negative images. This algorithm,

when the detector returns the values where the object is located, gives each pixel a score according to where the pixel and the object are located. When the frame is changed, the object is located where the highest score obtained after analyzing the frame is located [45].

- MIL: algorithm like Boosting. The difference is that this tracker does not require the use of negative images, only positive images that are obtained through the region of interest obtained by the object detector. It also takes other regions close to the object and converts them into positive images as well. Has superior performance than Boosting and does not have as many errors.
- KCF: is a mixture of the previous Boosting and MIL algorithms. This algorithm uses the same technique as MIL of obtaining several regions of interest by superimposing one on top of the other and giving a score to each region of interest depending on the main region detected. And so, in the next frame this algorithm is given the maximum score and the updated bounding box [45].
- TLD: The algorithm tries to learn from the errors it obtains and tries to improve the error for the following frames. To carry out this process, it saves the different regions of interest it obtains and develops new tracking models[45].
- Medianflow: an algorithm that tracks the object by looking at its previous trajectories and trying to predict where it will go in the next frame. This algorithm gives accurate results when the trajectory of the objects is usually easy to predict, so the speed of these objects cannot be large. Unlike other algorithms, Median Flow reports correctly when it has stopped following the object [45].
- CSRT: algorithm that has some facility in finding the main features, location, and classes of an object. Experimentally, it is a tracker that has adequate results when working with object detection algorithms [45]. This is a more accurate algorithm than KCF, but a bit slower in terms of processing.
- MOSSE: algorithm useful for possible changes in light, deformation, and position of an object. This algorithm can detect when the object is not in the image based on a metric that measures the maximum sharpness of the correlation plane known as the peak-to-sidelobe ratio [45]. Like other algorithms discussed above, it is accurate and efficient when tracking one object, however, when tracking multiple objects in the same video, this algorithm loses accuracy.

Once, the football has been detected the drone had to follow the ball properly and for the flight to be smooth and without oscillations the PID (proportional, integral and derivative) controller needs to be explained.

The controller receives error values, which come from the subtraction between the reference values, which is the desired value, and the input values, which are the actual values of the system at a given time.

The PID controller consists of the sum of three gains that are completely independent of each other. Mathematically, this type of PID control system has the following equation:

$$K_p * Error(t) + K_i * \int_0^t Error(t) dt + K_d * \frac{Error(t)}{d(t)} \quad (1)$$



To correctly adjust the gains and thus achieve an optimal system behaviour, the PID parameters must be explained, as follows:  $K_p$ , in charge of reducing error and increasing the speed of response,  $K_i$ , to correct the error as soon as possible when the error is large and finally,  $K_d$ , if the system becomes very unstable, the tendency is to increase this constant to stabilise it [50].

To tune a PID controller there are different ways among them we can find some theoretical ways and some manual ways following some established heuristics. We find some methods as flat characterisation, application of tuning rules, auto tuning and manual adjustment by heuristics [51].

The next step is to talk about the proposal, which addresses the problem of tracking a football from a ground station, a laptop in this case, which communicates with the low-cost drone, Tello [53], from the company DJI [8]. For this purpose, a system has been designed capable of detecting and subsequently tracking the ball in a football field by autonomously controlling a drone.

To better explain the proposal, we will divide it into three main sections: the initialisation of the system, the sensor module and the control module.

The initialisation of the system consists of the connection between the computer and the drone via Wi-Fi. Once connected, the drone sends back to the computer the frames to be analysed and to be able to receive the commands to control the drone.

Subsequently, the programme has to be run, the first thing that is done is to activate the motors and start the camera, as in this drone we can activate and deactivate the operation of the camera depending on what we want to do.

Once we have reached this point, the next task is to take off the drone, a process that is carried out using the *takeoff()* function, which lifts the drone to 1.2 metres and then manually we raise the drone using the *move\_up(10)* function as many times as the  $\uparrow$  arrow on the keyboard is pressed until we reach the position where we have a good view of the football field. Once the altitude is reached, press the "a" key on the keyboard and the drone switches to autonomous flight.

The next module to be analysed is the sensor module, which is divided into two parts: detection and tracking.

With respect to detection, two models have been created to detect objects in images automatically. These models are YOLOv3 and Haar Cascade, although an algorithm to select the region of interest manually has also been used in the process.

In order to obtain an affordable model, several YOLOv3 models have been created, the first one was created using the Google database of images [55], which using a script provided by THE AI GUY [56], the images of the Football category have been downloaded, those images are used to create a model in charge of detecting where the ball of football is in the videos used and then we save the frames with their corresponding text file that have the bounding box measures ( $xmin$ ,  $ymmin$ ,  $xmax$ ,  $ymax$ ), measures that have to be normalised into YOLOv3 format.

Of the 500,000 frames of the videos, as the YOLOv3 model was not entirely good, only about 40,000 well-labelled frames were obtained. Despite this, it is a good image database for the creation of the model. These images were used to make three YOLO models with different learning rates (0.4, 0.01, 0.001). The model with the best training error is the third one, which uses a learning rate of 0.001.

Haar Cascade for the creation of this model, the instructions provided by Dasaradh Makes [58] were followed. Around 5,000 photos have been used, which, unlike the YOLO model, does not require many images and, in addition, the text files do not require any treatment.

For the development of this project, seven different trackers will be analysed: Boosting, MIL, KCF, TLD, MEDIANFLOW, CSRT and MOSSE.

Depending on whether the tracker has been detected good or bad, it returns a Boolean value True or False. If False is returned, the algorithm is made to re-detect the ball using either YOLO or Haar Cascade or the manual method. If, on the other hand, True is returned, the last section of the project, the control module, is passed.

Finally, the control module receives the information of the region of interest (x, y, w, h) and where the centre of the image is located, because that is where the ball will be always placed at the screen. For it to be in the required position, the error on the X-axis is calculated by subtracting the value resulting from dividing the width of the image by two from the x-value. The same has been done for the y-axis.

These errors have been calculated to calculate the speed using fixed values and values obtained to use the PID control formula to obtain values to be applied to the drone using the roll and yaw only.

Once the data is obtained in the yaw and roll, it must pass through a saturation module because the values returned are too high, causing the values to be outside the range of the function used and also serves to make the drone's movement more controllable.

Finally, these values are sent to the drone so that it moves and tries to place the ball correctly in the centre of the image.

Once the project has been explained in its entirety, it is time to validate the models obtained and the control values, to see if the behaviour of the drone when following the football is correct without having many oscillations or losses of the ball.

The library used, djitellopy, had movement functions intended only for movement in one direction of the axis, but it had a function (*send\_rc\_control()*) with which the drone was allowed to perform several movements at the same time. Finally, after testing the functions, it was observed that the use of the latter function was more efficient when sending commands to the drone.

For the validation of the detection system, three manual, YOLOv3 and Haar Cascade selected detection methods will be purchased. To see which of the three algorithms performs best, three different experiments have been performed, using three quality metrics Mean Inference Time, Mean Hit Ratio and Accuracy.

The values used to make a comparison and thus find out which algorithm is better were obtained from a two-minute video and analysed frame by frame with the three different algorithms.

The Average Inference Time consists of adding the time it took to detect each frame and dividing it by the total number of frames in the analyzed video and using this we have obtained the following graph.

Once the calculations were done, it was obtained that Haar Cascade is the algorithm that takes less time to run (40 ms) and on the contrary the manual selection algorithm is the one that takes more time (3700 ms) to detect the object.

Average Hit Ratio (RAM) this metric represents the number of soccer balls correctly detected in each frame and the number of balls that should have been represented. As in a soccer match there is only one ball, this metric consists of dividing the number of detections by the total number of images. In this metric we observe that the manual selection algorithm is perfect with 100 %, and the automatic learning algorithms have some failures, YOLOv3 and Haar Cascade with 85 % and 60 % respectively.

Finally, the last metric is the Accuracy of the algorithms which is evaluated with the terms True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). To obtain the precision, the sum of the true values is divided by the sum of all values. And the same thing happened again as with the average hit ratio once the operations are done: the manual selection algorithm is perfect (100 %) and YOLOv3 and Haar Cascade with lower values 89 % and 64 % respectively.

Therefore, although the manual selection is perfect in terms of accuracy, the time taken to detect the ball is very high, which makes us discard this algorithm. And although the time it takes YOLOv3 to detect the ball is higher than Haar Cascade, the accuracy of YOLOv3 is 25 % higher than Haar, with these data, the detection algorithm to be used in this project is YOLOv3.

The next step is to validate the tracking algorithms to know which tracking algorithm performs better than the rest. To see which algorithm is the best, four quality metrics will be used: Average Inference Time, Average Hit Ratio, Accuracy and Intersection over the Union of the detected region of interest.

The values that have been used to make a comparison and thus find out which algorithm is better, have been obtained from the same video that has been used in the validation of the detection.

To see which algorithm is better than another, part of the final script has been used, in which the object detection and tracking loop is used. The function of this loop focuses on whether the tracking algorithm returns the information that it has not been able to recognize where the ball is located. The detection algorithm that has been used is YOLOv3, being the algorithm chosen in the detection validation.

Having commented on how the process was carried out to obtain the data for the experimentation of the monitoring part, it is time to analyze the different experiments.

The first metric to be analyzed is the Average Hit Ratio, where Boosting, MIL, TLD or CSRT algorithms with results of 1.10 %, 6.13 %, 0 % and 7.37 % respectively, are very low results, without exceeding 10 %, so they could be discarded.

In terms of accuracy, the same is true as for the average hit ratio with Boosting (1.10 %), MIL (6.13 %), TLD (0 %) or CSRT (7.38 %) algorithms, which do not exceed 10 %. Therefore, in this situation, these can be discarded, and in this way reduce the comparison of algorithms from seven to three.

The intersection over the union is a metric that returns the goodness of fit of the region of interest obtained over the expected one using only the remaining algorithms KCF, MEDIANFLOW and MOSSE. This measure implies that a value close to 100% guarantees a good fit and a value close to 0% guarantees a null fit.

These three algorithms obtain equal maximum values of 100% at certain times. The biggest difference is in the minimum values and therefore in the average of these values between KCF and MOSSE with MEDIANFLOW. The latter barely exceeds 60% on average while the other two algorithms are above 70% fit.

But before discarding MEDIANFLOW let us analyze the Mean Inference Time with the three algorithms.

Despite the time of the MEDIANFLOW algorithm is the lowest of the three since it takes 29 ms to process the image. Finally, it will be discarded, since the second algorithm that takes less time is KCF with 86 ms, which together with its previous good results makes it the perfect algorithm for the development of the tracking action.

MOSSE has also been discarded because it takes about 800 ms to detect the ball in the image, which if we add the 800 ms by the detector makes it a very high value to be able to perform the detection and tracking in a smooth way, since it is close to 2 seconds.

For the last module of the system, the validation of the control module, a comparison of five measurements obtained through the heuristic used, which consists of the following: it starts with average values, if there is a large oscillation in time and the error is not stabilized in time the derivative part is decreased by half, because it can be a very high value and the values sent to the drone from one iteration to another can be very disparate causing the drone to have many oscillations from left to right.

Once it has been tested with the new derivative value, if there is still some evident oscillation, the proportional value will also be decreased by half. And this process is carried out until we find values with which the oscillation is minimal, and the error is stabilized. the error is stabilized.

To carry out this process and obtain the PID values to be used, we have made a script in which the YOLOv3 algorithm and KCF tracking are used as detectors, since they are the algorithms that have been previously selected as the most accurate for this project.

Once the procedure has been explained, the first values chosen to start with are [0.5,0,0.5] since they are average values and if the opposite of the established heuristic occurs, the opposite process is carried out.

Once seen that the values of the PID, [0.5,0,0.5], did not give a good result and following the heuristic, it has been decided to decrease the value of the derivative part to the half being this new value 0.25, its graph is the following one.

This process has been carried out until the appropriate value was reached so that there would not be too much oscillation on the part of the drone is [0.25,0,0.25]. Since the drone with values of [0.25,0,0.125] and [0.125,0,0.125] caused some oscillations because they wanted to solve the error very quickly and the movements produced were very sudden.

After having carried out the whole process of research, design, implementation and testing, some conclusions can be drawn to make an overall assessment of the final degree work.

Regarding the sensorization module, formed by two parts, the detector and the tracker. Three types of algorithms were analyzed for the detector, one of them selects the region of interest manually and the other two select the region of interest automatically (YOLO and Haar Cascade).

It was observed that the manual method, although very accurate, was very slow to process the image and was discarded. Therefore, comparing the automatic algorithms it was observed that YOLOv3 took about 20 times longer than Haar Cascade to detect the soccer ball; the former takes about 800ms and the latter 40ms and comparing accuracy and average hit ratio between YOLOv3 and Haar Cascade, it was obtained that the results of the YOLOv3 algorithm were better. In this situation, the delay of YOLOv3 in detecting the ball could be avoided.

With respect to the tracking algorithms, we first analyzed the average hit ratio and accuracy, in which the Boosting, MIL, TLD and CSRT algorithms do not exceed 10%, discarding them as possible solutions.

Boosting, MIL, TLD and CSRT algorithms do not exceed 10%, discarding them as possible solutions. Remaining only 3 algorithms and 2 metrics, where analyzing the intersection over the union, the algorithm with the lowest percentage is MEDIANFLOW, which although the average inference time is very low, comparing it with the KCF algorithm that has an affordable average inference time and its IoU is 72.29%. It makes it a good tracking algorithm making it the selected one.

Finally, it remains to draw conclusions from the last module, the control module. The values have been set heuristically, seeing how the drone reacted to the PID values set. Having said that the PID values that have been used for the correct flight of the drone have been [0.25,0,0.25]. Since the error it had compared to the rest of the values tested was lower.

So, the final values used are YOLOv3 and KCF as algorithms of detections and tracking and [0.25,0,0.25] for the module of control of the system.

## DEDICATORIA

Bueno, pues con este trabajo se acaba una de las etapas más importantes de mi vida en la cual me he formado como persona y aprendido muchas cosas gracias a la universidad. Esta etapa ha tenido sus altibajos como cualquier otra etapa, pero finalmente aquí me encuentro frente al ordenador escribiendo el trabajo de fin de grado. Y toda esta gran etapa no podría haberse desarrollado de la misma forma sin la ayuda de mucha gente a la cual me gustaría agradecerse.

Primero de todo me gustaría darle las gracias a mi familia, mi tía, mi tío y mi abuela, pero en especial a mis padres y mi hermana, que son los que sin ellos no habría llegado hasta donde estoy, ya que no me dejaron nunca que me rindiese en este duro camino.

A "Fondo Norte", el mejor grupo de amigos de la universidad que se puede tener. Sin ellos no hubiese sido lo mismo esta etapa, en especial a Alicia me llevo a una gran amiga de la carrera. Por esos viajes y esas fiestas que hemos tenido que nos sirvieron para desconectar un poco de la universidad y conocernos mejor. Por más momentos así. Y no me puedo olvidar de mis dos amigos que llevamos juntos desde el colegio David y Ortiz, los cuales me han servido de apoyo en todo momento. Y no olvidarnos de Victoria, una persona que me ha ayudado, guiado y apoyado muchísimo estas últimas semanas de desarrollo del TFG.

También me gustaría agradecerse a Fly-Fut, S.L. que gracias a ellos he podido realizar las pruebas de vuelo del dron y me han cedido videos para poder trabajar sobre ellos y de esta forma crear los modelos.

Por último, me gustaría agradecer a Juan Pedro Llerena Caña y a Jesús García Herrero, por los consejos y las aportaciones que me han hecho para poder desarrollar este trabajo. Y por aceptar la propuesta que os comenté desde el primer momento.

## ÍNDICE GENERAL

1	INTRODUCCIÓN .....	1
1.1	Motivación .....	1
1.2	Planteamiento del problema.....	2
1.3	Objetivos.....	2
1.4	Estructura del Documento .....	2
2	CONTEXTO SOCIO-ECONÓMICO.....	4
2.1	Marco Regulator .....	4
2.2	Contexto socioeconómico.....	6
3	FUNDAMENTOS TEÓRICOS Y ESTADO DEL ARTE .....	8
3.1	Estudios previos .....	8
3.2	Vehículo aéreo no tripulado.....	10
3.2.1	¿Qué es?.....	10
3.2.2	Tipos de Drones.....	11
3.2.3	Partes de un Dron .....	12
3.2.4	Navegación del Dron.....	13
3.2.5	Sistemas integrados .....	13
3.3	Estación de Tierra .....	14
3.4	Protocolo de comunicación.....	15
3.4.1	PWM – Modulación por Ancho de Pulsos .....	15
3.4.2	PPM – Modulación por Posición de Pulsos .....	15
3.4.3	PCM – Modulación del Código del Pulso .....	16
3.4.4	Protocolo Serial .....	16
3.4.5	Protocolo UDP.....	16
3.5	Inteligencia Artificial .....	16
3.6	Fundamentos de visión por computador.....	17
3.6.1	Detección .....	17
3.6.1.1	Algoritmos de Bajo Nivel .....	18
3.6.1.1.1	Comparación por plantillas .....	18
3.6.1.1.2	Sustracción del fondo .....	18
3.6.1.1.3	Haar Cascade.....	18
3.6.1.2	Algoritmos de alto nivel.....	20

3.6.1.2.1	R-CNN .....	20
3.6.1.2.2	Fast R-CNN.....	20
3.6.1.2.3	Faster R-CNN.....	20
3.6.1.2.4	YOLOv3.....	21
3.6.2	Seguimiento .....	22
3.6.2.1	Boosting .....	22
3.6.2.2	MIL.....	22
3.6.2.3	KCF .....	23
3.6.2.4	TLD .....	23
3.6.2.5	MedianFlow .....	24
3.6.2.6	CSRT.....	25
3.6.2.7	MOSSE.....	25
3.7	Controlador de estados.....	25
3.7.1	Controlador PID .....	25
4	PROPUESTA.....	28
4.1	Inicialización del sistema.....	28
4.2	Módulo de visión .....	30
4.2.1	Entrenamiento modelo YOLOv3 .....	31
4.2.2	Entrenamiento modelo Haar Cascade.....	33
4.3	Módulo de Control.....	34
5	VALIDACIÓN.....	37
5.1	Movimiento del Dron.....	37
5.2	Validación Detección.....	39
5.3	Validación Seguimiento.....	43
5.4	Validación Control.....	49
6	CONCLUSIONES Y TRABAJOS FUTUROS .....	54
7	GESTIÓN DEL PROYECTO .....	57
7.1	Planificación del proyecto .....	57
7.2	Presupuesto del proyecto .....	57
8	BIBLIOGRAFÍA.....	60





## ÍNDICE DE FIGURAS

FIGURA 2.1 CATEGORÍA ABIERTA [13] .....	5
FIGURA 2.2 MAPA ESPAÑA ENAIRE [14] .....	6
FIGURA 3.1 CÁMARAS VEO [4] Y PIXELLOT [5] .....	8
FIGURA 3.2 CÁMARA PIX4TEAM [6] .....	9
FIGURA 3.3 ENJAMBRE DRONES [12].....	10
FIGURA 3.4 VEHÍCULO AÉREO NO TRIPULADO [17].....	10
FIGURA 3.5 TIPOS DE MULTIRROTOR [20] .....	12
FIGURA 3.6 HELICES MULTIRROTOR .....	12
FIGURA 3.7 MOVIMIENTOS DE UN DRON [22] .....	13
FIGURA 3.8 ESTACIÓN DE TIERRA [28] .....	15
FIGURA 3.9 OBJETO DETECTADO .....	17
FIGURA 3.10 OBTENCIÓN DE LAS CARACTERÍSTICAS DE UNA IMAGEN .....	19
FIGURA 3.11 DIAGRAMA CLASIFICADOR EN CASCADA .....	19
FIGURA 3.12 PASOS ALGORITMO YOLOV3 [44] .....	21
FIGURA 3.13 EJEMPLO BOOSTING.....	22
FIGURA 3.14 EJEMPLO MIL .....	23
FIGURA 3.15 EJEMPLO KCF.....	23
FIGURA 3.16 EJEMPLO TLD .....	24
FIGURA 3.17 EJEMPLO MEDIANFLOW.....	24
FIGURA 3.18 CONTROLADOR PID.....	25
FIGURA 3.19 DIAGRAMA CARACTERIZACION PLANA [52].....	26
FIGURA 4.1 SECCIONES PRINCIPALES .....	28
FIGURA 4.2 PROPUESTA GENERAL.....	29
FIGURA 4.3 INICIALIZACIÓN INICIAL .....	29
FIGURA 4.4 VISTA DESDE EL DRON .....	30
FIGURA 4.5 MÓDULO VISIÓN .....	30
FIGURA 4.6 MÓDULO DE CONTROL.....	34
FIGURA 4.7 EJEMPLO ERROR .....	34
FIGURA 5.1 ALGORITMOS DETECCIÓN.....	39
FIGURA 5.2 COMPARACIÓN TIM DETECCIÓN .....	40
FIGURA 5.3 COMPARACIÓN RAM DETECCIÓN .....	41
FIGURA 5.4 COMPARACIÓN ACCURACY DETECCIÓN .....	42
FIGURA 5.5 DIAGRAMA ESQUEMATIZADO BUCLE DETECCIÓN .....	43
FIGURA 5.6 ALGORITMOS SEGUIMIENTO .....	44
FIGURA 5.7 DIAGRAMA ESQUEMATIZADO BUCLE SEGUIMIENTO .....	45
FIGURA 5.8 COMPARACIÓN RAM SEGUIMIENTO.....	46
FIGURA 5.9 COMPARACIÓN ACCURACY SEGUIMIENTO.....	47
FIGURA 5.10 DIVISIÓN IOU .....	47
FIGURA 5.11 COMPARACIÓN IOU SEGUIMIENTO.....	48
FIGURA 5.12 COMPARACIÓN TIM SEGUIMIENTO .....	49
FIGURA 5.13 VALOR PID 0.5,0,0.5 .....	51
FIGURA 5.14 VALOR PID 0.5,0,0.25 .....	51
FIGURA 5.15 VALORES PID 0.25,0,0.25.....	52
FIGURA 5.16 VALORES PID COMPARACIÓN .....	52
FIGURA 5.17 VALORES PID FINALES .....	53
FIGURA 6.1 DIAGRAMA FINAL.....	55
FIGURA 7.1 DIAGRAMA DE GANTT PLANTEAMIENTO .....	57
FIGURA 7.2 DIAGRAMA DE GANTT IMPLEMENTACIÓN.....	57
FIGURA 7.3 DIAGRAMA DE GANTT VALIDACIÓN Y ESCRITURA.....	57



## ÍNDICE DE TABLAS

TABLA 3.1 COMPARACIÓN ALGORITMOS BAJO NIVEL [36]	19
TABLA 4.1 MODELOS YOLOV3	32
TABLA 5.1 RESULTADOS DETECCIÓN	42
TABLA 5.2 RESULTADOS RAM SEGUIMIENTO	46
TABLA 5.3 RESULTADOS ACCURACY SEGUIMIENTO	47
TABLA 5.4 RESULTADOS IOU SEGUIMIENTO	48
TABLA 5.5 RESULTADOS TIM SEGUIMIENTO	49
TABLA 5.6 VALORES PID	50
TABLA 7.1 DESGLOSE PRESUPUESTO HARDWARE	58
TABLA 7.2 DESGLOSE PRESUPUESTO SOFTWARE	58
TABLA 7.3 DESGLOSE PRESUPUESTO RECURSOS HUMANOS	59
TABLA 7.4 DESGLOSE COSTE TOTAL	59



## **GLOSARIO**

DNS Sistema de Nombres de Dominio. 10

GCS Ground Control Station. 4

IP Internet Protocol. 10

PID Proporcional, Integral y Derivativo. 10

RPAS Remote Piloted Aircraft System. 5

TCP Transmission Control Protocol. 10

UAV Unmanned Aerial Vehicle. 5

UDP User Datagram Protocol. 10

VANT Vehículo Aéreo no Tripulado. 5

VPN Virtual Private Network. 10

SVM *Support Vector Machine*. 19



# 1 INTRODUCCIÓN

## 1.1 Motivación

Actualmente, a pesar de la cantidad de restricciones que tienen los drones, existe un creciente uso de estos.

En los últimos años, la demanda de drones ha aumentado un 370% [1]. Todos ellos pueden utilizarse en ámbitos muy diversos como puede ser la búsqueda y rescate de personas, en el cine y espectáculos, en la agricultura, ... , también forman parte de operaciones militares, policiales... En cualquiera de estas actividades hay unos pasos que un dron tiene que realizar siempre y, es que un dron tiene que despegar, volar y aterrizar. De estas tres tareas, en la actualidad se encuentran automatizados los procesos de aterrizaje y despegue, por lo que la parte más interesante que queda por automatizar de dicho proceso es la parte del vuelo.

Uno de los aspectos más importantes en el día a día de cualquier persona está relacionado con el campo de la tecnología. Podemos señalar que con los avances que se han producido en este terreno se está cambiando de forma radical la forma de vivir de la gente. Esto ha provocado que surjan nuevos campos de investigación como puede ser la biologización, la ingeniería digital o los múltiples campos en la inteligencia artificial como aprendizaje por refuerzo, sistemas distribuidos no homogéneos, así como el aprendizaje profundo donde podemos encontrar la visión artificial por computador [2].

La principal parte del proyecto se basa en la visión por computador. Para ello, vamos a llevar a cabo el estudio del procesamiento, el análisis y la obtención de diferentes características procedentes de imágenes o vídeos de forma que un ordenador pueda entender qué es lo que está analizando.

A pesar de que realizar un experimento en un entorno real con drones, supone algunos gastos superiores respecto a realizarlo por simulación, en este caso, se ha optado por llevarlo a cabo en el mundo real ya que se disponía de los elementos necesarios para ello.

Por último, quiero destacar que el motivo fundamental que me ha llevado a investigar en este campo de visión por computador y control del dron se debe básicamente a los problemas en este campo que tuve trabajando para Fly-Fut [3].

En este trabajo manejo de forma manual un proceso en el que el dron tiene que enfocar todo el rato a la pelota de fútbol, y de esta forma, grabar el partido que se está jugando. Muchas veces grabamos dos personas dos partidos de fútbol 7 en un campo de fútbol 11. Si este proceso estuviese automatizado, se podría optimizar el trabajo ya que con una sola persona se conseguirían grabar los dos partidos.

Por tanto, el problema planteado en este proyecto consiste en resolver la situación planteada anteriormente, en la que un dron pueda ser capaz de detectar dónde se encuentra la pelota de fútbol en el campo y que el dron mantenga en pantalla todo el tiempo la pelota mediante procedimientos de control.

Este trabajo sirve como una aproximación al proyecto final, que será mucho más completo si los módulos de visión y de control fueran más precisos.



## **1.2 Planteamiento del problema**

Es obvio que la automatización de procesos supone un gran avance en el mundo de la tecnología. También es obvio que esto permite que un proceso que requiere del conocimiento de varios expertos se pueda realizar de forma autónoma, una vez que el sistema se haya creado sin error.

Con este planteamiento y uniéndolo al problema principal del trabajo de fin de grado, nos encontramos con una interesante solución a la falta de pilotos de drones cualificados en España. En la actualidad para poder hacer uso de un dron, aunque sea para uso recreativo, hay que disponer de un título que te acredite para poder usar los drones.

Dado que el principal problema es la falta de pilotos de drones en España, una posible solución es la automatización del proceso de vuelo de los drones para que su uso esté realizado por un solo piloto o, incluso, sin la necesidad de piloto, contando únicamente con una sola estación de control, aunque se utilicen uno o varios drones. De esta manera, se podría manejar un dron para hacer el proceso de seguimiento de la pelota de fútbol de forma automática sin que el piloto tenga que estar pendiente del vuelo y del partido.

## **1.3 Objetivos**

Principalmente he planteado este trabajo de fin de grado como el inicio del entendimiento de las aplicaciones que tienen la visión por computador. Y en concreto, nos centraremos en el mundo de los drones , particularmente, en el seguimiento de una pelota de fútbol de forma autónoma. Todo esto, nos lleva a que el dron se maneje solo y para ello, se va a realizar una pequeña investigación sobre los módulos de control.

Para poder desarrollar este proyecto vamos a desarrollar prototipos de software que sean capaces de reconocer la pelota y analizar qué prototipo actúa mejor. Con este fin, se va a tener que realizar un gran etiquetado de las imágenes para poder crear dos modelos que aprendan de forma correcta.

Posteriormente, se va a tener un primer contacto con el sistema de control PID, cuyo objetivo principal es aprender a ajustar los parámetros Proporcional, Integral y Derivativo. De esta forma, buscamos que sea capaz de eliminar en el tiempo las perturbaciones que puedan existir.

Una vez establecidos ambos objetivos, se realizará un código que ayude al dron a moverse para mantener la pelota encuadrada en la imagen para, finalmente, realizar una validación de todos los sistemas establecidos y verificar su funcionamiento.

## **1.4 Estructura del Documento**

La estructura de este documento consta de un total de siete capítulos cuyos objetivos se muestran a continuación:

Capítulo 1. Introducción del proyecto. - Se realiza una breve introducción del problema a resolver y se introducen las motivaciones y los objetivos que se pretende conseguir. También se incluye una breve estructura del documento.

Capítulo 2. Estado del arte. - Se explicarán los estudios previos que existen acerca del proyecto a resolver, el marco regulador de los drones en España y en qué contexto se encuentran actualmente los drones.

Capítulo 3. Fundamentos teóricos. - En este apartado se encuentran las bases teóricas necesarias para el desarrollo y la comprensión del proyecto.

Capítulo 4. Propuesta. - En este capítulo se explica el proyecto y cómo se han creado los distintos algoritmos, así como la parte de control del proyecto.

Capítulo 5. Validación. - Apartado destinado a la justificación de qué algoritmos y valores se han elegido para su uso en el proyecto.

Capítulo 6. Conclusiones y trabajos futuros. - Se expondrán las conclusiones del proyecto y líneas futuras a seguir en el desarrollo de este.

Capítulo 7. Gestión del proyecto. - Pensando en la planificación, establecer un presupuesto adecuado para desarrollar el proyecto.

## 2 CONTEXTO SOCIO-ECONÓMICO

### 2.1 Marco Regulador

En este apartado se explican aspectos legales que se han cumplido en el desarrollo de este proyecto. Entre ellas encontramos la normativa del uso de los RPAs/UAVs (vehículos aéreos no tripulados).

La normativa de vuelo de drones que se ha seguido ha sido la establecida por el Reglamento de Ejecución (UE) 2019/947 y el Reglamento Delegado (UE) 2019/945 [13]. Normativa que obliga a los usuarios de drones a registrarse en la Agencia Estatal de Seguridad Aérea (AESA) siempre que el dron pese más de 250 g y/o tenga una cámara. Esta aeronave deberá ir identificada con una placa ignífuga que tiene que contener los datos del fabricante, el modelo, datos del piloto y el número de serie.

Con esta nueva normativa se elimina la diferencia que existía entre vuelo profesional y vuelo recreativo, haciendo que cualquier vuelo actualmente tenga las mismas restricciones.

Entre estas limitaciones que tienen los drones en España encontramos las siguientes:

- El dron siempre volará en alcance visual VLOS (Visual Line of Sight) por parte del piloto, salvo si se vuela en categoría STS que el dron puede manejarse en BVLOS (Beyond Visual Line of Sight).
- El dron no podrá superar los 120 metros de altura en vuelo y no se puede acercarse a un aeropuerto o aeródromo a menos de 8 kilómetros.
- No vulnerar la Ley de Protección de Datos en las imágenes captadas.
- Un seguro de responsabilidad civil.
- Necesidad de la placa ignífuga.

Actualmente existen tres categorías de vuelo y para distinguir las tres categorías se tendrá en cuenta lo arriesgado que sea la operativa de vuelo entre ellas.

**Categoría abierta.** Aquellos tipos de vuelos sin riesgo que no requieren autorización. En este caso el dron debe encontrarse al alcance visual del piloto, no puede superar los 120 metros de altura, y si durante la operación aparece una aeronave tripulada se debe aterrizar lo antes posible y cuando se vaya la aeronave reanudar la operación. No se pueden sobrevolar poblaciones o grupos grandes de personas.

**Categoría certificada.** Está creada para los vuelos de alto riesgo, donde los protocolos a seguir son similares a los de la aviación tripulada. Se usa en drones que requieren el transporte de personas o mercancías peligrosas y miden más de 3 metros de largo.

**Categoría específica.** Son vuelos de riesgo medio que no necesitan declaración o autorización por parte de AESA, pero si necesita el envío de un estudio aeronáutico de seguridad a AESA.

Lo común para una persona, es que el vuelo que se vaya a realizar sea en categoría abierta, sin la necesidad de declarar el vuelo. En esta categoría es importante familiarizarse con el siguiente esquema.

LIMITACIÓN SUBCATEGORÍA	REQUISITOS DE AERONAVES	REQUISITOS DE PILOTOS
<b>A1</b> Se permite el <u>sobrevuelo</u> de personas ajenas a la operación	Construcción privada o previa a la norma de <250 g y < 19 m/s	Familiarizarse con el <b>manual de usuario</b> del fabricante
	Clase C0 (<250 g)	Familiarizarse con el <b>manual de usuario</b> del fabricante
	Clase C1 (<900 g y < 80J con e-ID y Geo-awareness)	Familiarizarse con el <b>manual de usuario</b> del fabricante Completar un <b>curso online</b> Superar <b>examen teórico online</b>
<b>A2</b> Se permite el vuelo <u>cerca</u> de personas ajenas a la operación Manteniendo, una distancia de seguridad (30 - 5 metros)	Clase C2 (<4 kg con low-speed, e-ID y Geo-awareness)	Familiarizarse con el <b>manual de usuario</b> del fabricante Poseer un <b>certificado de competencia de piloto remoto</b> , obtenido mediante formación y examen online, autopráctica y examen presencial
<b>A3</b> Operaciones en áreas donde <u>no se espera</u> poner en peligro a personas ajenas a la operación Manteniéndose a < 150 metros de áreas residenciales, comerciales, industriales o recreacionales	Construcción privada o previa a la norma de <25 kg	Lo mismo que la Clase C1 en A1
	Clase C2 (<4 kg con e-ID y Geo-awareness)	
	Clase C3 (<25 kg con e-ID y Geo-awareness)	
	Clase C4 (<25 kg)	

FIGURA 2.1 CATEGORÍA ABIERTA [13]

En España para poder realizar un vuelo hay que mirar las restricciones que existen en el terreno sobre el que se quiere realizar la operación. Además existen algunas áreas donde el uso de los drones está sujeto a permisos y condiciones especiales, por lo que, para saber qué zonas están permitidas o no existe la página web de Enaire [14], la cual establece las áreas divididas en categorías:

- Zona prohibida: Aquellos lugares cercanos a aeropuertos, zonas militares, gubernamentales, etc...
- Zona restringida: Aquellas zonas establecidas como parques naturales o lugares históricos a los que se les puede causar daños materiales o de fauna.
- Zonas peligrosas: Zonas donde se encuentran tendidos eléctricos, actividad industrial, áreas nucleares, etc...
- Zonas sensibles: Áreas donde el vuelo está restringido por ser corredores de aves migratorias o se encuentran gran cantidad de aves.
- Zonas de Control Aéreo (CTR): Áreas cercanas a aeropuertos y helipuertos.
- Zonas restringidas al vuelo fotográfico: por motivos de seguridad no está permitido el uso de cámaras.
- Zonas protegidas: Aquellos parques naturales, zonas arqueológicas, reservas naturales, etc...

A pesar de todas estas zonas, y viendo el mapa de España (FIGURA 2.2) con todas las zonas marcadas casi en rojo, si se solicitan los permisos a los estamentos requeridos y se coordina de forma adecuada con los helipuertos, aeropuertos, se puede conseguir volar en la mayoría de las zonas salvo por ejemplo las zonas prohibidas, en categoría abierta.

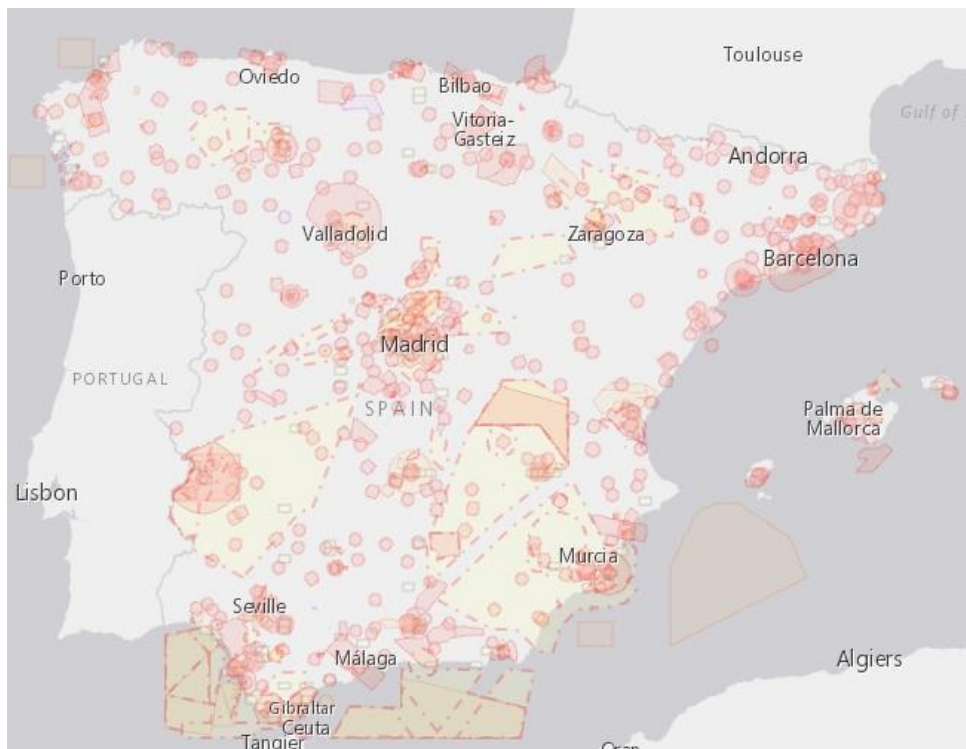


FIGURA 2.2 MAPA ESPAÑA ENAIRE [14]

También hay que tener en cuenta, a la hora de volar un dron, la ley de protección de datos, ya que durante el vuelo se está grabando a personas. Por lo tanto, hay que asegurarse de que las imágenes tomadas están sujetas al Reglamento General de Protección de Datos (RGDP), norma que se encarga de regular cómo se trata la información en terreno de la Unión Europea.

Esta normativa se creó con el objetivo principal de proteger el derecho a cualquier persona a mantener protegidos sus datos personales. Los pasos que se deben seguir para la adaptación al RDGP son variados, entre ellos se encuentra el consentimiento por parte del cliente, al cual se le debe informar de que van a ser grabados; y deben firmar un contrato sabiendo que se les va a grabar, etc...[15]

## 2.2 Contexto socioeconómico

Este proyecto se ha creado con el objetivo de automatizar un proceso el cual requiere de personas cualificadas y se puede realizar de manera autónoma. Este sistema desarrollado tiene un beneficio principal que es el seguimiento del balón ya que un humano puede distraerse con la jugada y perder de vista la pelota, pero el dron y la estación de tierra al estar funcionando continuamente con un buen modelo de seguimiento y de control, no pierde la pelota en ningún momento del partido.

Este proyecto tiene un impacto social, ya que las grabaciones desde el aire quedarían grabadas de forma diferente y a los jugadores les llamaría la atención la forma en la que se graban los partidos. Con este sistema se pueden ver ángulos que con una cámara estática con un trípode no se puede ver correctamente.

Por lo que ante esta situación se generaría una gran expectación por parte de los jugadores y, sobre todo, por parte de los entrenadores querrían adquirir dicho producto, para poder estudiar posteriormente los partidos con su equipo y analizar los que se ha hecho bien y

lo que se ha hecho mal. Se les podría ofrecer la contratación del servicio y se les podría grabar, Además, si hubiese dos partidos en un campo, se podrían colocar dos drones a la vez y de esta forma con un número de pilotos bajo, se podrían abarcar una gran cantidad de campos de fútbol.

Esta última medida, tendría un gran impacto económico, ya que aunque a priori tuviese unos costes superiores a los generados por grabar un partido con una cámara estática los resultados son mejores y los equipos lo acabarían amortizando.

Como inconveniente, se puede encontrar que siguiendo el marco regulador si el campo de fútbol se encuentra cerca de una zona restringida, prohibida, podría ser muy complicado conseguir los permisos de vuelo de los drones, por lo que solo existirían algunos campos de fútbol en los que se podría realizar la operación sin ninguna restricción.

### 3 FUNDAMENTOS TEÓRICOS Y ESTADO DEL ARTE

En este apartado se lleva a cabo una explicación sobre los proyectos que existen en la actualidad similares, así como los conceptos y términos usados durante el proyecto. La intención de esta introducción terminológica es dejar claramente definido todo el vocabulario que va a ir apareciendo a lo largo de todo el proyecto y que están relacionados con él.

#### 3.1 Estudios previos

Además de tener en cuenta los métodos actuales que se usan para realizar las grabaciones de partidos de fútbol, se van a analizar, también, otros tipos de trabajos similares en los que se esté usando la visión por computador.

El trabajo de seguimiento de una pelota en cualquier tipo de deporte existe, pero este tipo de trabajo actualmente se realiza desde una cámara estática. La mayoría de estos procesos se llevan a cabo a través de una cámara, que es capaz de abarcar como mínimo un ángulo de 180 grados, para, posteriormente, procesar el vídeo e ir seleccionando aquellos fotogramas donde se encuentra la pelota y recortar el ancho de vídeo, es decir, en vez de verlo de forma panorámica verlo de forma normal.

Entre estos sistemas encontramos empresas especializadas en la grabación de partidos de cualquier tipo de deporte, pero no se tienen demasiados datos de estas ya que son sistemas muy recientes. Algunas de estas empresas son Veo Technologies [4] y a Pixellot [5] que realizan el proceso de grabación del partido en 180 grados y lo procesan para obtener un vídeo siguiendo las zonas en la que se encuentra la pelota. Asimismo, Veo ha desarrollado un algoritmo que es capaz de realizar el seguimiento a tiempo real.



FIGURA 3.1 CÁMARAS VEO [4] Y PIXELLOT [5]

Otra empresa que realiza un proceso similar a Veo es PIX4TEAM [6] Esta empresa se diferencia de las otras dos mencionadas anteriormente en que para grabar los partidos se necesita una cámara de vídeo aparte. Dicha cámara se coloca en la parte superior de una cámara secundaria que gracias al motor que lleva es capaz de rotar e ir realizando el seguimiento de las jugadas.



FIGURA 3.2 CÁMARA PIX4TEAM [6]

A pesar de estos avances en la tecnología de seguimiento de las jugadas de fútbol con cámaras estáticas en las bandas de los campos, la visión que se obtiene desde el aire sobre las jugadas de fútbol es, para muchos analistas de equipos de fútbol profesionales, así como para los entrenadores, muy superior a la obtenida por una cámara estática.

En cuanto a estudios similares de seguimiento de objetos con dron podemos encontrar, sistemas creados con OpenCV, MAVSDK y PX4[7] mediante detección de objetos por segmentación del color. La compañía DJI tiene desarrollado un sistema de seguimiento de objetos, llamado Active Track [8], que consiste en seleccionar de manera manual la región de interés.

También, existen otros tipos de seguimiento con dron. Estos casos no requieren ni selección manual, ni detección automática, simplemente el dron se encarga de seguir desde una distancia establecida un transmisor. Este sistema se conoce como Follow-Me [9], utilidad que hace que el dron se conecte a un dispositivo con GPS que actúa como transmisor y permite que el vehículo aéreo no tripulado rastree dicho dispositivo.

Con respecto al control autónomo de un dron, existen muchos proyectos que son capaces de obtener un buen rendimiento con el movimiento del dron sin tener a una persona manejándolo. Uno de ellos es el software de la empresa Ptpotecma[10], capaz de alternar entre diferentes planes de vuelo y realizar movimientos óptimos a pesar de los obstáculos que puedan existir en el entorno. Esto lo realiza gracias a los datos recibidos por los sensores que posee el dron.

Otro tipo de software de control de drones es aquel que es capaz de crear estructuras en el aire con muchos drones, conocido como enjambre de drones. Esto se realiza a partir de softwares especializados para enjambres como puede ser el UgS DDC (Unmanned Ground Control Software Drone Dance Controller) [11]. Estos enjambres suelen ser utilizados para realizar coreografías en el cielo y sustituir los fuegos artificiales por los drones. Por ejemplo, estos enjambres se han usado hace poco para dar la bienvenida a un nuevo año en numerosos países y también se están usando para crear códigos QR en el cielo, repartiendo publicidad a un gran número de gente como se puede observar en la siguiente FIGURA 3.3.





FIGURA 3.3 ENJAMBRE DRONES [12]

## 3.2 Vehículo aéreo no tripulado

### 3.2.1 ¿Qué es?

Antes de definir que es un vehículo aéreo no tripulado vamos a fijarnos en su denominación y en las siglas que acompañan al nombre.

Por un lado, encontramos la terminología Vehículo Aéreo no Tripulado (VANT). Se trata de la opción española y sus siglas según su nombre en español. Y por otro lado, encontramos la versión en inglés RPAS (*Remote Piloted Aircraft System*) o UAV (*Unmanned Aerial Vehicle*) [1] .

En general, las agencias de seguridad aérea y los organismos oficiales prefieren usar estas últimas que provienen del inglés.

Utilicemos la versión inglesa o la española, sabemos que un vehículo aéreo no tripulado es una aeronave, que no necesita tripulación para ser controlado y que dicho control se ejerce mediante un control remoto o mediante un software, haciendo que este actúe de forma autónoma.

Por tanto, basándonos en esta definición, podemos afirmar que existen dos tipos de VANT dependiendo del tipo de control. Aquellos controlados de forma manual o controlados de forma automática. Es decir, los que se gestionan por control remoto o mediante un software.



FIGURA 3.4 VEHÍCULO AÉREO NO TRIPULADO [17]

Coloquialmente, a estos vehículos aéreos no tripulados se les denomina dron. El término dron proviene del término anglosajón *drone*, que tiene su origen en el nombre que se le asigna a la abeja macho o zángano (*mala bee*).

Al UAV se le denominó dron por la similitud que tiene su sonido con el zumbido de las abejas e, incluso, por lo molesto que ambos sonidos pueden llegar a ser. Finalmente, este término se popularizó y se extendió para hablar de ellos [16].

Los primeros usos de los drones en el mundo están relacionados con su aplicación en el ámbito militar. La primera aparición, de la que tenemos constancia, fue durante la I Guerra Mundial. Por supuesto, estos primeros drones se controlaban a distancia de forma remota, frente a los drones actuales capaces de realizar este tipo de vuelos de forma autónoma [18].

Posteriormente a la I Guerra Mundial y después de comprobar las posibilidades que brindaban, así como el éxito que este tipo de vehículos comenzaban a tener en sus misiones, empezaron a surgir empresas que decidieron invertir fuertemente en el desarrollo de nuevos sistemas que pudieran exportarse a los UAV.

De esta forma, en la actualidad, los drones, no solamente, tienen como finalidad única ser usados para escenarios de combate, sino que también se venden y distribuyen para otros fines de lo más variado como la toma de fotografías y videos principalmente. Gracias a que esta tecnología posee una cámara, se usa, también, en escenarios como el control de incendios, búsqueda de personas, estudios topográficos, usos agrícolas, así como otros muchos campos tanto lúdicos como profesionales.

### **3.2.2 Tipos de Drones**

Existen diferentes tipos de drones que se pueden clasificar de diversas maneras. Para la clasificación podemos tener en cuenta su uso, sus dimensiones, su autonomía, ... Sin embargo, la principal clasificación de los drones se basa en su sustentación en el aire. Teniendo en cuenta en este criterio la clasificación sería la siguiente [19]:

- Ala fija: Estos tienen sus alas similares a las alas de un avión, es decir, sin movimiento. Su principal ventaja es su gran autonomía y, gracias a su eficiencia aerodinámica, pueden permanecer en el aire durante largos periodos de tiempo sin utilizar ningún motor para propulsarse, lo que los hace muy seguros en caso de algún fallo del motor. Sin embargo, la principal desventaja es que requieren de una gran superficie para poder despegar y aterrizar, como si de un avión se tratara.
- Ala rotatoria: Como su nombre indica, son alas que giran y según el tipo de giro que tengan se dividen en dos tipos:
  - Helicóptero: Actualmente, son los drones más versátiles que existen porque tienen una gran carga útil y poseen una gran autonomía, que viene proporcionada por poseer un solo motor con una gran hélice. La principal desventaja que tienen los helicópteros es que son muy complejos de crear a nivel mecánico.
  - Multirrotor: Son los drones más populares que existen hoy en día. Este tipo de aeronaves son muy polivalentes y además, gracias a su fácil manejo, son muy usados en la actualidad. Dependiendo de la cantidad de motores que posea el dron, existen diferentes modelos que podemos

observar en la FIGURA 3.5. Su principal desventaja es que tienen muy poca autonomía.

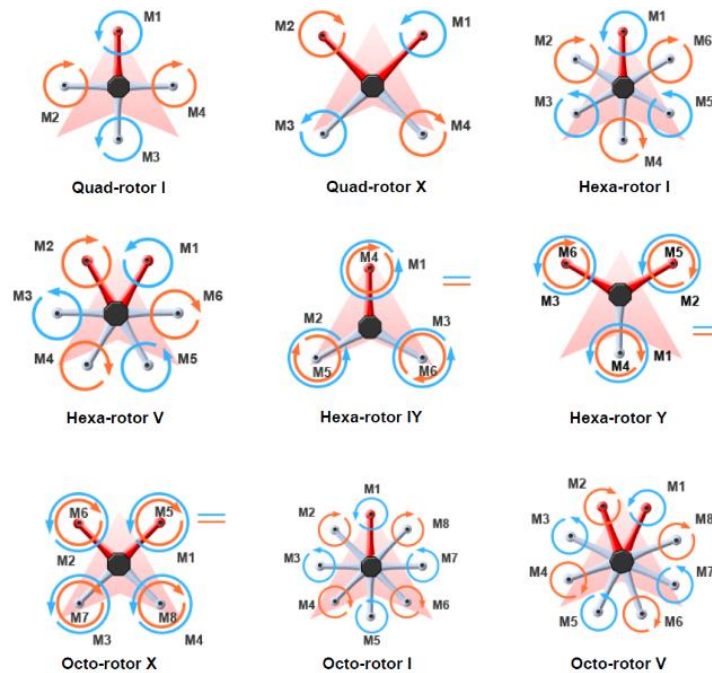


FIGURA 3.5 TIPOS DE MULTIRROTOR [20]

### 3.2.3 Partes de un Dron

Es muy importante conocer las partes de un dron, ya que regularmente, el piloto de un dron debe inspeccionar las partes de este y comprobar si están en perfecto estado y si no lo están, reemplazarlas por unas que si lo estén. Por ello se va a hacer una breve presentación de las diferentes partes de un dron [21].

- Chasis

Es la estructura central del dron. En él se van a ir implementado los diferentes componentes, por lo tanto, dependiendo, por ejemplo, de la cantidad de motores que tenga un dron, el chasis será de una forma u otra. Además, para intentar hacer el dron lo más ligero posible, pero resistente, se van a utilizar materiales ligeros como la fibra de vidrio o de carbono.

- Hélices

Las hélices son aquellas que se encargan de elevar a la aeronave por los aires. Por ejemplo, para elevar a un quadcóptero y obtener una buena estabilidad se necesitan 4 hélices. En este caso las hélices se distribuyen en pares dos a dos y tienen diferentes sentidos de giro.



FIGURA 3.6 HELICES MULTIRROTOR

- Motores

Los motores se encargan de hacer funcionar al dron. Ellos son los que hacen girar las hélices y de esta forma generar un empuje que permite al UAV volar. Hay diferentes tipos de motores que se diferencian por tamaño, potencia, velocidad, si son trifásicos o bifásicos, pero en un multirrotor todos los motores se colocan en los extremos del chasis.

- **Batería**

Una parte importante de los drones es la batería. Esta es la que proporciona la alimentación necesaria para que pueda volar durante un tiempo determinado. Normalmente, son del tipo Li-Po (polímero de litio) aunque existen otras como Ni-Cd (níquel-cadmio), Ni-MH (níquel-metal-hidruro) y Ion-Litio.

### 3.2.4 Navegación del Dron

En este apartado vamos a analizar cómo navega un dron.

Durante el vuelo, el dron realiza diferentes tipos de movimientos. Dichos movimientos se producen en los principales ejes del sistema cartesiano, conocidos como balanceo (*roll*), cabeceo (*pitch*) y guiñada (*yaw*) y el acelerador (*throttle*). Seguidamente vamos a explicar cada uno de estos movimientos:

- **Balanceo:** El dron realiza un movimiento horizontal de izquierda a derecha. Para ello el dron se inclinará en la dirección a la que quiere dirigirse.
- **Cabeceo:** El dron realiza un movimiento hacia delante y hacia atrás. Para ello el dron inclinará la cabeza hacia abajo o hacia arriba respectivamente.
- **Guiñada:** El dron realiza un movimiento en torno al eje central de la aeronave. Este se moverá en sentido horario o antihorario.
- **Acelerador:** El dron realiza un movimiento vertical de abajo a arriba. Para ello el dron acelera para ascender y desacelera para descender.

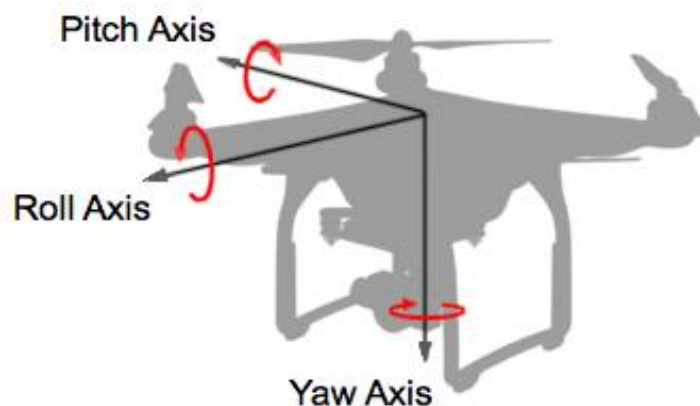


FIGURA 3.7 MOVIMIENTOS DE UN DRON [22]

### 3.2.5 Sistemas integrados

Un dron, como se ha explicado anteriormente, se puede utilizar para muy diferentes actividades/misiones en ámbitos muy diversos. Para poder realizar tan dispares actividades, todos estos drones tienen diferentes tipos de sistemas integrados, que ayudan a realizar aquellas tareas para las que han sido creados.

El sistema principal de un dron es el controlador de vuelo. Se trata del cerebro del dron. Dentro del controlador de vuelo encontramos componentes como la IMU, que usa el giroscopio y el acelerómetro para medir la aceleración y la rotación del dron, así como, el procesador, responsable de que todo funcione, desde los motores, la cámara, los sensores, la navegación. El procesador entre otras tareas es el encargado de dar la potencia necesaria a los motores. En dicho procesador encontramos los UART, encargados de enviar y recibir la información [23].

El GPS (Sistema de Posicionamiento Global) sirve, como su nombre indica, para determinar posiciones geográficas. Este sistema usa las matemáticas para marcar la posición exacta a través de triángulos, recibiendo el nombre de trilateración. El GPS en los drones, junto con la IMU, proporciona la información necesaria para controlar un dron. Entre las principales funciones del GPS en un dron encontramos el vuelo estacionario y el RTH (*Return to Home*) [24].

Una de las partes del dron que más usos tiene actualmente es la cámara. Esta es la responsable de que los drones tengan tantos usos en la sociedad de hoy en día. La cámara no solo sirve para tomar fotografías o vídeos, sino que también se puede utilizar para realizar actividades de seguimiento, reconocimiento de zonas, levantamientos topográficos, etc. Un dron puede llevar una cámara fija o puede tener cámaras modulares, es decir, sustituir una por otra. Además, para que la imagen sea transmitida de manera correcta, estas cámaras suelen estar unidas al dron a través de un gimbal, encargado de estabilizar las imágenes a través de motores que sirven para reducir las vibraciones generadas por el dron, el viento... [25]

Por último, y relacionado con el ámbito de la seguridad, en un dron encontramos los sensores de ultrasonidos. Estos sensores sirven para evitar que los drones colisionen contra alguna pared, contra un árbol, o contra cualquier elemento estático o móvil. Dichos sensores poseen una gran responsabilidad en la realización de vuelos autónomos, ya que permiten que desarrollen los trayectos esquivando los obstáculos que se pueden encontrar. Otro sistema de seguridad que poseen muchos drones es el paracaídas, que se activará cuando el dron se apague, o alcance una inclinación superior a 80 grados respecto a la horizontal. Estos paracaídas son imprescindibles cuando se quiere volar sobre una población a gran altura [26].

Finalmente, uno de los sistemas más sencillos son las luces leds que suelen llevar. Estas sirven para localizar dónde está la parte delantera del dron y la trasera en caso de estar volando en condiciones visuales adversas.

### **3.3 Estación de Tierra**

Dado que el sistema se va a desarrollar en torno a un dron, es importante saber que para poder comunicarse con el UAV mientras está en el aire, hay que disponer de una estación terrestre o GCS (*Ground Control Station*). Normalmente, esta estación terrestre se usa con el fin de monitorear el rendimiento, la posición actual y controlar la cámara del UAV, todo esto en tiempo real [27]. Una estación de tierra puede ser la mostrada en la FIGURA 3.8 o puede ser un ordenador, un móvil, una tablet....



FIGURA 3.8 ESTACIÓN DE TIERRA [28]

Una estación terrestre se comunica con el dron de forma inalámbrica y es capaz de configurar nuevos parámetros dependiendo de los datos recibidos y del fin con el que se haya realizado el software que se está ejecutando [27].

Existen gran cantidad de softwares que facultan el uso de ordenadores, móviles, tabletas, etc. como si se tratara de una estación de tierra. Entre estos softwares podemos encontrar algunos como MAVProxy, QGroundControl, AdroPilot, MAVPilot, etc [27].

### 3.4 Protocolo de comunicación

Un protocolo de comunicación es una forma de encriptación y compresión de los datos. Esto ocurre de la siguiente forma, el emisor de los datos envía los datos comprimidos y encriptados, los cuales son recibidos por el receptor del dron que entrega la información recibida a la controladora de vuelo.

Por lo tanto, para que un dron pueda realizar misiones de forma autónoma es necesario que exista un protocolo de comunicación entre el dron y una estación terrestre.

Existen los siguientes tipos de protocolos de comunicación:

#### 3.4.1 PWM – Modulación por Ancho de Pulsos

Es el protocolo de comunicación más antiguo. Este protocolo se utiliza principalmente en coches y aviones de radio control. Permite usar una señal digital proporcionada por un microcontrolador, a través de canales, para hacer funcionar un servo. En un dron este protocolo suele usarse para soltar algún tipo de carga, como por ejemplo, se pueden lanzar semillas en un campo o algún tipo de producto para fumigar [29].

Este protocolo que estamos analizando usa una frecuencia de 50Hz, lo que equivale a un periodo de 20ms, donde la única parte útil de estos 20ms son 2ms, lo que provoca que 18ms queden sin respuesta.

Por último, tenemos que recalcar que para que pueda funcionar este protocolo se requiere un elevado número de cables.

#### 3.4.2 PPM – Modulación por Posición de Pulsos

Se trata de un protocolo similar a PWM, pero con la ventaja de que en este caso solo se necesita un cable frente al elevado número de cables del anterior.

Este protocolo se considera analógico, ya que funciona midiendo el tiempo de los estados. Los canales son enviados uno tras otro haciendo que los 18ms de periodo desperdiciados sean reducidos dependiendo del número de canales usados [29].

### **3.4.3 PCM – Modulación del Código del Pulso**

Es un protocolo de señal digital, capaz de detectar una señal errónea, que intentará corregir, para que los datos recibidos sean los correctos. PCM es un protocolo más confiable y, por lo tanto, recibe un menor número de interferencias que los protocolos comentados con anterioridad [30].

### **3.4.4 Protocolo Serial**

Un protocolo Serial usa múltiples canales para evitar interferencias haciendo uso de tres cables, uno para señal, otro para tierra y otro para la potencia del receptor. Este tipo de receptor usa un puerto serial que pueden ser de varios tipos [30].

### **3.4.5 Protocolo UDP**

UDP (*User Datagram Protocol*) es uno de los protocolos de Internet más importantes que existen, ya que permite enviar con garantías un conjunto de datagramas sin establecer una conexión a las capas inferiores como TCP o IP.

La transmisión de datos se realiza sin establecer una conexión previa, es decir, se envía directamente sin previo aviso. Esto conduce a una transmisión de datos poco fiable entre el remitente y el receptor de los datos. Este protocolo no proporciona control de flujo, lo que significa que si un dispositivo es más rápido que el otro, la información puede perderse o los paquetes entrantes pueden estar incompletos [31].

En cuanto a las características principales de este protocolo encontramos que UDP funciona sin conexión a internet. Los datagramas enviados se mandan directamente a través de la dirección IP. Además, intenta mantener el protocolo de comunicación abierto entre el emisor y el receptor el menor tiempo de transmisión posible, lo que provoca que sea un protocolo rápido y sin retardos [32].

En cuanto a sus principales usos, este protocolo se utiliza, principalmente, para consultas DNS, conexiones VPN y para la emisión en directo de audio y vídeo. El proceso de la emisión de video en vivo es una transmisión de información continua y rápida, la pérdida de alguno de estos paquetes de datos puede causar cierta distorsión de la imagen, pero no afecta a la reproducción del vídeo.

## **3.5 Inteligencia Artificial**

¿Qué se entiende por Inteligencia Artificial? Llegar a definir “Inteligencia Artificial” se ha convertido ya en sí mismo en un debate dentro de las organizaciones científicas. Realmente no existe una respuesta exacta y única sobre la misma, pero tomando algunas de las ideas podríamos definirla como “la combinación de algoritmos planteados con el propósito de crear máquinas que presenten las mismas capacidades que el ser humano”. Según esta definición, cuando hablamos de inteligencia artificial nos referimos a una máquina que puede tomar decisiones propias, tal y como lo haría una persona, y para ello tendría que basarse en una serie de datos y algoritmos que previamente poseería [33].

Sobre los orígenes de la inteligencia artificial, podemos marcar como uno de los primeros referentes a Alan Turing. Nos encontramos alrededor de los años 30, aunque el principal punto de partida se consideran los inicios de los años 50, con la publicación de “*Computing machinery and intelligence*”[34].

La inteligencia artificial tuvo un gran avance en los años 80 con la introducción por parte de John Hopfield y David Rumelhart del aprendizaje profundo que permitía a los ordenadores aprender a través de su experiencia [34].

Actualmente, la IA parece no tener límites ya que se utiliza en una gran cantidad de ámbitos dentro de nuestro día a día. Además, también diariamente surgen nuevos campos en los que poder usarla, lo que permite optimizar una gran cantidad de procesos. Entre los ámbitos en los que actualmente se usa la Inteligencia Artificial encontramos:

- Visión artificial.
- Minería de datos.
- Planificación.
- Robótica.
- Seguridad.
- Reconocimiento del lenguaje
- Entre otros...

### **3.6 Fundamentos de visión por computador**

#### **3.6.1 Detección**

Los humanos somos capaces de observar una imagen o un vídeo y distinguir qué objetos hay en él. Esta tarea para un ordenador es bastante más compleja. Para poder resolver dicha la tarea se usa la visión por computador, la cual consiste en enseñar al ordenador a ver. Para ello, es necesario crear redes neuronales convolucionales, que es un tipo de algoritmo de aprendizaje automático, con las cuales se pueden diferenciar los objetos que se han visto con anterioridad. Sin embargo, este sistema tiene como defecto que el objeto presentado debe tener más o menos la misma forma que el objeto que se ha visto con anterioridad. En caso contrario no lo podrá detectar [35].

Un algoritmo que esté creado para la detección de objetos debe ser capaz de detectar el objeto y localizarlo devolviendo las coordenadas X e Y de dicho objeto y dibujar un rectángulo su alrededor, como podemos observar en la FIGURA 3.9

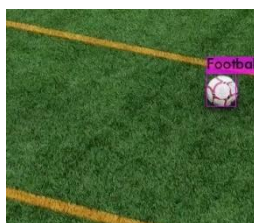


FIGURA 3.9 OBJETO DETECTADO

Los algoritmos de detección de objetos, tal y como se ha comentado antes, buscan objetos del mundo real partiendo de imágenes. Un problema que existe para la detección y que podemos resaltar, puede consistir en el etiquetado del objeto. A la hora de resolver una tarea de detección en una imagen, esta puede tener una o varias regiones de interés con



sus correspondientes etiquetas. Para resolver esta tarea se van a explicar algunos algoritmos de bajo nivel que son usados para desarrollar esta tarea [36].

### **3.6.1.1 Algoritmos de Bajo Nivel**

#### **3.6.1.1.1 Comparación por plantillas**

Es un algoritmo que recibe una plantilla con un objeto, es decir, una imagen pequeña de lo que se desea detectar. Realmente, deseamos detectar el objeto de dicha plantilla en una imagen de mayor tamaño. Esta comparación se hace mediante la proximidad de los detalles de la plantilla y la imagen según las coordenadas de ambas. Como inconveniente encontramos, que si aparece el objeto en la imagen grande, con alguna variación en la rotación o en la escala, este no será detectado [36]. Este algoritmo suele ser usado por ejemplo para detectar las letras en una matrícula de coche.

#### **3.6.1.1.2 Sustracción del fondo**

Este algoritmo se usa con una cámara estática y donde el fondo de la imagen no pueda tener ruido, es decir, zonas de la imagen que no sean importantes o que puedan asemejarse al objeto que vamos a detectar. Suele ser usado para detectar objetos de manera rápida y de forma simple[36]. Generalmente es un algoritmo usado para saber cuántas personas pasan por una puerta en un centro comercial.

#### **3.6.1.1.3 Haar Cascade**

Haar Cascade es un algoritmo de aprendizaje automático, encargado de la detección de objetos. Este algoritmo fue propuesto en el año 2001 por Paul Viola y Micheal Jones a través del artículo *Rapid Detection Using a Boosted Cascade of Simple Features* [37]. Dicho artículo trata sobre la capacidad de la detección de los rostros en imágenes. Sin embargo, este sistema permite, también, la detección de otro tipo de objetos.

Para poder entrenar correctamente este algoritmo hay que insertar, en una función de cascada, numerosos clasificadores débiles llamados *stages*. Cada uno de estos *stages* analiza una parte diferente de la imagen, con una gran cantidad de imágenes positivas y negativas de las que finalmente se obtiene un archivo XML. A través de este archivo se pueden detectar objetos en cualquier otro contenido multimedia, independientemente del tamaño y la localización en la que se encuentre dicho objeto en la imagen [38].

Las imágenes positivas son aquellas que contienen única y exclusivamente regiones del objeto que se quiere detectar. Por el contrario, en las imágenes negativas no puede haber ninguna región de dicho objeto y si la hubiese el detector podría no funcionar correctamente.

La explicación de este algoritmo se puede dividir en cuatro etapas principales:

- Calcular Características Haar
- Crear Imágenes Integrales
- Uso de Adaboost
- Implementar Clasificadores en Cascada

El detector se encarga de ordenar las imágenes de acuerdo con ciertas características; para ello se utilizan dos, tres y cuatro rectángulos del mismo tamaño para crear ventanas en la

imagen. Finalmente, se sumarán los píxeles encontrados en el área blanca y se restarán aquellos que se encuentren en el rectángulo negro como se muestra en la FIGURA 3.10.

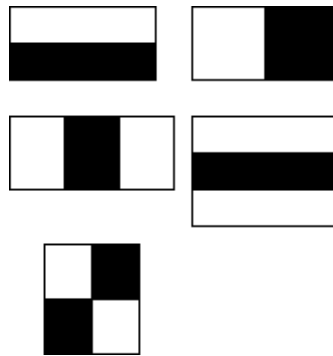


FIGURA 3.10 OBTENCIÓN DE LAS CARACTERÍSTICAS DE UNA IMAGEN

Sin embargo, cuando la imagen es muy grande, el cálculo de estas funciones puede resultar complejo y requiere de la realización de muchas operaciones. Para evitar este cálculo pesado y mejorar la eficiencia del proceso, se crean varias matrices de píxeles a través de las cuales se podrán obtener las características de una forma más sencilla.

A la hora de obtener las características de las imágenes casi todas estas serán irrelevantes dado que solamente interesa la región del objeto que queremos detectar. Para evitar las características que pueden meter ruido a nuestro detector se usa la función de *Adaboost*. Esta función combina los clasificadores más débiles, denominados pasos de decisión, que han dado mejor resultado siendo entrenados por la técnica boosting y de esta forma construyen un buen modelo mediante un clasificador en cascada, como se puede ver en la FIGURA 3.11 [39].

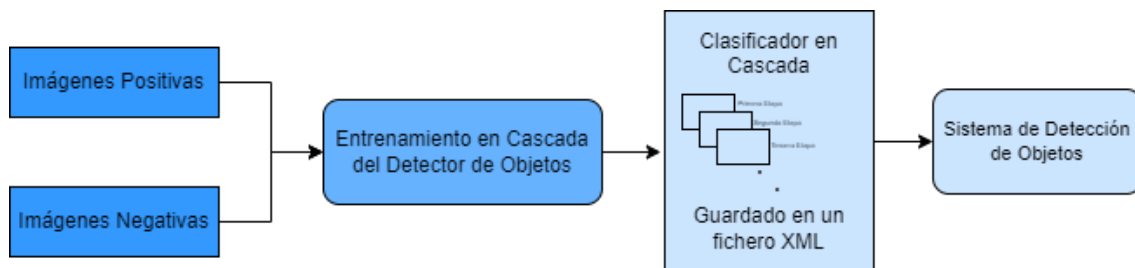


FIGURA 3.11 DIAGRAMA CLASIFICADOR EN CASCADA

Cada clasificador marca la región observada como positiva o negativa; positiva si hay sujeto y negativa si no lo hay. Si se marca como negativo, la región está completa y pasamos a la siguiente región. Si es positivo, se pasa al siguiente estado y así sucesivamente hasta que el último estado lo marque como positivo y en ese caso el detector lo señala como objeto [40].

Algoritmos	Invarianza ante cambios lumínicos	Invarianza ante cambios de escala	Invarianza ante rotación	Coste computacional
Comparación por plantilla	Baja	Baja	Baja	Alto
Substracción del fondo	Baja	Baja	Baja	Baja
Haar Cascade	Alta	Alta	Alta	Alta

TABLA 3.1 COMPARACIÓN ALGORITMOS BAJO NIVEL [36]

Como se puede observar en la TABLA 3.1, a pesar de que Haar Cascade es un algoritmo que necesita un alto coste computacional, como algoritmo de detección de objetos obtiene unos resultados luminosos y cambios en la escala del objeto mejores que el resto de los algoritmos, por lo que esto provoca que Haar Cascade sea uno de los algoritmos elegidos para el desarrollo de este proyecto.

### **3.6.1.2 Algoritmos de alto nivel**

#### **3.6.1.2.1 R-CNN**

Para tratar de evitar el problema de selección de las regiones de interés de una imagen, Ross Girshick propuso un método que consiste en usar este algoritmo de búsqueda selectiva que es capaz de extraer dos mil regiones de una imagen [41].

El algoritmo de búsqueda selectiva genera una segmentación de la imagen y combina las regiones similares más pequeñas en regiones más grandes [42].

Estas dos mil regiones se dimensionan a un tamaño fijo y se introducen en una red neuronal convolucional, formada por cinco capas convolucionales y dos capas conectadas, produciendo un vector de características de 4096 dimensiones. Finalmente, este vector obtenido se introduce en un clasificador SVM el cual se encarga de dividir las regiones por categorías [41].

Este algoritmo necesita mucho tiempo para poder ser entrenado, ya que necesita dividir cada imagen en dos mil regiones. Además, no puede ser utilizado en tiempo real, puesto que necesita de media unos 50 segundos en testear cada fotograma de un vídeo [41].

#### **3.6.1.2.2 Fast R-CNN**

En este algoritmo se solucionan algunos de los problemas del R-CNN. Para acelerar el proceso de aprendizaje, en vez de introducir en la red neuronal convolucional las regiones propuestas, se introduce la propia imagen con el fin de producir un mapa de características convolucionales. Y una vez han creado el mapa, se obtienen regiones de interés a partir del algoritmo de búsqueda selectiva, las cuales son introducidas a las capas conectadas de la red neuronal [41].

Por lo que, la principal razón por la que este algoritmo es más rápido que R-CNN es debido a que no necesita dividir inicialmente cada imagen o fotograma en dos mil regiones diferentes. Pero al ser más rápido que el algoritmo anterior se ve alterado en cierta medida la fiabilidad del algoritmo [41].

#### **3.6.1.2.3 Faster R-CNN**

Es un algoritmo que no usa el algoritmo de búsqueda selectiva y, por el contrario, permite a la red neuronal aprender a partir de las regiones propuestas [41].

Al igual que Fast R-CNN, este algoritmo introduce la imagen directamente en la red neuronal convolucional. Pero, como se ha comentado anteriormente, en vez de usar búsqueda selectiva, este algoritmo tiene una red neuronal aparte usada para predecir las nuevas regiones de interés [41].

El algoritmo solo tarda alrededor de 0.2 segundos por fotograma de vídeo en ejecutarse, por lo que se podría usar para la detección de objetos en tiempo real [41].

#### 3.6.1.2.4 YOLOv3

El algoritmo YOLO (*You Only Look Once*), es un sistema cuyo objetivo principal es la detección de objetos en tiempo real. Para ello, se utilizan redes convolucionales junto con aprendizaje profundo, de forma que la imagen quede dividida en distintas regiones de  $S \times S$ . Posteriormente, Yolo crea celdas de identificación que se vinculan con las probabilidades a cada clase predichas en cada una de las regiones mencionadas. Y finalmente se procede a eliminar las cajas que se encuentran por debajo de un porcentaje marcado, FIGURA 3.12. Asimismo, el algoritmo aprende píxel por píxel permitiendo la generalización para cualquier entrada que no pertenece al conjunto de entrenamiento [43].

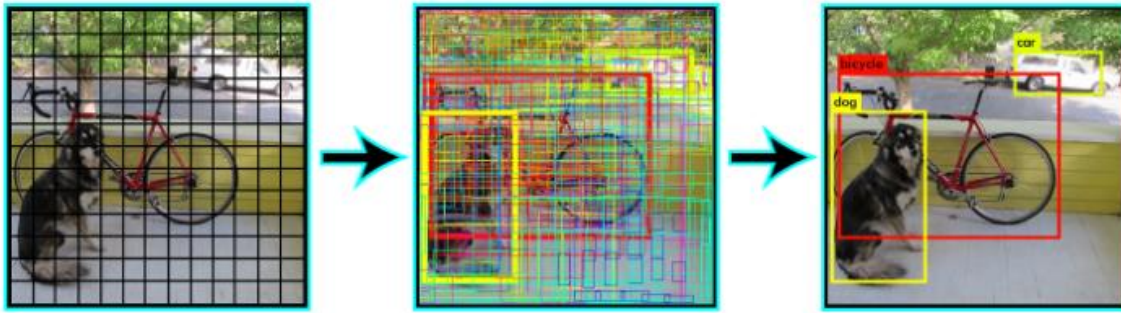


FIGURA 3.12 PASOS ALGORITMO YOLOv3 [44]

La mayoría de los proyectos de Machine Learning que consisten en la detección de objetos en imágenes, no solamente se dedican a identificar de qué tipo de objeto se trata, sino que también hay que localizarlo dentro de la imagen (obtener las coordenadas de la “caja” que lo contiene). Para ello, Yolo crea tantas “cajas” como número de objetos haya en la imagen, identificando así los objetos con su correspondiente clase. Por lo tanto, Yolo no solo realiza la detección del objeto, sino que también localiza el espacio donde está [44].

Este algoritmo normalmente es entrenado usando Darknet que es una red convolucional la cual consta de 24 capas.

YOLO es un algoritmo más eficaz que los algoritmos comentados con anterioridad. Es capaz de procesar cerca de sesenta fotogramas en un segundo, por lo que comparando con los algoritmos de alto nivel explicados es el mejor algoritmo para la detección de objetos en tiempo real. Aunque como desventaja suele ser lento a la hora de realizar la detección [42].

Normalmente las medidas del *bounding box* que se recibe cuando YOLO detecta un objeto sin medidas sin normalizarlas. Para que se pueda crear un modelo de YOLO estas medidas deben estar normalizadas en el formato de YOLOv3 usando las siguientes ecuaciones (2),(3),(4) y (5).

$$x_{YOLO} = \left( \frac{x_{min} + \frac{(x_{max} - x_{min})}{2}}{2} \right) / width \quad (2)$$

$$y_{YOLO} = \left( \frac{y_{min} + \frac{(y_{max} - y_{min})}{2}}{2} \right) / height \quad (3)$$

$$w_{YOLO} = \frac{(x_{max} - x_{min})}{width} \quad (4)$$

$$h_{YOLO} = \frac{(y_{max} - y_{min})}{height} \quad (5)$$

### 3.6.2 Seguimiento

Un algoritmo de tracking se usa para localizar y seguir un objeto en sucesivos frames. Para poder realizar esto existen siete técnicas diferentes las cuales van a ser analizadas a continuación. La ventaja que tienen los algoritmos de tracking con respecto a los algoritmos de detección es que son más rápidos, no necesita analizar el total de la imagen [45].

#### 3.6.2.1 Boosting

Tracker basado en el algoritmo Adaboost, usado en el detector de objetos Haar Cascade. Este tracker, por lo tanto, como se ha explicado anteriormente, con Haar Cascade, necesita imágenes negativas y positivas. Para obtener estas imágenes, la región de interés es seleccionada como imagen positiva y, el fondo de la imagen lo divide en regiones del mismo tamaño que la ROI y lo convierte en imágenes negativas. Es un algoritmo que ya está algo anticuado, debido a que existen algoritmos mejores. *Boosting* funciona de forma correcta a excepción de si pierde el objeto, el cual no es capaz de saber si lo ha perdido o no[45].

Este algoritmo, cuando el detector le devuelve los valores en los que se encuentra el objeto, otorga a cada píxel una puntuación según donde se encuentre el píxel y el objeto. Cuando se cambia de fotograma, el objeto se encuentra donde se encuentre la mayor puntuación obtenida tras analizar el fotograma [45]

Como se puede observar en la FIGURA 3.13 el cuadrado en verde pertenece a la región de interés que debe ser rastreada y el resto que se encuentra en blanco y negro, son las imágenes negativas que usa este algoritmo.



FIGURA 3.13 EJEMPLO BOOSTING

#### 3.6.2.2 MIL

MIL (*Multiple Instance Learning*) es un algoritmo similar a Boosting. La diferencia es que este tracker no necesita el uso de imágenes negativas, solamente imágenes positivas

que son obtenidas a través de la región de interés obtenida por el detector de objetos. Y además coge otras regiones cercanas al objeto para convertirlas también en imágenes positivas. Como se puede observar en la FIGURA 3.14, donde el cuadrado verde es la imagen que proviene del detector y los cuadrados grises aquellos que realiza este rastreador para poder dar una localización más precisa [45].



FIGURA 3.14 EJEMPLO MIL

En este algoritmo, al igual que en Boosting, los errores de seguimiento que se producen no se reportan de forma correcta, que puede ser un efecto secundario de su uso. Pero tiene un rendimiento superior a Boosting y no tiene tantos errores.

### 3.6.2.3 KCF

KCF (*Kernelized Correlation Filters*), es un rastreador mezcla de los algoritmos anteriores Boosting y MIL. Este algoritmo usa la misma técnica que MIL de obtener varias regiones de interés sobreponiendo unas sobre otras y dando una puntuación a cada región de interés dependiendo de la región principal detectada. Y de esta forma en el siguiente fotograma se otorga a este algoritmo la puntuación máxima y la *bounding box* actualizada [45].

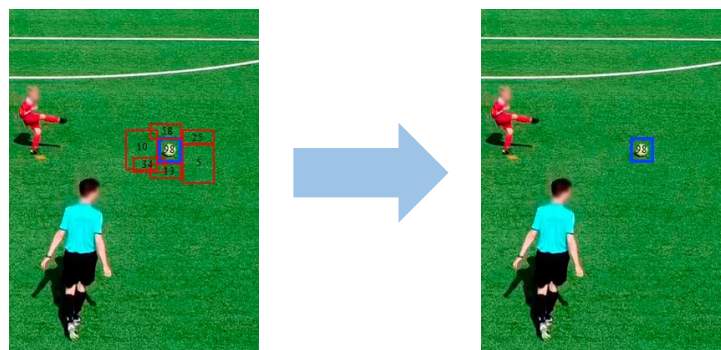


FIGURA 3.15 EJEMPLO KCF

Este algoritmo no es capaz de continuar funcionando si el objeto se ha perdido o ha dejado de ser detectado, pero, por el contrario, es más preciso y rápido que los dos algoritmos anteriores. Es uno de los algoritmos apropiados para este proyecto y en caso de pérdida del objeto se volvería a pasar por el detector de objetos que se esté usando.

### 3.6.2.4 TLD

TLD (*Tracking, Learning, and Detection*) que como su propio nombre indica, divide el proceso el seguimiento de un objeto en tres procesos. El algoritmo trata de aprender de los errores que obtiene e intenta mejorar el error para los siguientes frames. Para realizar

este proceso va guardando las diferentes regiones de interés que obtiene y desarrollando nuevos modelos de seguimiento[45]. Como se puede observar en la figura FIGURA 3.16 encontramos un porcentaje de semejanza en el centro del cuadrado verde al modelo que va creando TLD con las múltiples fotos a la derecha de la imagen que sirven a este algoritmo para que el algoritmo vaya aprendiendo y mejorando así el modelo.



FIGURA 3.16 EJEMPLO TLD

Este algoritmo, debido al aprendizaje, es capaz de volver a encontrar un objeto que haya sido ocultado o haya desaparecido de la zona que la cámara enfoca. Lo que hace que el proceso de rastrear un objeto sea más rápido que los algoritmos comentados con anterioridad, puesto que no hay que volver a pasar por la etapa de detección del algoritmo.

### 3.6.2.5 MedianFlow

Median Flow es un algoritmo que va rastreando el objeto fijándose en qué trayectorias ha estado anteriormente intentando predecir a qué zona irá en el siguiente fotograma. Este algoritmo da resultados precisos cuando la trayectoria de los objetos suele ser fácil de predecir, por lo que la velocidad de estos objetos no puede ser grande. A diferencia de otros algoritmos, Median Flow reporta de forma correcta cuando ha dejado de seguir al objeto [45].

Este algoritmo, para poder localizar dónde está el objeto en el siguiente fotograma, usa el principio del algoritmo de Lucas-Kanade, que intenta de localizar puntos en la imagen en el siguiente fotograma. Este proceso consiste en dar una lista de puntos en los ejes X e Y, y obtener la mediana de estos puntos y será en principio la localización futura del objeto [46]. En la FIGURA 3.17 podemos observar cómo funciona este algoritmo con los puntos en el interior de la región de interés.



FIGURA 3.17 EJEMPLO MEDIANFLOW



### 3.6.2.6 CSRT

CSRT (*Channel and Spatial Reliability Tracking*) es un algoritmo que tiene cierta facilidad de encontrar las características, localización y clases principales de un objeto. Experimentalmente, es un rastreador que tiene resultados adecuados a la hora de trabajar con algoritmos de detección de objetos [45].

Este es un algoritmo más preciso que KCF, pero un poco más lento en términos de procesamiento, por lo que tolera pocos frames por segundo, lo que no es muy útil para este proyecto, dado que requiere que el seguimiento se haga en tiempo real.

### 3.6.2.7 MOSSE

MOSSE (*Minimum Output Sum of Squared Error*) es un algoritmo útil para posibles cambios de luz, deformaciones y posición de un objeto. Este algoritmo es capaz de detectar cuando el objeto no está en la imagen basándose en una métrica que mide la nitidez máxima del plano de correlación conocida como *peak-to-sidelobe ratio* [45].

Al igual que otros algoritmos comentados con anterioridad, es preciso y eficaz a la hora de rastrear un objeto, sin embargo, cuando hay que rastrear varios objetos en el mismo vídeo este algoritmo pierde accuracy. Además, como ventaja, tiene que es muy sencillo de crear. MOSSE es capaz de funcionar correctamente a unos frames por segundo superiores a 450 sin ningún retardo [45], [47].

## 3.7 Controlador de estados

### 3.7.1 Controlador PID

Un controlador PID es un sistema que controla un circuito de lazo cerrado de forma que se pueda obtener el estado de salida deseada. Este controlador está formado por tres acciones que son las siguientes: una proporcional, una integral y otra derivativa. El conjunto de estas tres acciones da origen a las siglas PID[48].

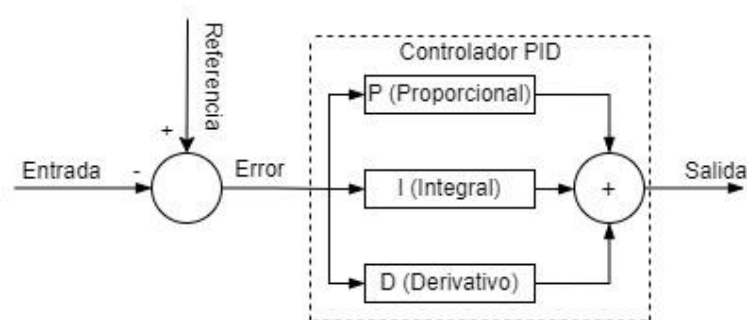


FIGURA 3.18 CONTROLADOR PID

Como se observa en la FIGURA 3.18 el controlador recibe unos valores de error, los cuales provienen de la resta que se realiza entre los valores de referencia, que es el valor deseado, y los valores de entrada, que son los valores reales del sistema en un momento determinado. Esta operación genera una señal de error que indicará la diferencia entre el sistema real y el sistema deseado. Por lo tanto, si la señal de error es grande, significa que el sistema actual está demasiado lejos del deseado, y si, por el contrario, es pequeña, significa que el sistema actual es casi similar al requerido [49].



El controlador PID está formado por el sumatorio de tres ganancias totalmente independientes unas de otras. Matemáticamente, este tipo sistema de control PID tiene la siguiente ecuación:

$$K_p * Error(t) + K_i * \int_0^t Error(t) dt + K_d * \frac{Error(t)}{d(t)} \quad (6)$$

Para ajustar correctamente las ganancias y de esta forma poder lograr un comportamiento óptimo del sistema hay que efectuar una sintonización de los parámetros del PID, estos parámetros son los siguientes:

- $K_p$ : Corresponde a la parte proporcional del sistema, es la que encargada de disminuir el error y aumentar la velocidad de respuesta. Con un valor alto se pueden producir oscilaciones y alguna sobrecorrección grande [50].
- $K_i$ : Corresponde a la parte integral del sistema, se encarga de eliminar el error que aparece cuando se encontraba en un estado estacionario, es decir, por perturbaciones externas como puede ser el viento. Esta constante integra la desviación producida en el tiempo sumando la parte proporcional. Un valor alto sirve para disminuir el error en momentos de ráfagas de viento [50].
- $K_d$ : Corresponde a la parte derivativa del sistema, si el sistema se volviese muy inestable se tiende a aumentar esta constante con el fin de estabilizarlo. Constante que se encarga de minimizar el error producido lo antes posible, a pesar de que se produzcan oscilaciones. El objetivo de esta constante es que el vuelo se haga de manera suave y no tenga apenas oscilaciones y un valor alto de esta variable puede producir vibraciones a la hora del vuelo, debido a que puede a ver valores muy dispares, por ejemplo una orden mande un -10 y otra un 10 [50].

Como se ha analizado anteriormente las ganancias son totalmente independientes, por lo que existen controladores PID, PI, PD, P o I. En este proyecto solamente se requiere el uso de la parte proporcional y derivativa, ya que se va a trabajar sobre un entorno ideal, debido a esto el sistema solo va a hacer uso de un controlador PD, dado que es un proceso estable en el tiempo. El controlador PD se usa para variar la velocidad del dron en el eje X y poder de esta forma centrar el objeto en el medio de la pantalla.

Para ajustar un controlador PID existen diferentes formas entre ellas podemos encontrar algunas formas teóricas y otras manuales siguiendo alguna heurística establecida. Entre ellas encontramos las siguientes maneras:

- Caracterización plana: se obtiene de forma experimental la respuesta de la planta ante una entrada escalón unitario. La planta para ello tiene que ser estable y no tiene parte integral, por lo que salida tendrá forma de S. Esto provoca que solo sea aplicable a salidas cuya grafica tenga forma de S [51] y [52].

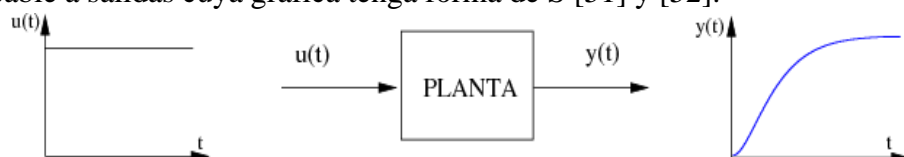


FIGURA 3.19 DIAGRAMA CARACTERIZACION PLANA [52]

- Aplicación reglas de sintonización: consiste en introducir una entrada determinada al sistema y analizar la salida producida. Al analizar esta salida se aplicará cualquier regla de sintonización entre la que encontramos la de Ziegler-Nichols [51].
- Auto tuning: Actualmente hay muchos algoritmos que son capaces de actualizar en el tiempo los valores del PID. Esto provoca que este tipo de controlador tenga bastante éxito [51].
- Ajuste manual: A veces, solamente se puede recurrir al ajuste realizado de forma manual. Se suelen seguir alguna heurística por parte del usuario que lo está realizando. Y se pueden llegar a conseguir mejores resultados que el resto de los ajustes [51].

## 4 PROPUESTA

En este trabajo se aborda el problema de seguimiento de una pelota de fútbol desde una estación de tierra, un portátil en este caso, que se comunica con el dron de bajo coste, Tello [53], de la compañía DJI [8]. Para ello se ha diseñado un sistema capaz de detectar y posteriormente seguir la pelota en un campo de fútbol controlando de forma autónoma un dron.

El dron usado, se creó con fines educativos, para todas las edades y se puede programar a través de Scratch, una SDK de la propia compañía u otros lenguajes de programación como Python.

En cuanto a las características, este dron tiene un peso aproximado de 80 g. Integra un sensor telemétrico, un barómetro, una señal de Wi-Fi de 2.4 GHz y una cámara de 1280 x 720 píxeles con 30 frames/segundo.

Con respecto al rendimiento de vuelo, logra como velocidad máxima los 10 metros/segundo y es capaz de alcanzar una altitud máxima de 30 m con una distancia máxima de 100 m. Como inconveniente tiene que tan solo posee una autonomía de 13 minutos [53].

Para poder realizar este proceso, se usa la cámara que está equipada en el dron, librerías del lenguaje de programación Python, redes neuronales y un protocolo de comunicación entre el dron y la estación de tierra.

Para explicar mejor la propuesta, la vamos a dividir en tres. Como podemos observar en la FIGURA 4.1 se muestran tres secciones principales: la inicialización del sistema, el módulo de visión y el de control.

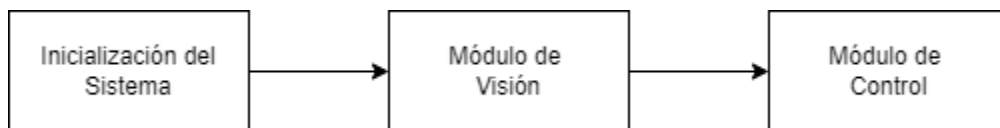


FIGURA 4.1 SECCIONES PRINCIPALES

### 4.1 Inicialización del sistema

Parte del proyecto que consiste en la conexión entre la estación de tierra y el dron. Como podemos observar en la FIGURA 4.2, el portátil, la estación de tierra, y el dron se conectan a partir del protocolo de comunicación UDP (3.4.5), usando el sistema Wi-Fi del ordenador y buscando la red TELLO-634D08, red que activa el dron una vez se ha encendido.

Una vez conectado el dron devuelve al portátil el vídeo que graba con la cámara en tiempo real, así como la telemetría. Y una vez se ha analizado en el portátil lo recibido por parte del dron, este envía los comandos necesarios para controlar de la forma requerida el dron y encuadrar la pelota de fútbol en la imagen.

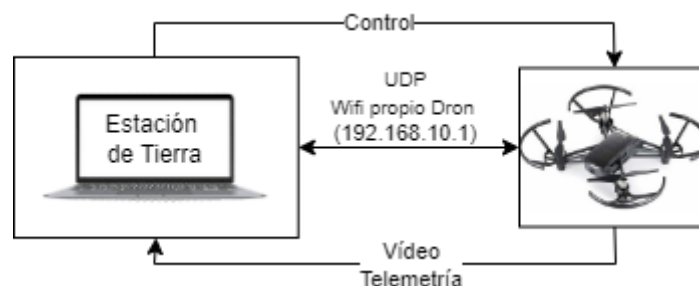


FIGURA 4.2 PROPUESTA GENERAL

Antes de ver el proceso de la detección y control, se analiza el código que se ha creado para la ejecución de esta propuesta. El entorno de desarrollo integrado que se ha usado para la ejecución del programa es PyCharm Professional[54].

Para ejecutar el programa, en primer lugar, hay que comentar el uso del lenguaje de programación Python, que junto a la librería *OpenSource* llamada *djitellopy*, que usa la función *connect()* se comprueba si existe conexión entre la estación de tierra y el dron. En caso de que esta conexión exista, se va a poder ejecutar el resto del programa de reconocimiento y vuelo por parte del dron.

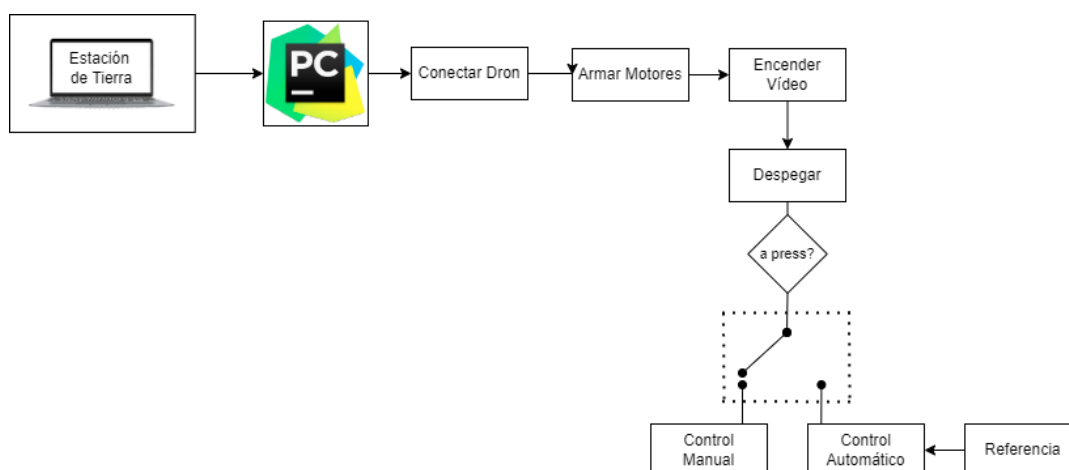


FIGURA 4.3 INICIALIZACIÓN INICIAL

Cuando se ejecuta el programa lo primero que se realiza es la activación de los motores y se inicia la cámara, ya que en este dron podemos activar y desactivar el funcionamiento de la cámara según lo que queramos realizar.

Una vez, llegado a este punto, la siguiente tarea es despegar el dron, proceso que se realiza con la función *takeoff()* que se levanta a 1.2 metros y posteriormente de forma manual. Elevamos el dron usando la función *move\_up(10)* tantas veces como la flecha ↑ del teclado se pulse hasta llegar a la posición donde se tenga una buena visión del campo de fútbol, como la de la FIGURA 4.4

Con la posición del dron alcanzada, cerca de los 3,5 metros para que la cámara del dron muestre de forma adecuada la posición del juego en la que se encuentra el balón y evitar que solamente se muestre en el vídeo el balón, se procede a pulsar la tecla “a” del teclado, para pasar del control manual al control automático haciendo que la estación de tierra comience a enviar datos de control constantemente, dependiendo de la detección de la pelota y el encuadre que se tenga de esta.



FIGURA 4.4 VISTA DESDE EL DRON

Una vez, el dron ha despegado y está estabilizado en el aire, comienza a funcionar la segunda sección del proyecto, que consiste en compartir los fotogramas que va detectando la cámara con la estación de tierra.

La estación de tierra es la encargada de procesar la imagen recibida y proporcionar a la tercera sección, el módulo de control, la posición estimada de la pelota de fútbol en el sistema de referencia de la cámara.

## 4.2 Módulo de visión

El módulo de visión se divide en dos partes: detección y seguimiento. El diagrama de funcionamiento de este módulo es el siguiente:

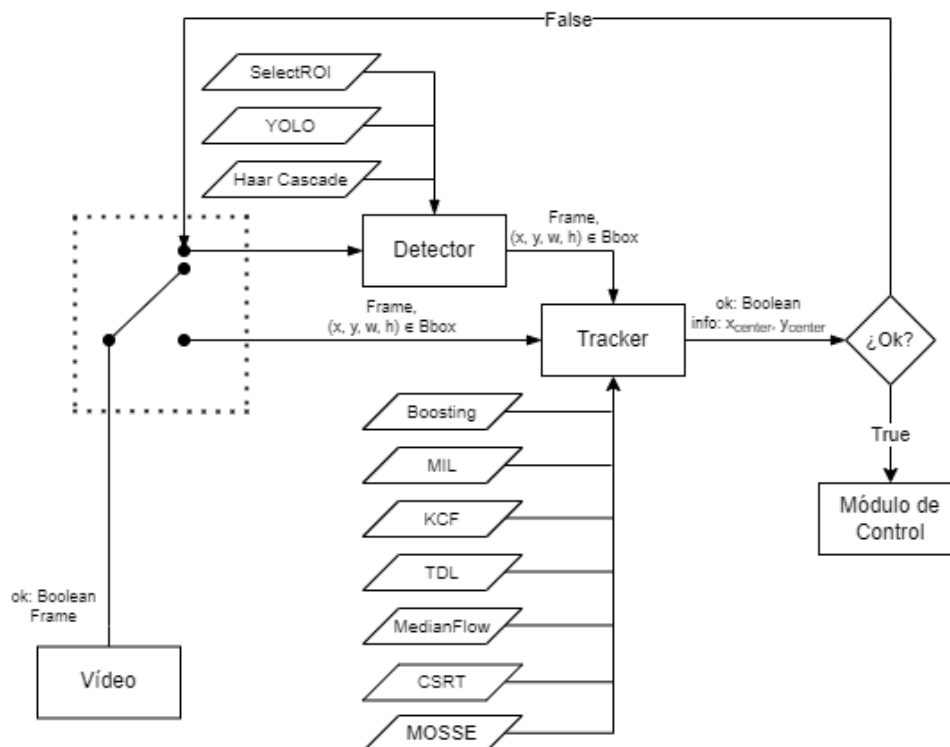


FIGURA 4.5 MÓDULO VISIÓN

El objetivo del sistema de detección dentro del módulo de visión es detectar dónde se encuentra la pelota de fútbol. Para ello, lo primero que se va a comentar es que, para poder

realizar este proceso, se han creado dos modelos de detección de objetos en imágenes de forma automática. Estos modelos son YOLOv3 y Haar Cascade, aunque también ha sido utilizado en el proceso un algoritmo para seleccionar la región de interés de forma manual (*Select ROI*).

En primer lugar nos vamos a centrar en YOLOv3. YOLOv3 es un algoritmo que permite generar redes convolucionales por lo que los conjuntos de datos de entrenamiento y test están balanceados para prevenir que la red sufra *overfitting*. El principal problema que tiene este algoritmo es que necesita una gran cantidad de imágenes etiquetadas correctamente para poder ser entrenado de forma correcta.

Para poder obtener un modelo asequible se han creado varios modelos YOLOv3, el primero se creó usando la base de datos de Google de imágenes [55], que usando un script proporcionado por *THE AI GUY* [56], se han descargado las imágenes de la categoría *Football*, que tiene cerca de 5000 imágenes etiquetadas, es decir, cada imagen tiene un documento de texto donde están almacenadas las medidas del *bounding box*.

Una vez descargadas las imágenes se ha procedido a crear un modelo YOLOv3 con estas imágenes para posterior usar este modelo para detectar donde se encuentra la pelota de fútbol y guardar las imágenes con su correspondiente fichero de texto con las medidas del *bounding box* ( $x_{min}, y_{min}, x_{max}, y_{max}$ ).

#### 4.2.1 Entrenamiento modelo YOLO

Todo el proceso comentado anteriormente se ha realizado para crear un modelo final más adelante, con imágenes de vídeos que se han grabado de partidos de fútbol, vídeos proporcionados por la empresa Fly-Fut[3].

Para crear modelos YOLOv3 se han seguido los pasos explicados por PJReddie [57], en primer lugar, hay que instalar Darknet, programa de fuente abierta escrito en C y en CUDA que proporciona una red neuronal convolucional, en el ordenador. Programa que sirve para crear el modelo de YOLOv3, de forma más sencilla y eficaz.

El primer paso para crear un modelo YOLOv3 es estructurar el fichero donde se va a crear este modelo. En este directorio hay una carpeta llamada *data* donde hay un fichero llamado *obj.data* que se tiene que editar según el número de tipos de clases que se quiera detectar en el modelo. En el caso de este proyecto, solo se va a detectar la pelota de fútbol *classes=1*, además posee otros tres puntos que son *train* donde se encuentran las imágenes de entrenamiento, *names* ubicación del fichero que contiene los nombres de los objetos a detectar y *backup* donde se va a guardar una copia de seguridad del modelo que se está realizando.

Otro fichero es *obj.names*, el cual hay que editar su contenido y nombrar las categorías que se van a detectar, por ejemplo poner el nombre Pelota, para que cuando detecte la pelota la etiquete como tal. También, hay un directorio llamado *obj* donde se ubican las imágenes con sus ficheros de texto correspondientes.

Posteriormente hay que ejecutar el fichero *generate\_train.py* encargado de escribir en *data/train.txt* el nombre de donde se encuentran las imágenes de entrenamiento de la red.

Y finalmente, antes de ejecutar el modelo para conseguir los pesos del modelo, se analizan los datos para crear el modelo YOLO; este fichero es *yolov3-custom.cfg* en la carpeta *cfg*.

Para esta creación se han usado veinticinco capas convolucionales seguidas de dos capas completamente conectadas y al final de cada una se realiza un *max-pooling*. Además, a este primer modelo se le ha aplicado una tasa de aprendizaje de 0.001. Y tiene que realizar un total de cuatro mil etapas para lograr la creación del modelo.

Una vez analizado este paso se ejecuta en el ordenador el siguiente comando:  
*darknet.exe detector train data/obj.data cfg/yolov3-custom.cfg darknet53.conv74*

Este modelo devolvió un error de entrenamiento de 0.3080 por lo que, aunque no sea un error bueno puede ser un buen modelo para continuar con el proyecto, ya que el proceso de etiquetado se realiza de forma automática extrayendo fotograma a fotograma.

El siguiente paso fue analizar ocho vídeos de partidos de fútbol de duración entre diez y quince minutos, haciendo que el modelo obtenido con anterioridad devuelva cada fotograma del vídeo con su respectivo documento de texto donde se encuentran los datos normalizados de la región de interés. Realizando esto se obtuvieron cerca de medio millón de fotogramas, los cuales debieron ser analizados de forma manual, para ver si se había detectado bien o no la pelota en la imagen.

Cuando se analizaron todas las imágenes y se eliminaron aquellos fotogramas en los que, o no había pelota de fútbol en ella, o estaba mal analizado, estas se almacenaron en la carpeta *data* para posteriormente realizar un buen modelo.

De los vídeos usados para la extracción de fotogramas para posteriormente entrenar nuevos modelos de YOLO, se extrajeron cerca de medio millón de fotogramas pero como el modelo inicial de YOLO, no era del todo bueno, apenas se obtuvieron alrededor de 40.000 fotogramas correctamente etiquetadas, con su correspondiente *bounding box*. A pesar de esto, es una buena base de datos de imágenes para la creación del modelo. Estas imágenes fueron usadas para realizar tres modelos YOLO con diferente tasa de aprendizaje, para posteriormente analizar a través del error de entrenamiento cual tenía un mejor resultado.

MODELOS	TASA DE APRENDIZAJE	FILTROS	CICLOS	ERROR DE ENTRENAMIENTO
Modelo1	0.4	27	4000	NaN
Modelo2	0.01	27	4000	0.2426
Modelo3	0.001	27	4000	0.0661

TABLA 4.1 MODELOS YOLOv3

La creación de los modelos tardaba alrededor de siete horas en completarse una vez creados los tres modelos diferentes con la tasa de aprendizaje diferente se llegó a la conclusión que cuanto mayor era la tasa de aprendizaje mayor era el error de entrenamiento por lo que como se puede apreciar en la TABLA 4.1 el mejor modelo es el Modelo3 que es el que se ha usado para el proyecto.

En segundo lugar vamos a analizar Haar Cascade.

Haar Cascade es un algoritmo implementado en aprendizaje automático siguiendo una función en cascada entrenada a partir de un conjunto de imágenes positivas y negativas.

Y para la creación de este modelo se han seguido las instrucciones aportadas por Dasaradh Makes [58].

#### 4.2.2 Entrenamiento modelo Haar Cascade

Se han usado alrededor de 5.000 fotos que, como diferencia con el modelo de YOLO, no necesitan gran cantidad de imágenes y, además, los ficheros de texto son diferentes a los que YOLO necesita, dado que Haar Cascade no requiere de ningún tratamiento de los datos, es decir, no necesita normalización.

El proceso para crear un buen modelo Haar Cascade consiste en lo siguiente: colocar las imágenes en formato .bmp en la carpeta *positive/rawdata*, y en la carpeta *positive* se encuentra un fichero no nombre *info.txt*, el cual tiene que contener la ubicación de las imágenes y los valores correspondientes al *bounding box*. Para realizar este fichero se ha usado un script que era capaz de leer todos los archivos de texto pertenecientes a los diferentes fotogramas e insertar la información en el fichero *info.txt*.

El siguiente paso es insertar un número de imágenes negativas superior al de imágenes positivas. Una imagen negativa es aquella en la que no aparece una pelota de fútbol y las fotografías están en blanco y negro. En este caso se han introducido en la carpeta *negative* alrededor de 60.000 fotografías.

Finalmente, hay que modificar el archivo *haarTraining.bat* y editar el número de fotos positivas que hay en el fichero (*-npos 5148*), el número de fotos negativas (*-neg 64010*) y finalmente el número de etapas necesarias para realizar el modelo (*-nstages 14*) son los datos que han sido usados para la creación del modelo final. El cual se crea ejecutando el archivo *haarTraining.bat* y una vez se ha ejecutado se tiene que ejecutar *convert.bat* para convertir los ficheros creados por el primer archivo en un archivo de formato .xml.

Una vez creados los modelos y siguiendo el orden de FIGURA 4.5, el primer paso, una vez recibido un fotograma, es detectarlo. Una vez que se ha detectado se devuelven los valores de la *bounding box*, región de interés, de la imagen, y son iniciados en el tracker a través de la función *tracker.init(img, bbox)*.

El tracker se inicializa usando la función *tracker = selectTracker(track)*, y aquí se elige cual de todos los diferentes tracker se quiere usar.

Para el desarrollo de este proyecto se va a hacer uso de siete tracker diferentes que serán analizados, Boosting, MIL, KCF, TLD, MEDIANFLOW, CSRT y MOSSE.

Según se haya detectado bien o mal por el tracker, este devuelve un valor booleano True o False. Si devolviese False, el algoritmo está hecho para que se vuelva a detectar la pelota usando o YOLO o Haar Cascade o el método manual. Si por el contrario, se devolviese True, se pasaría a la última sección del proyecto, el módulo de control.



### 4.3 Módulo de Control

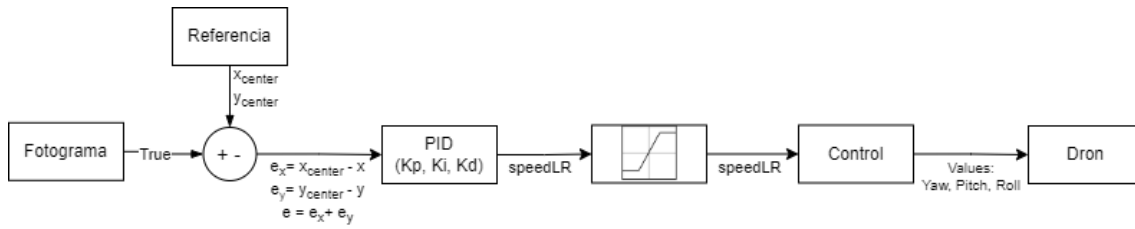


FIGURA 4.6 MÓDULO DE CONTROL

Este módulo recibe la información de la región de interés (x, y, w, h) y donde se encuentra el centro de la imagen, debido a que, ahí, es donde se va a intentar que esté situada en todo momento la pelota. Como se puede observar en la FIGURA 4.7 el error que se produce es bastante grande ya que la pelota está detectada en el recuadro azul y queremos que la pelota esté situada en todo momento en el recuadro rojo.



FIGURA 4.7 EJEMPLO ERROR

Para que esté situada en la posición requerida se calcula el error en el eje X, realizando la resta entre el valor resultante al dividir el ancho de la imagen entre dos y el valor x de la posición del círculo amarillo en la imagen, que se corresponde el primer valor de la *bounding box* de la pelota. Con esta operación se obtiene el error en el eje X. Además también se calcula el error en el eje Y restando el valor resultante al dividir la altura de la imagen entre dos y el valor y de la posición del círculo amarillo de la imagen, segundo valor de la *bounding box*.

Estos errores se han calculado para poder calcular la velocidad usando valores fijos y valores obtenidos usando la fórmula de control del PID; valores que se les van a aplicar al dron usando el *roll* y el *yaw* únicamente. Para eso se han usado las siguientes ecuaciones:

$$Roll = -5 \text{ si } errorX < -50 \text{ o } Roll = 5 \text{ si } errorX > 50 \quad (7)$$

$$\begin{aligned} error &= errorX + errorY \\ Yaw &= K_p \cdot error + K_d \cdot (error - pError[error]) \end{aligned} \quad (8)$$

La ecuación (8) pretende minimizar el error a lo largo del tiempo y para ello usa una constante  $K_p$  multiplicada por el error actual y lo suma a otra constante  $K_d$  multiplicada por la resta del error actual menos el error anterior.

Una vez obtenidos los datos en el *yaw*, estos tienen que pasar por un módulo de saturación debido a que los valores devueltos son muy altos, provocando que los valores se encuentren fuera del rango de la función usada y además sirve también para que el movimiento del dron sea más controlable. Estos valores se han ajustado a  $[-100, 100]$ , y en el caso del *roll* se han ajustado a  $[-5, 5]$ .

Finalmente, estos valores son enviados al dron para que este se mueva e intente situar la pelota correctamente en el centro de la imagen.

Cuando se ha realizado una primera iteración de este proceso, el dron envía al detector o al tracker otro fotograma y vuelve a empezar el proceso de seguimiento más control del dron.



## 5 VALIDACIÓN

El objetivo de este apartado es realizar un análisis que permita validar como es el sistema creado, a través de diferentes resultados obtenidos realizando pruebas de forma aislada y de esta manera analizar el comportamiento del dron, los puntos fuertes del sistema, las debilidades y poder obtener las conclusiones del proyecto.

La ejecución de las pruebas necesarias para realizar el análisis del proyecto sigue la siguiente secuencia de pasos:

- 1) Establecer la conexión entre la estación de tierra y el dron.
- 2) Una vez conectado, se encienden los motores y la cámara del dron, activando de esta forma el vídeo.
- 3) Se despega y una vez se ha estabilizado en el aire se inicia la detección del objeto.
- 4) Para la detección de la pelota se ha realizado el mismo experimento para los tres algoritmos.
- 5) Posteriormente se ha realizado un experimento para ver cuál de los siete algoritmos de seguimiento es el mejor para el caso requerido.
- 6) Finalmente, se realizará un análisis con los que observaremos cuales son los mejores valores para el control del dron y localizar la pelota en la posición propuesta.

### 5.1 Movimiento del Dron

El primer experimento consiste en controlar el dron desde la estación de control. Para ello se han desarrollado varios scripts con las diferentes funciones que aporta la librería *djitellopy*, con el fin de analizar qué función se comporta mejor para usarla en el proyecto.

#### 5.1.1 `move_right(d)`

En este experimento se pretende evaluar el comportamiento del dron al realizar un movimiento en el sentido positivo del eje X, usando un control remoto sería el mismo movimiento que realizar *yaw* hacia la derecha. Esta función tiene como posibles valores de *d* los comprendidos entre 20 y 500 cm.

#### 5.1.2 `move_left(d)`

En este experimento se pretende evaluar el comportamiento del dron al realizar un movimiento en el sentido negativo del eje X, usando un control remoto sería el mismo movimiento que realizar *yaw* hacia la izquierda. Esta función tiene como posibles valores de *d* los comprendidos entre 20 y 500 cm.

#### 5.1.3 `move_up(d)`

En este experimento se pretende evaluar el comportamiento del dron al realizar un movimiento en el sentido positivo del eje Y, usando un control remoto sería el mismo movimiento que realizar *pitch* hacia arriba. Esta función tiene como posibles valores de *d* los comprendidos entre 20 y 500 cm.

#### 5.1.4 move\_down(d)

En este experimento se pretende evaluar el comportamiento del dron al realizar un movimiento en el sentido negativo del eje Y , usando un control remoto sería el mismo movimiento que realizar *pitch* hacia abajo. Esta función tiene como posibles valores de  $d$  los comprendidos entre 20 y 500 cm.

#### 5.1.5 move\_forward(d)

En este experimento se pretende evaluar el comportamiento del dron al realizar un movimiento en el sentido positivo del eje Z, usando un control remoto sería el mismo movimiento que realizar *throttle* hacia arriba. Esta función tiene como posibles valores de  $d$  los comprendidos entre 20 y 500 cm.

#### 5.1.6 move\_back(d)

En este experimento se pretende evaluar el comportamiento del dron al realizar un movimiento en el sentido negativo del eje Z, usando un control remoto sería el mismo movimiento que realizar *throttle* hacia abajo. Esta función tiene como posibles valores de  $d$  los comprendidos entre 20 y 500 cm.

#### 5.1.7 rotate\_clockwise( $\alpha$ )

En este experimento se pretende evaluar el comportamiento del dron al realizar un movimiento realizando un giro sobre su eje en el sentido de las agujas del reloj, usando un control remoto sería el mismo movimiento que realizar *roll* hacia la derecha. Esta función tiene como posibles valores de  $\alpha$  los comprendidos entre 0 y 360 grados.

#### 5.1.8 rotate\_counter\_clockwise( $\alpha$ )

En este experimento se pretende evaluar el comportamiento del dron al realizar un movimiento realizando un giro sobre su eje en el sentido contrario de las agujas del reloj, usando un control remoto sería el mismo movimiento que realizar *roll* hacia la izquierda. Esta función tiene como posibles valores de  $\alpha$  los comprendidos entre 0 y 360 grados.

#### 5.1.9 send\_rc\_control(v1,v2,v3,v4)

En este experimento se pretende evaluar el comportamiento de dicha función, la cual permite realizar todos los movimientos mencionados en los puntos 5.1.1 al 5.1.8 en una sola función.

Esto se debe a que los valores de  $v1$  corresponden al roll,  $v2$  corresponden al pitch,  $v3$  corresponden al throttle y  $v4$  corresponden al yaw.

- Si  $v1$  es positivo el dron realiza la función `rotate_clockwise( $\alpha$ )` por el contrario si fuese negativo realiza la función `rotate_counter_clockwise( $\alpha$ )`.
- Si  $v2$  es positivo el dron realiza la función `move_up(d)` por el contrario si fuese negativo realiza la función `move_down(d)`
- Si  $v3$  es positivo el dron realiza la función `move_forward(d)` por el contrario si fuese negativo realiza la función `move_back(d)`
- Si  $v4$  es positivo el dron realiza la función `move_right(d)` por el contrario si fuese negativo realiza la función `move_left(d)`

Los valores de v1, v2, v3 y v4 representan la velocidad que el dron usa para desplazarse en los diferentes ejes. Estos valores oscilan entre el -100 y el 100.

Una vez se han realizado los experimentos se obtuvo como resultado que la mejor opción era el uso de la función *send\_rc\_control(v1,v2,v3,v4)*, ya que con el uso de una sola función se puede controlar el dron en todos sus movimientos. Por lo que, en el caso de tener que realizar un movimiento que consista en ascender 10 cm, mover el dron a la izquierda 5 cm y girarlo 20 grados de forma secuencial, es decir, primero subir 10 cm luego desplazarse 5cm a la izquierda y finalmente girar 20 grados, utilizaremos la función *send\_rc\_control* que permite realizar todos los movimientos a la vez.

## 5.2 Validación Detección

Para la validación del sistema de detección se compararán tres métodos de detección como se puede ver en FIGURA 5.1, Manual (Select ROI), YOLOv3 y Haar Cascade. El primero de estos consiste en una selección de la región de interés de forma manual usando el ratón del ordenador y marcando con un rectángulo donde se encuentra la pelota de fútbol. Mientras tanto, por el contrario, los detectores YOLOv3 y Haar Cascade, requieren de una fase de aprendizaje automático, con la que se detecta la región de interés de forma automática.

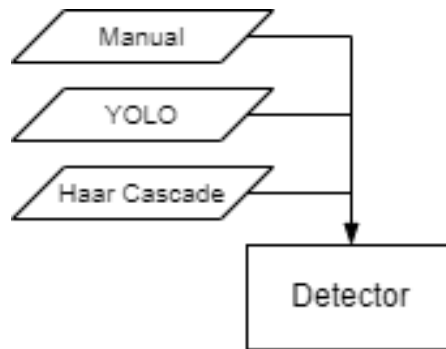


FIGURA 5.1 ALGORITMOS DETECCIÓN

Para ver qué algoritmo de los tres tiene un mejor rendimiento se han realizado tres experimentos diferentes, utilizando tres métricas de calidad Tiempo de Inferencia Medio (TIM) (9), Ratio de Acierto Medio (RAM) (10) del algoritmo y la Accuracy (11) del algoritmo.

$$TIM = \frac{\sum_i^{Num.Imágenes} Tiempo\_de\_Inferencia(i)}{Num.Imágenes} \quad (9)$$

$$RAM = \frac{\sum_i^{Num.Imágenes} \frac{detecciones(i)}{balones(i)}}{Num.Imágenes} \quad (10)$$

$$Accuracy = \frac{VN + VP}{VN + VP + FN + FP} \quad (11)$$

Los valores que se han usado para poder realizar una comparación y de esta forma averiguar qué algoritmo es mejor, se han obtenido de un vídeo de duración de dos minutos y se ha ido analizando fotograma a fotograma con los tres algoritmos diferentes, y se ha

obtenido el tiempo que ha tardado en detectar el objeto y si lo ha detectado de forma correcta o no.

Este proceso se ha desarrollado usando Python 3.8 en el programa de desarrollo PyCharm 2021.2.2. Los experimentos se han llevado a cabo en el Sistema Operativo de Windows 10, 64 bits, con un procesador AMD Ryzen 7 3800X con 8 núcleos de CPU, 3.9GHz con 32 GB RAM y 2x1TB SSD + 2x2TB HDD de memoria interna y con una tarjeta gráfica de Nvidia GeForce RTX 2060.

El Tiempo de Inferencia Medio (TIM), es el tiempo necesario para resolver la detección en una imagen del objeto, en este caso la pelota de fútbol. Para obtener los resultados de esta métrica se ejecutó la ecuación (9), que consiste en sumar el tiempo que ha tardado en realizar la detección de cada fotograma y dividirlo por el total de fotogramas que hay en el vídeo analizado y usando esto se ha podido obtener el siguiente gráfico.

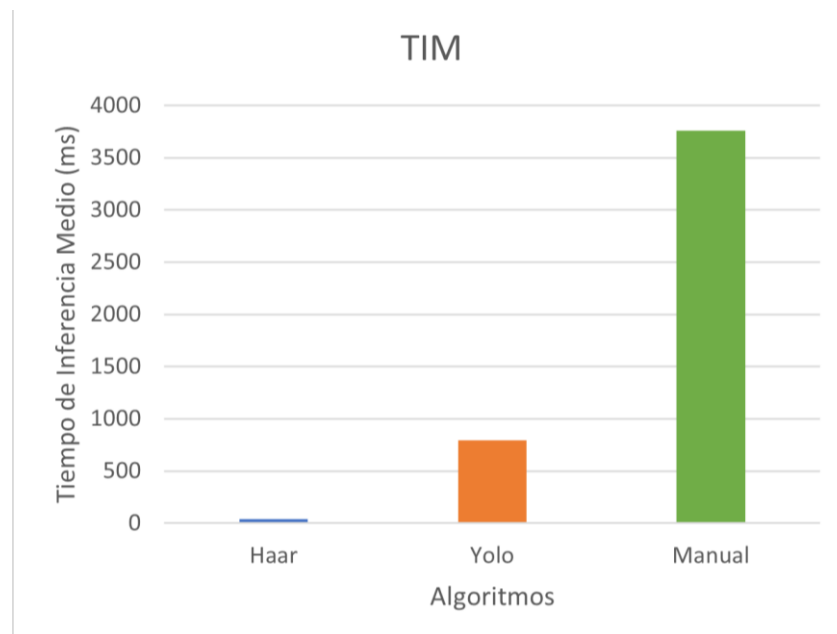


FIGURA 5.2 COMPARACIÓN TIM DETECCIÓN

El gráfico muestra los tiempos de los tres algoritmos usados, como se puede observar el algoritmo Haar Cascade es el algoritmo que menos tarda en ejecutarse y por el contrario el algoritmo de selección manual es el que más tiempo emplea en detectar el objeto. Esto se debe a que este algoritmo se realiza de forma manual, ya que entre que se encuentra el objeto y se selecciona pasa bastante tiempo. Por tanto, como se está trabajando en milisegundos para poder ver cuál de los tres algoritmos tarda menos en ejecutarse podemos apreciar bien la diferencia entre la selección manual y la selección mediante aprendizaje automático ya que nos encontramos con una diferencia de casi tres segundos.

La siguiente métrica para analizar es el Ratio de Acierto Medio (RAM). Dicha métrica representa la cantidad de balones de fútbol detectados correctamente en cada frame y la cantidad de balones que debería haber representado.

Como estamos ante un caso en el que solamente existe una pelota en un partido de fútbol la parte superior de la ecuación (10) en la que se realiza la división  $\frac{detecciones(i)}{balones(i)}$ , la parte inferior es un 1 para todos los casos por lo que la ecuación se convierte en:

$$RAM = \frac{\sum_i^{Num.Imágenes} detecciones(i)}{Num.Imágenes} \quad (12)$$

Aplicando esta ecuación a los datos recogidos y analizados se obtienen los resultados de la siguiente gráfica:

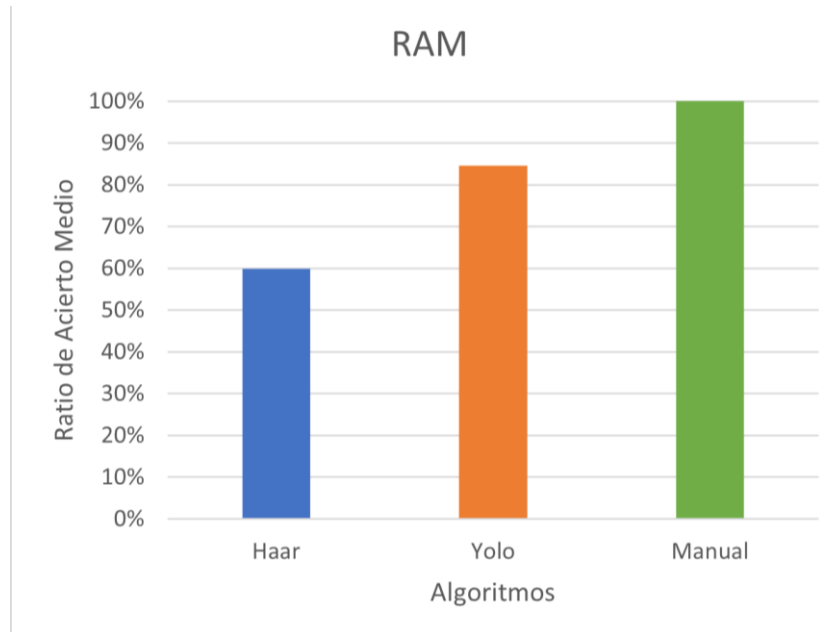


FIGURA 5.3 COMPARACIÓN RAM DETECCIÓN

Se puede observar en el gráfico que la selección de la región de interés de manera manual obtiene unos resultados perfectos, dado que la pelota es seleccionada por un humano y que por el contrario los algoritmos de aprendizaje automático obtienen resultados algo inferiores. Esto se debe a que los modelos de detección automática del balón no son del todo perfectos y que puede haber algo de ruido a la hora de la selección de la región, ruido que es eliminado con la detección manual.

Finalmente, la última métrica es la Accuracy de los algoritmos que se evalúa con los términos de Verdadero Positivo (VP), Verdadero Negativo (VN), Falso Positivo (FP) y Falso Negativo (FN). Estos valores se han obtenido separando los fotogramas en carpetas diferentes de forma manual. Si en el fotograma estaba la pelota correctamente detectada esta pertenecía a la clase VP, si el algoritmo predecía que no había ninguna pelota en la imagen, pertenecía a la clase VN. Por el contrario, si existe una pelota en la imagen y el detector dice que no existe una pelota en el fotograma o selecciona como pelota otro elemento de la imagen, pertenece a la clase FN. Y por último, si en la imagen no hay pelota pero se selecciona otro elemento es considerado como FP.

Usando la ecuación (11) sobre los tres algoritmos se han obtenido los resultados de la FIGURA 5.4.



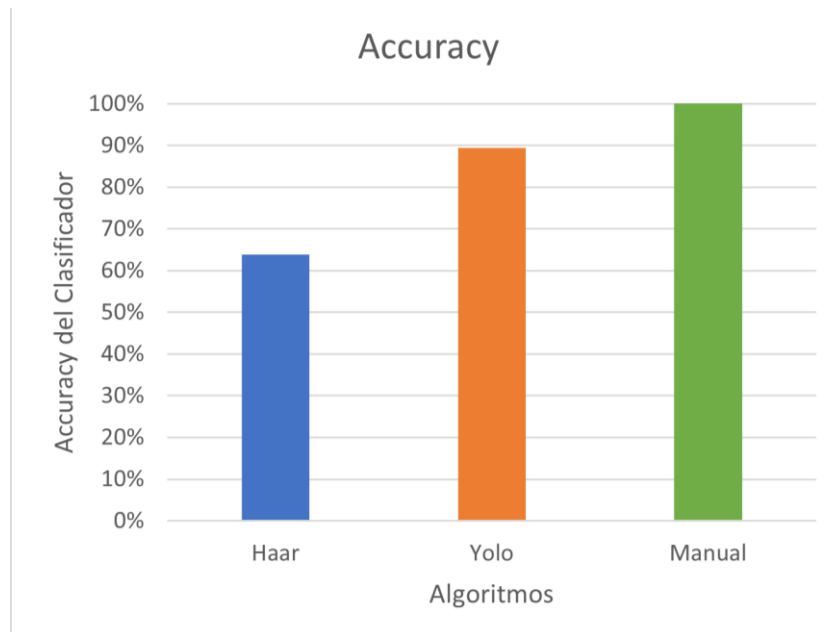


FIGURA 5.4 COMPARACIÓN ACCURACY DETECCIÓN

ALGORITMOS	TIM (ms)	RAM (%)	ACCURACY (%)
Haar Cascade	39.95	60	64
YOLOv3	792.29	85	89
Manual	3755.05	100	100

TABLA 5.1 RESULTADOS DETECCIÓN

Al igual que se pudo analizar con la FIGURA 5.3, la accuracy del método manual es 100% perfecto, debido a que un humano es capaz de seleccionar siempre de forma correcta la región de interés, y en esta gráfica, como es entendible, los algoritmos de selección automática tienen una accuracy inferior.

Como se ha podido observar en los dos últimos gráficos (FIGURA 5.3 y FIGURA 5.4), algoritmo de selección manual es aquel que tiene un porcentaje de acierto del 100% siendo esto lo idóneo, pero sin embargo, podemos observar que el tiempo de inferencia medio es superior a los 3.500 ms lo que es lo mismo a 3 segundos y medio.

Ante este tiempo tenemos que descartar el método manual debido a que el tiempo que tarda en detectar la pelota es muy elevado. Dado que, como vemos en el diagrama esquematizado de la FIGURA 5.5, podemos ver que en caso de pérdida del objeto por parte del seguimiento, flecha False, este tiene que volver a detectar dónde se encuentra la pelota de fútbol, por lo que sí el algoritmo manual tarda de media en detectar la pelota 3,5 segundos, si el algoritmo de seguimiento no fuese muy acertado, se tendría que estar analizando la imagen de forma continuada para detectar el objeto y, por lo tanto, se perderían bastantes jugadas de un partido de fútbol.

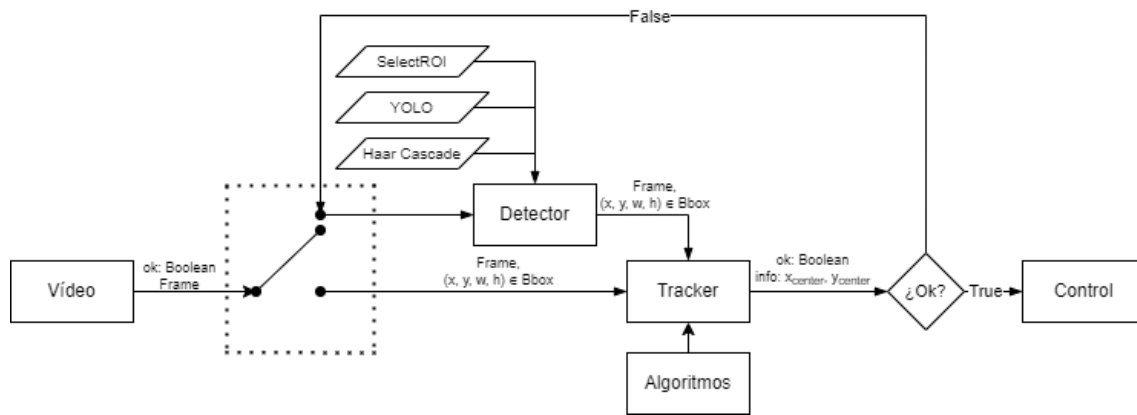


FIGURA 5.5 DIAGRAMA ESQUEMATIZADO BUCLE DETECCIÓN

Ante esta situación solo nos queda analizar cuál de los algoritmos de aprendizaje automático son mejores. Si Haar Cascade o YOLO. Para ello vamos a empezar a analizar el tiempo de inferencia medio.

Ya hemos visto anteriormente que YOLO tiene un tiempo superior a Haar Cascade en unas 20 veces. Esto nos llevaría a descartar YOLO como algoritmo, pero, también, como se puede ver tanto en las gráficas del ratio de acierto medio como de la accuracy YOLO tiene un porcentaje superior a Haar Cascade exactamente YOLO es un 25% superior en ambas gráficas.

Finalmente, concluimos que el mejor algoritmo y el que es usado en el proyecto es YOLO, porque aunque tarda más que Haar Cascade este no supera 1 segundo en reconocer el objeto, lo que lo convierte en un tiempo aceptable en realizar la detección.

Además, tiene un porcentaje de acierto a la hora de encontrar el objeto superior que Haar Cascade.

### 5.3 Validación Seguimiento

La validación del sistema de seguimiento se ha realizado con el fin de saber que algoritmo de tracking realiza un mejor desempeño que el resto, este algoritmo será el utilizado para realizar el seguimiento de la pelota en el campo. Como podemos ver en la FIGURA 5.6 se han usado siete algoritmos diferentes para seguir la región de interés detectada por el algoritmo YOLOv3.

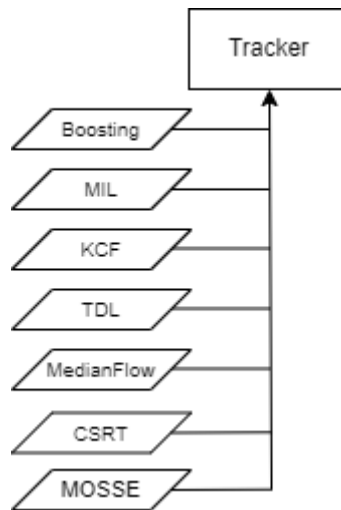


FIGURA 5.6 ALGORITMOS SEGUIMIENTO

Para ver que algoritmo tiene un mejor rendimiento se han realizado cuatro experimentos diferentes, utilizando cuatro métricas de calidad Tiempo de Inferencia Medio (TIM) (9), Ratio de Acierto Medio (RAM) (10) del algoritmo, la Accuracy (11) del algoritmo y la Intersección sobre la Unión (IoU) (13) de la región de interés detectada.

$$IoU = \frac{\text{Área Intersección}}{\text{Área Unión}} \quad (13)$$

Los valores que se han usado para poder realizar una comparación y de esta forma averiguar qué algoritmo es mejor, se han obtenido del mismo vídeo que se ha usado en la validación de la detección. Entre los valores recogidos encontramos el tiempo que tarda en procesar cada frame, si el frame observado lo ha detectado de forma correcta o no, si lo ha perdido o no y cuantas veces lo ha perdido, el tiempo empleado total en realizar el vídeo y las coordenadas de la *bounding box* [x, y, w, h].

Este proceso se ha desarrollado usando el mismo lenguaje de programación y de ordenador utilizado en la validación de detección de objetos.

Para ver qué algoritmo es mejor que otro se ha utilizado parte del script final, en el cual se utiliza el bucle de detección y seguimiento del objeto (FIGURA 5.7) sin la parte de control. La función de este bucle se centra en si el algoritmo de seguimiento devuelve la información de que no ha sido capaz de reconocer dónde se encuentra la pelota. El algoritmo de detección, YOLO, es el encargado de encontrar la pelota y si no la encuentra vuelve a intentarlo y cuando la pelota se encuentre de nuevo, el algoritmo de seguimiento vuelve a funcionar correctamente.

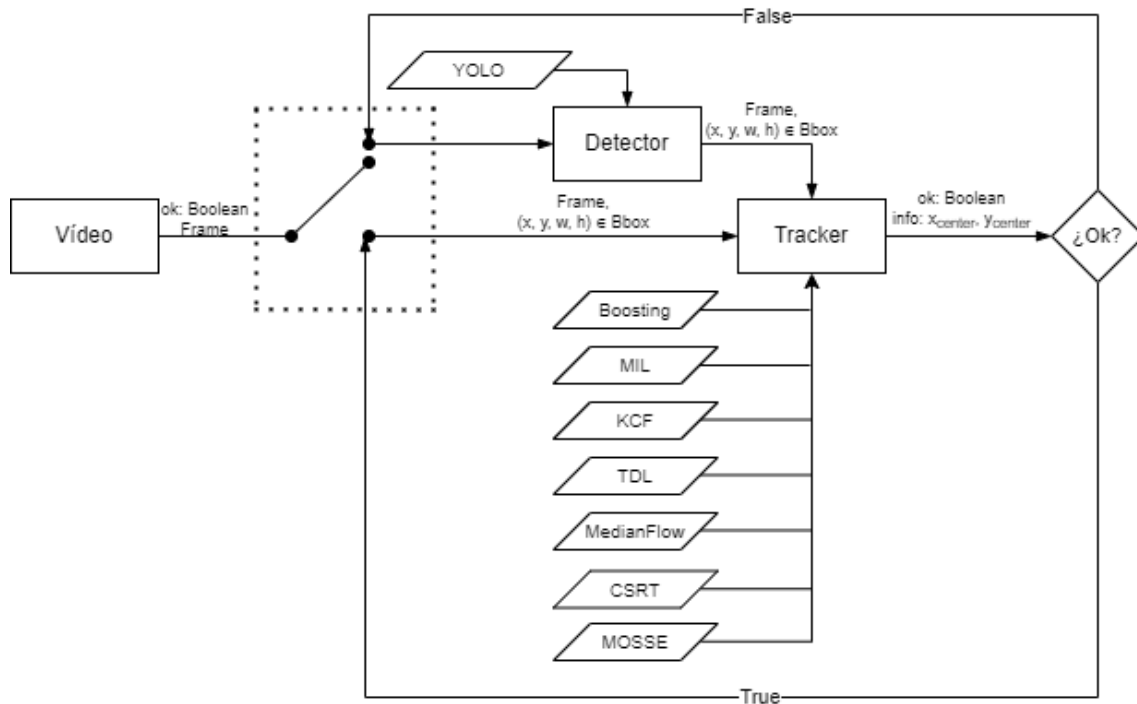


FIGURA 5.7 DIAGRAMA ESQUEMATIZADO BUCLE SEGUIMIENTO

Se ha analizado el bucle en general, tanto detección como seguimiento juntos, debido a que hay algunos algoritmos de seguimiento como KCF, MedianFlow, CSRT y MOSSE, ya que si solamente analizáramos los algoritmos de seguimiento no se podían comparar entre ellos puesto que apenas se obtenían datos útiles para trabajar con ellos.

Una vez comentado como se ha realizado el proceso para obtener los datos para la experimentación de la parte del seguimiento, toca analizar los diferentes experimentos.

La primera métrica que va a ser analizada es el Ratio de Acierto Medio (RAM), la cual representa la cantidad de balones de fútbol detectados correctamente en cada frame y la cantidad de balones que debería haber representado. Al igual en la validación de la detección en el vídeo analizado solo hay un balón de fútbol por lo que la ecuación usada será (11). Con esta ecuación se han obtenido los siguientes resultados:

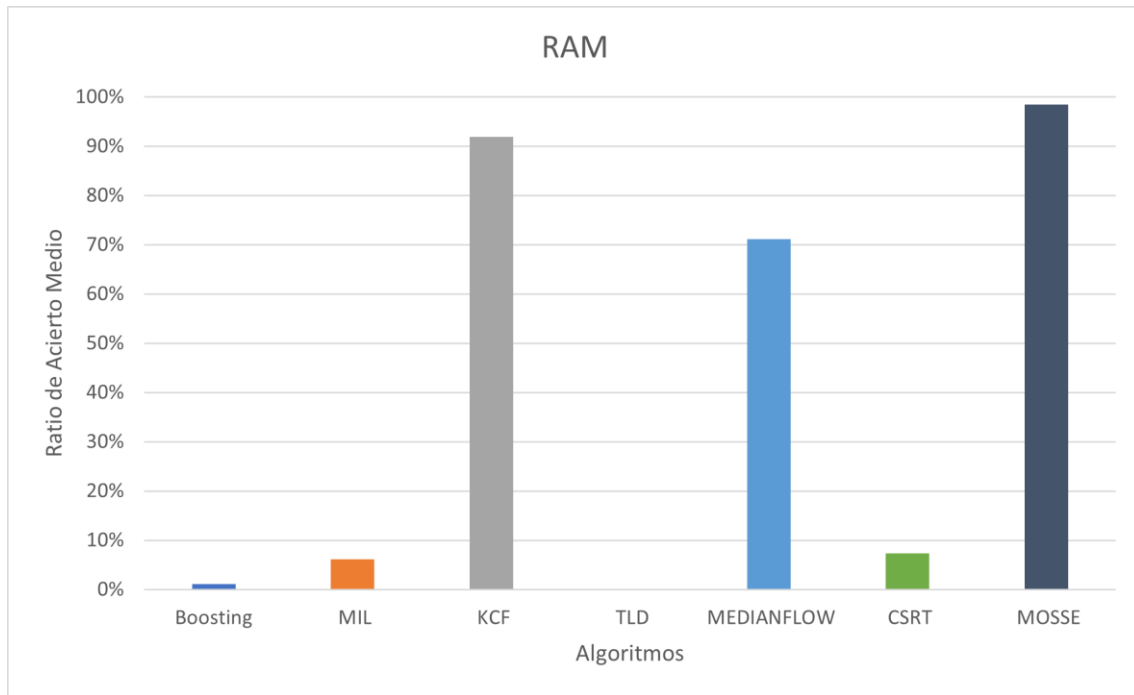


FIGURA 5.8 COMPARACIÓN RAM SEGUIMIENTO

ALGORITMOS	RAM (%)
Boosting	1.10
MIL	6.13
KCF	91.85
TLD	0
MEDIANFLOW	71.16
CSRT	7.37
MOSSE	98.44

TABLA 5.2 RESULTADOS RAM SEGUIMIENTO

Se puede observar en el gráfico que el seguimiento de la región de interés por parte de algoritmos como Boosting, MIL, TLD o CSRT, obtiene unos resultados muy bajos, que apenas superan el 10 %. Por lo que podrían ser totalmente descartados, pero antes de descartarlos, analizaremos la accuracy de estos algoritmos.

La métrica de la Accuracy de los algoritmos se evalúa con los términos de Verdadero Positivo (VP), Verdadero Negativo (VN), Falso Positivo (FP) y Falso Negativo (FN). Si en el fotograma estaba la pelota correctamente detectada esta pertenecía a la clase VP, si el algoritmo predecía que no había ninguna pelota en la imagen pertenecía a la clase VN. Por el contrario, si existe una pelota en la imagen y el detector dice que no existe una pelota en el fotograma o selecciona como pelota otro elemento de la imagen, pertenece a la clase FN. Y finalmente, si en la imagen no hay pelota pero selecciona otro elemento es considerado como FP. Una vez analizado esto y usando la ecuación (12) se ha podido obtener el siguiente gráfico.

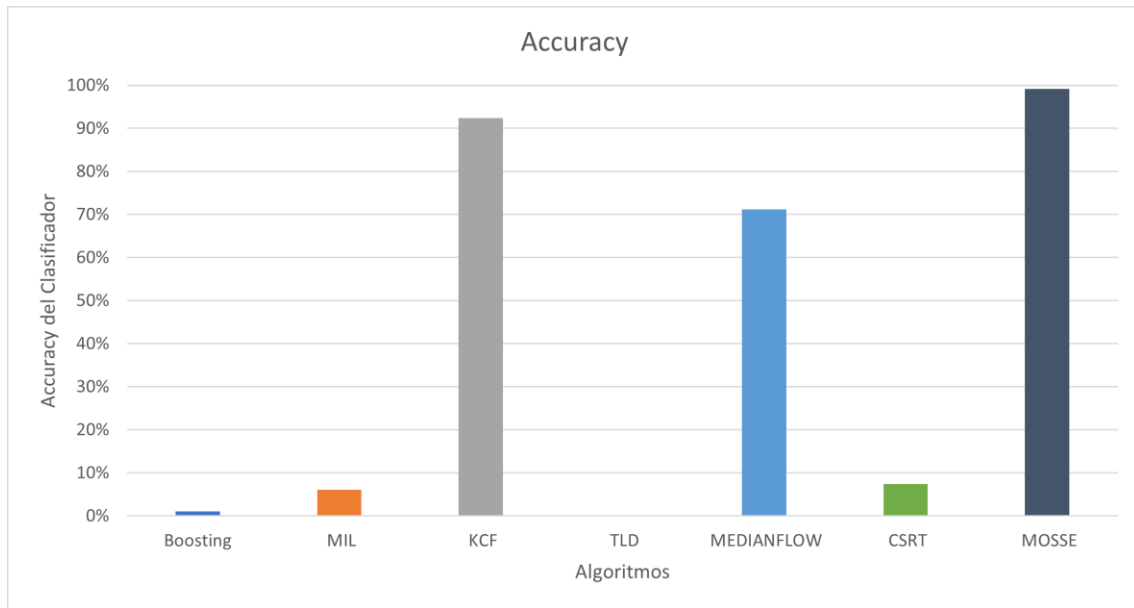


FIGURA 5.9 COMPARACIÓN ACCURACY SEGUIMIENTO

ALGORITMOS	ACCURACY (%)
Boosting	1.10
MIL	6.13
KCF	92.39
TLD	0
MEDIANFLOW	71.21
CSRT	7.38
MOSSE	99.18

TABLA 5.3 RESULTADOS ACCURACY SEGUIMIENTO

Al igual que ocurre con el cálculo del Ratio de Acierto Medio en la Accuracy, los algoritmos Boosting, MIL, TLD o CSRT, los resultados que obtiene cada algoritmo son muy bajos, apenas superan el 10 %. A pesar de que hay algunos valores que aumentan, este incremento apenas es superior al 1 % en todos los algoritmos. Por lo que, ante esta situación los algoritmos por debajo del 10 % pueden ser descartados, y de esta forma disminuir la comparación de algoritmos de siete a tres.

La siguiente métrica analizada es la Intersección sobre la Unión, métrica que devuelve la bondad de ajuste de la región de interés obtenida sobre la esperada usando solamente los algoritmos restantes KCF, MEDIANFLOW y MOSSE. Esta medida implica que un valor cercano al 100 % garantiza un ajuste bueno y un valor cercano al 0 % garantiza un ajuste nulo. Para ello, esta métrica divide la intersección entre la unión.



FIGURA 5.10 DIVISIÓN IoU

Aplicando la ecuación (13), se han obtenido los siguientes resultados:

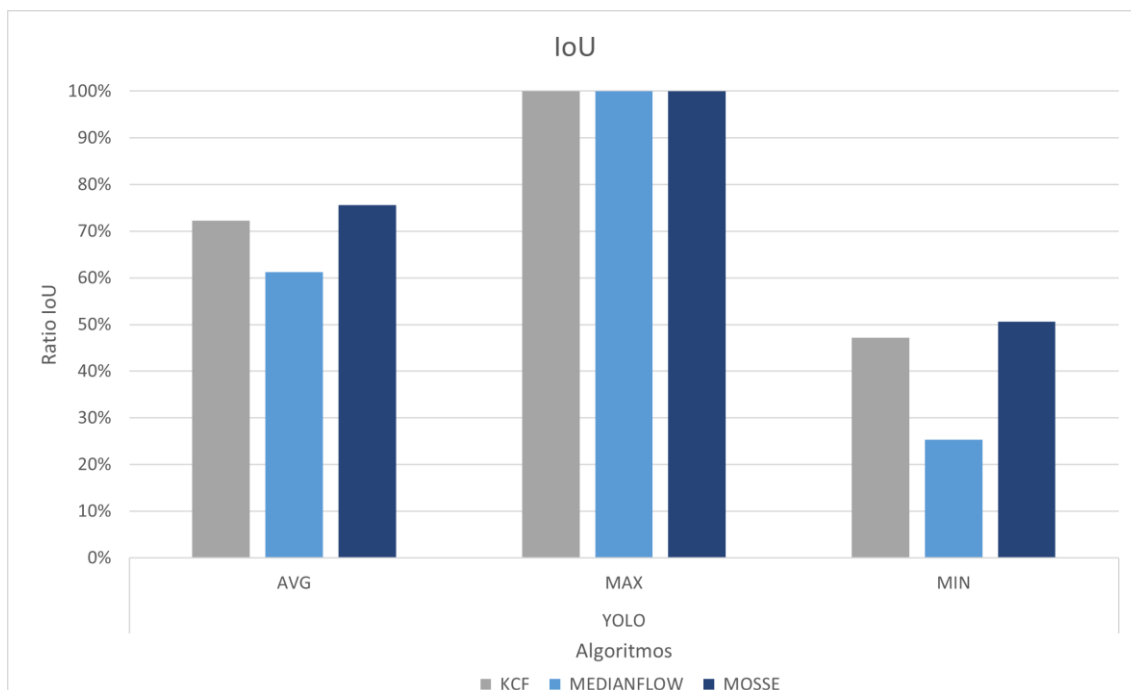


FIGURA 5.11 COMPARACIÓN IoU SEGUIMIENTO

IoU			
ALGORITMOS	MEDIA (%)	MÁXIMO (%)	MÍNIMO (%)
KCF	72.29	100	47.22
MEDIANFLOW	61.19	100	25.35
MOSSE	75.62	100	50.64

TABLA 5.4 RESULTADOS IoU SEGUIMIENTO

Como se puede apreciar en el gráfico, los valores máximos son iguales para los tres algoritmos, y sin embargo, notamos una pequeña diferencia en la media y en el valor mínimo de los algoritmos.

Se aprecia que KCF y MOSSE son algoritmos muy similares a la hora de obtener la región interés requerida y que MEDIANFLOW es un algoritmo algo peor que los dos anteriores, ya que apenas supera el 60 % de ajuste y los otros dos algoritmos se encuentran por encima del 70 % de ajuste.

Esto provocaría que tuviésemos que descartar el algoritmo MEDIANFLOW pero para confirmarlo se va a realizar un estudio del tiempo que tarda en detectar la pelota de fútbol.

Para este experimento se requiere el uso de la métrica Tiempo de Inferencia Medio (TIM), tiempo necesario para resolver la detección en un fotograma de la pelota de fútbol. Para obtener los resultados de esta métrica se ejecutó la ecuación (9) y se ha obtenido el siguiente gráfico y resultados.

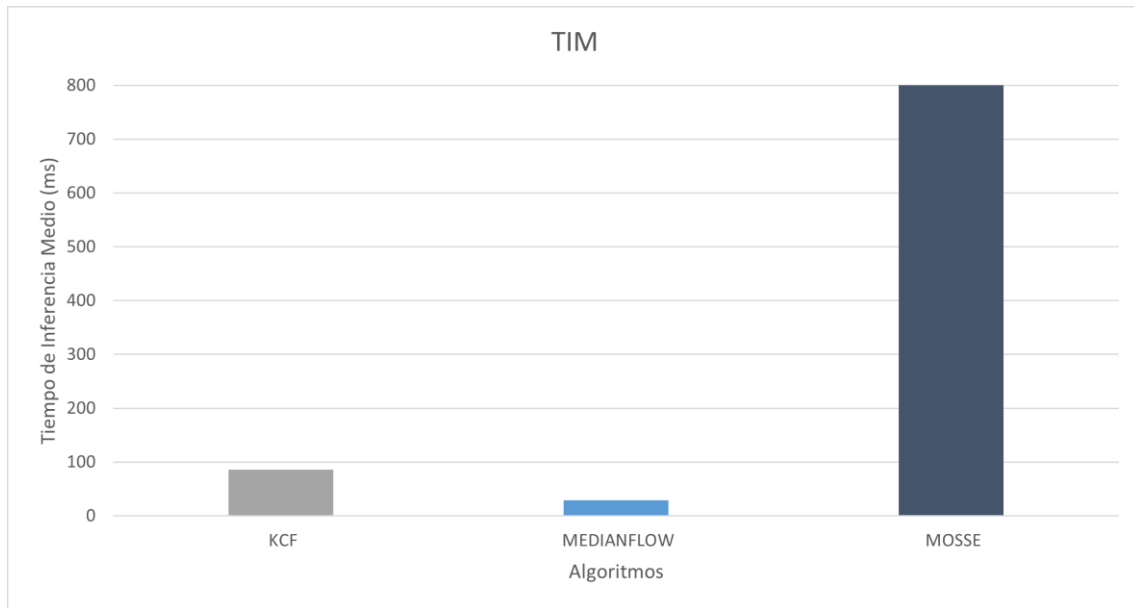


FIGURA 5.12 COMPARACIÓN TIM SEGUIMIENTO

ALGORITMOS	TIM (ms)
KCF	85.71
MEDIANFLOW	28.99
MOSSE	802.03

TABLA 5.5 RESULTADOS TIM SEGUIMIENTO

En el gráfico y en la tabla se puede observar cómo el algoritmo MEDIANFLOW tiene un tiempo de inferencia medio muy bajo cercano a los 30 ms, pero, por el contrario, en el experimento realizado sobre la métrica IoU, las medidas son muy bajas comparándolas con los otros dos algoritmos. Esto provoca que a pesar de tardar muy poco en detectar la pelota, a veces no la detectaría de forma correcta por lo que este algoritmo queda descartado.

MOSSE es un algoritmo que tarda alrededor de 800 ms en detectar la pelota en la imagen, lo que si le sumamos los 800 ms por parte del detector lo convierte en un valor muy alto para poder realizar la detección y el seguimiento de manera fluida, dado que se acerca a los 2 segundos. Esta situación provoca que el algoritmo MOSSE también quede descartado.

Finalmente, el algoritmo KCF, tiene un tiempo de inferencia medio que no supera los 90 ms, por lo que el conjunto de detección y seguimiento de la pelota de fútbol no supera el segundo, lo que hace que lo convierta en el algoritmo principal. Además, uniéndolo a su alto porcentaje de RAM, Accuracy y IoU lo convierte en el mejor algoritmo de los siete que han sido analizados.

#### 5.4 Validación Control

Para la validación del sistema de control se compararon 5 medidas diferentes, para el componente de control PID, donde solamente se le aportan valores a la parte PD de control, por lo que los valores de la parte integral (I) siempre serán ceros.



Dado que el proceso de ajuste de los valores del PID se ha realizado en un entorno ideal, donde no hay viento, y, por tanto, no hay desviaciones producidas por este viento, los valores del PID son valores que se tienen que encontrar entre el 0 y el 1.

La selección de los valores del PID se ha realizado de manera manual siguiendo la siguiente heurística.

Se empieza por valores medios, si hay una gran oscilación en el tiempo y no se estabiliza el error en el tiempo se disminuye la parte derivativa a la mitad, debido que puede ser un valor muy alto y los valores enviados al dron desde una iteración a otra pueden ser muy dispares haciendo que el dron tenga muchas oscilaciones de izquierda a derecha.

Una vez se ha probado con el valor derivativo nuevo, si sigue habiendo alguna oscilación evidente se procederá a disminuir el valor proporcional también a la mitad. Y se realiza este proceso hasta que encontremos unos valores con los que la oscilación sea mínima y se estabilice el error.

Para realizar este proceso y obtener los valores del PID que se van a usar se ha realizado un script en el que se usa como detector el algoritmo YOLOv3 y seguimiento KCF, ya que, son los algoritmos que han sido seleccionados anteriormente como los más precisos para este proyecto. Y se ha ejecutado el programa sobre una pelota estática sobre un campo de fútbol y de esta forma observar el error a lo largo del tiempo.

Este proceso se ha desarrollado usando Python 3.8 con Tello APIs en el programa de desarrollo PyCharm 2021.2.2. Los experimentos se han llevado a cabo en el Sistema Operativo de Windows 10, 64 bits, con un procesador AMD Ryzen 7 3800X con 8 núcleos de CPU, 3.9GHz con 32 GB RAM y 2x1TB SSD + 2x2TB HDD de memoria interna y con una tarjeta gráfica de Nvidia GeForce RTX 2060.

Una vez explicado el procedimiento los primeros valores que se han escogido para empezar son el [0.5,0,0.5] ya que son valores medios y si ocurre lo contrario a la heurística establecida se hace el proceso contrario.

En la TABLA 5.6 se pueden ver los valores que se han ido escogiendo, viendo el comportamiento del dron

P	I	D
0.5	0	0.5
0.5	0	0.25
0.25	0	0.25
0.25	0	0.125
0.125	0	0.125

TABLA 5.6 VALORES PID

Para empezar como se ha comentado en la heurística se ha comenzado con un valor intermedio del PID, de esta forma si el error no se estabilizaba en el tiempo pero provoca que el dron tenga poca oscilación habría que subir los valores del PID. Sin embargo, como se puede observar en la gráfica, y a pesar de que el error inicial es mínimo debido a que se inicializa el detector cuando la pelota se encuentra encuadrada en el centro, en el momento en el que recibe un pequeño error, como se puede observar en la siguiente gráfica alrededor de los 5 segundos, este empieza a incrementarse hasta que deja de

detectar la pelota en la pantalla en el segundo 10, de ahí que el error pase a ser 0 constantemente debido a que no se puede calcular dicho error.

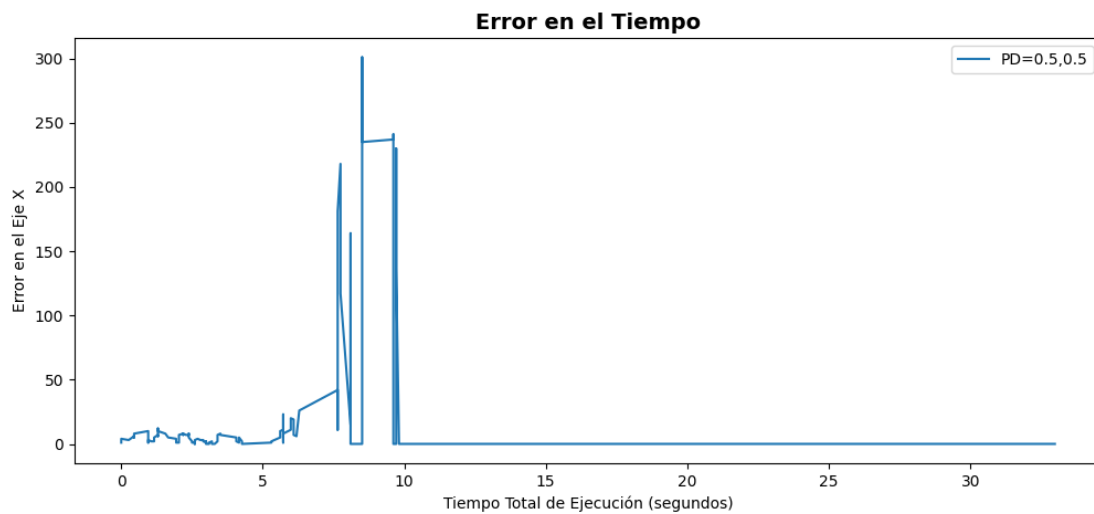


FIGURA 5.13 VALOR PID 0.5,0,0.5

Una vez visto que los valores del PID,  $[0.5,0,0.5]$ , no daban un buen resultado y siguiendo la heurística, se ha decidido disminuir el valor de la parte derivativa a la mitad siendo este nuevo valor 0.25, su gráfica es la siguiente.



FIGURA 5.14 VALOR PID 0.5,0,0.25

En esta gráfica se puede apreciar que ocurre algo similar al PID inicial, salvo por el hecho de que el error inicial es algo más estable. Pero a pesar de esto, llega un momento en el que después de intentar realizar la corrección de posición el dron deja de enfocar a la pelota. Esto provoca que estos nuevos valores queden descartados y se divida el valor proporcional por la mitad. Y como se observa en la FIGURA 5.15 el error es inferior, siendo este inferior a 50, no como en los errores producidos y analizados anteriormente. Además, en este caso la pelota permanece todo el tiempo en pantalla lo que es un avance con respecto a los casos anteriores.

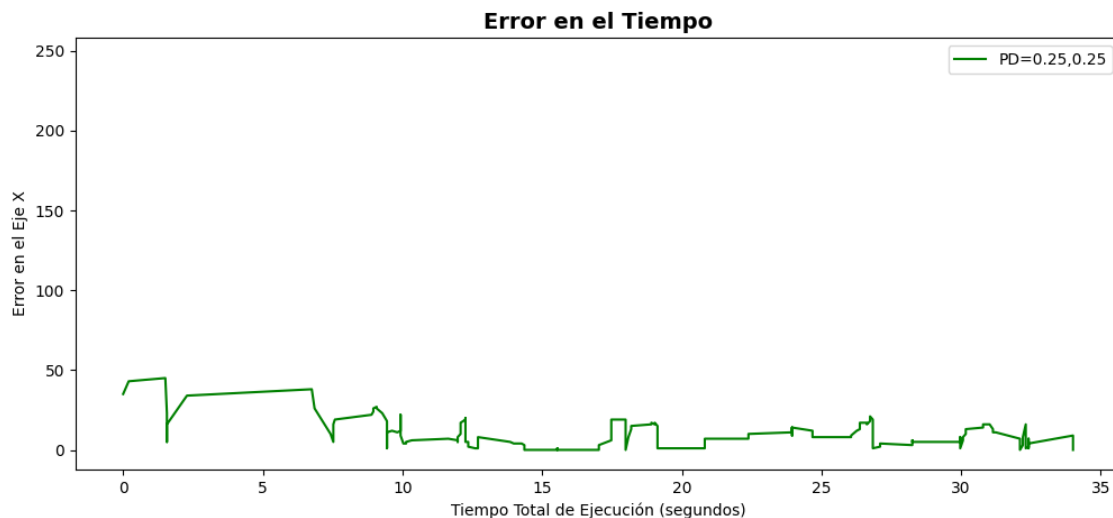


FIGURA 5.15 VALORES PID 0.25,0,0.25

El siguiente paso para ver si los valores anteriores son los finales o no es analizar el siguiente valor derivativo el cual es 0.125. Este nuevo valor da un error medio de 10.8803 que comparado con el error medio anterior que es 9.9909 es algo superior. Por lo que, al ser solamente inferior a 1 unidad.

El siguiente paso es analizar el valor del PID de [0.125,0,0.125], que tiene un error medio de 17.1658. Si observamos la siguiente gráfica vemos como la línea morada que pertenece como indica la leyenda al valor del PD [0.125,0.125] sufre bastante más oscilaciones que la verde y la roja.

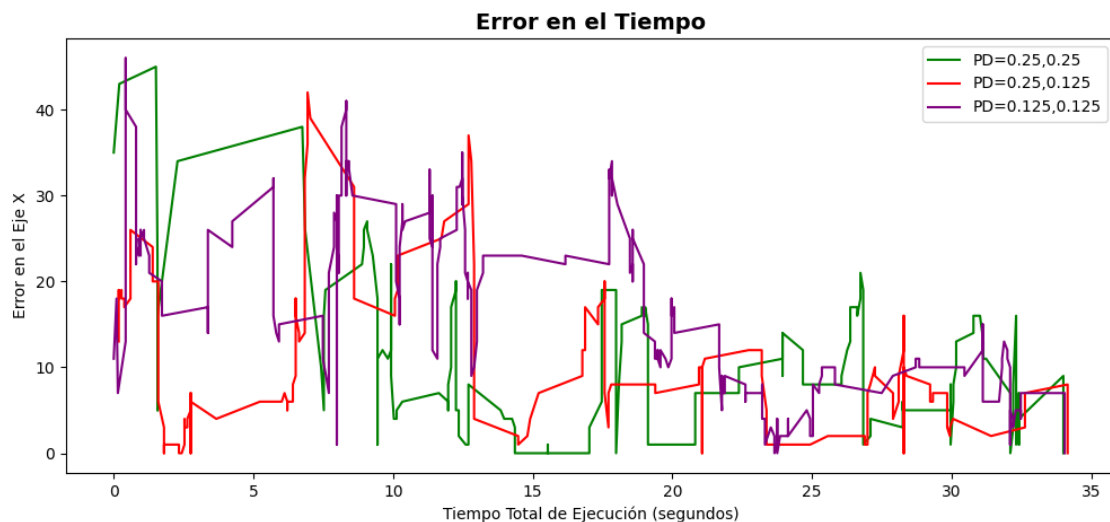


FIGURA 5.16 VALORES PID COMPARACIÓN

Ante esta situación solamente queda analizar los dos valores comentados anteriormente, la línea roja y la verde del gráfico anterior.

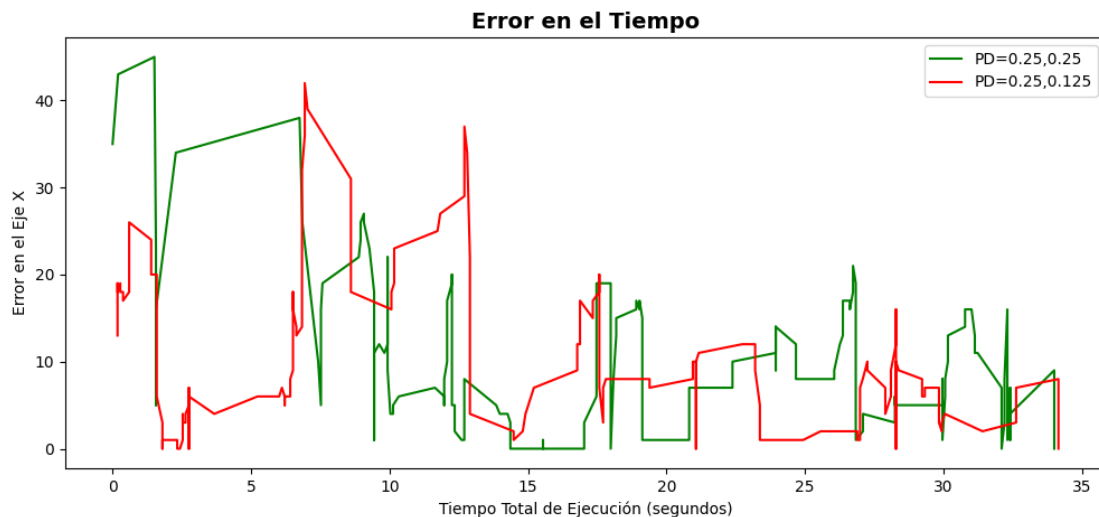


FIGURA 5.17 VALORES PID FINALES

Aquí se puede apreciar que el gráfico verde parece tener errores algo superiores que los que marca la línea roja. Pero los valores verdes tienen cambios algo más suaves produciendo que, aunque el error se alargue en el tiempo, el movimiento de la cámara es algo más fluido que el valor representado por la línea roja, debido a que los cambios como se puede apreciar en la FIGURA 5.17 son cambios más bruscos, pasan de un error de cerca de 40 a casi 5 en el segundo 12 por ejemplo, esto es producido por un giro brusco de la cámara.

Finalmente, y analizado que el movimiento del dron es algo más fluido usando el valor del PID de  $[0.25,0,0.25]$  es el elegido para el desarrollo final del proyecto.

## 6 CONCLUSIONES Y TRABAJOS FUTUROS

Después de haber realizado todo el proceso de investigación, diseño, implementación y prueba, se pueden extraer algunas conclusiones para realizar una valoración global del trabajo fin de grado.

Como se definió al principio, el proyecto estaba dividido principalmente en tres partes: la inicialización del sistema, el módulo de visión y el módulo de control.

La primera parte, requería como eje central la comprensión del funcionamiento del protocolo UDP que se usa en este proyecto, porque es un protocolo que realiza una transmisión de vídeo eficaz, aparte de porque era la única forma de conectar un portátil (estación de tierra) al DJI Tello. También cabría destacar el buen funcionamiento del código realizado en Python en el entorno de desarrollo integrado PyCharm.

La segunda parte se corresponde con el módulo de visión, formado por dos partes, el detector y el tracker.

Con respecto al detector se analizaron tres tipos de algoritmos, uno de ellos selecciona la región de interés de forma manual y los otros dos seleccionan la región de interés de manera automática (YOLO y Haar Cascade). Estos dos últimos algoritmos se han creado mediante aprendizaje profundo.

Una vez creados los modelos finales, se ha hecho un análisis de los tres algoritmos y se observó que el método manual era infalible, ya que una persona era la encargada de seleccionar la región de interés, pero esta selección tenía en contra que tardaba de media más de tres segundos y medio, tiempo que no podía ser viable porque se puede perder alguna acción. Por tanto, comparando los algoritmos automáticos se observó que YOLOv3 tardaba unas 20 veces más que Haar Cascade en detectar la pelota de fútbol; el primero tarda unos 800ms y el segundo 40ms. Y comparando accuracy y ratio de acierto medio (RAM) entre YOLOv3 y Haar Cascade se obtuvo que eran mejores los resultados del algoritmo YOLOv3.

Ante esta situación, se podía obviar la tardanza por parte de YOLOv3 en detectar la pelota, debido a que la accuracy es lo más importante para percatarse de que siempre seleccionará una región de interés adecuada. Por lo que el algoritmo de detección seleccionado finalmente ha sido YOLOv3.

Con respecto a los algoritmos de seguimiento se ha realizado un análisis usando cuatro tipos de métricas, el tiempo de inferencia medio, ratio de acierto medio, accuracy y la intersección sobre la unión. Estos algoritmos son Boosting, MIL, KCF, TLD, MEDIANFLOW, CSRT y MOSSE.

Primero se analizó el ratio de acierto medio y la accuracy, en los cuales los algoritmos de Boosting, MIL, TLD y CSRT no superan el 10% en ninguno de los casos, por lo que han sido descartados como algoritmos a usar en el proyecto. Ya solamente quedaban tres algoritmos, que analizando la intersección sobre la unión, el algoritmo con menor porcentaje es MEDIANFLOW, que a pesar de que el tiempo de inferencia medio es muy bajo, el algoritmo KCF también tiene un tiempo de inferencia medio asequible y su IoU es de 72,29% por lo que al tener valores buenos es el algoritmo de seguimiento seleccionado.

Por lo tanto, el módulo de inferencia quedaría hecho con el algoritmo YOLOv3 de detector de objetos y el algoritmo KCF como el seguidor de la pelota de fútbol que tiene como beneficio que devuelve un valor booleano False, cuando ha dejado de seguir a la pelota para que el algoritmo YOLOv3 pueda detectar la pelota.

Finalmente, queda por sacar conclusiones del último módulo, el de control. Los valores se han establecido de forma heurística, viendo como reaccionaba el dron a los valores PID establecidos. Si había mucha oscilación se disminuía el valor derivativo a la mitad y si continuaba habiendo descontrol en el vuelo se disminuía la parte proporcional y así sucesivamente hasta obtener que el error se estabilice.

Una vez dicho esto los valores PID que se han usado para el correcto vuelo del dron han sido 0.25,0,0.25. Ya que el error que tenía comparando con el resto de los valores probados era inferior y los movimientos del dron eran más fluidos haciendo que no tuviese tantas oscilaciones.

Como conclusión, se puede establecer que el diagrama final de este sistema sea como el de la FIGURA 6.1, con los algoritmos seleccionados de detección como de seguimiento, siendo estos YOLOv3 y KCF respectivamente. Y, además, quedan anotados los valores por parte del módulo de control, por lo que este diagrama final queda de la siguiente manera.

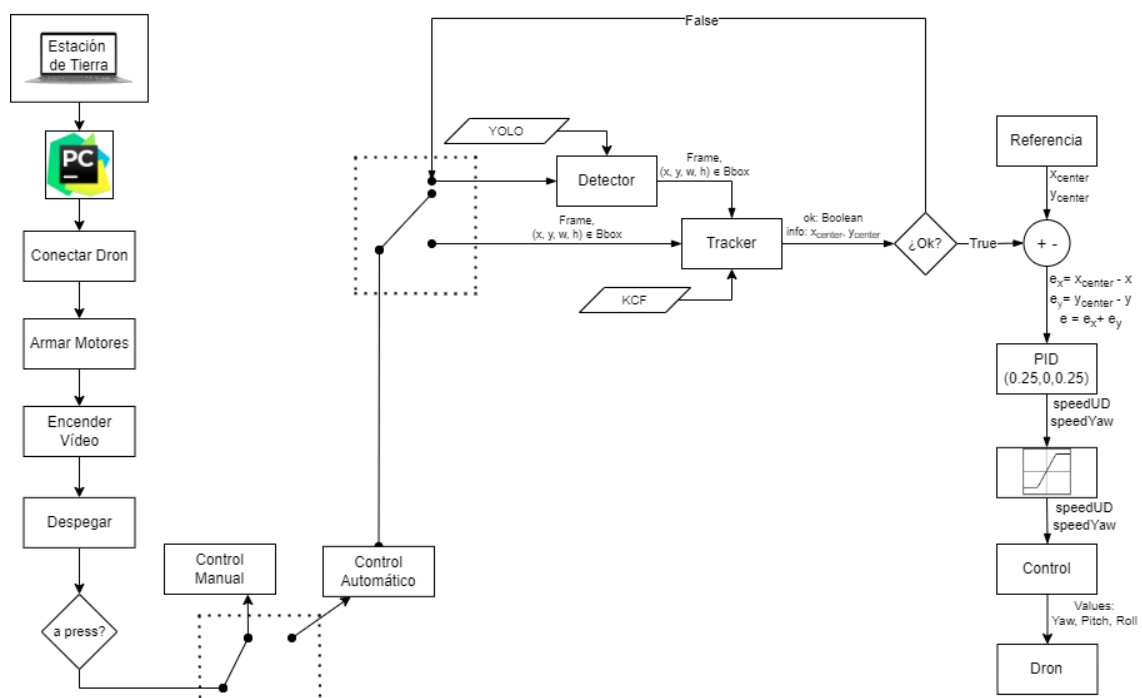


FIGURA 6.1 DIAGRAMA FINAL

En cuanto a los trabajos futuros, se establecen una mejora de los algoritmos de detección y seguimiento y del módulo de control.

Esto es debido a que en este proyecto, la principal finalidad era realizar un estudio sobre las técnicas tradicionales tanto de detección como de seguimiento, por lo que cumple con los objetivos establecidos inicialmente.

Sin embargo, como trabajo futuro se podría analizar otro tipo de algoritmos de detección que tarden menos tiempo en realizar la detección y sean algo más precisos a la hora de la detección, ya que a lo largo del desarrollo del proyecto, si la luz cambia con respecto a lo aprendido, el proceso de la detección varía mucho, incluso puede llegar a empeorar bastante. Esto provocaría que no pudiese realizar la detección en un partido de fútbol por la noche, debido a que la luz de los focos no es la misma que la del Sol.

Para solucionar este tipo de problemas habría que analizar los puntos en los que los modelos de detección y seguimiento sean más débiles y a partir de ahí marcar unas pautas para realizar modelos más robustos. Para ello, se podrían usar otras técnicas similares como pueden ser YOLOv3-tiny el cual obtiene resultados de accuracy similares a YOLOv3 pero con un tiempo de ejecución inferior.

Un ejemplo, que podría solucionar el problema de la luz, es obtener imágenes en todo tipo de circunstancias: con lluvia, de noche, de día, nublado y con diferente color de pelota y de esta forma la mayoría de las posibles combinaciones que puedan darse estén totalmente cubiertas por el modelo.

Con respecto a los algoritmos usados por parte del módulo de seguimiento, son los aportados por la librería cv2 de Python. Se puede realizar una investigación solamente sobre diferentes tipos de algoritmos de seguimiento y elegir qué tipo de tracker realiza una función más estable que los analizados para este trabajo.

A pesar de que otros algoritmos de seguimiento podrían desempeñar un trabajo mejor que el algoritmo elegido en este caso que es KCF, este algoritmo realiza un trabajo bastante estable y preciso para el proyecto y, también, en una primera línea de investigación.

Finalmente, con respecto al módulo de control, se ha realizado de una manera heurística pero en un trabajo futuro sería bastante interesante la inclusión de modelos de aprendizaje automático que fueran ajustando los valores del PID, con respecto a lo que le va ocurriendo al dron durante el vuelo. De esta manera, los valores serían casi perfectos y el vuelo del dron sería siempre estable independientemente de los valores que el modelo eligiese para cada momento.

Una idea para el inicio del modelo de aprendizaje automático para obtener los valores de control sería compartir con él los valores de la velocidad del viento en ese momento, la posición del dron y de la pelota.

Finalmente, me gustaría compartir todo lo desarrollado en este proyecto a través del link al [GitHub](#).

## 7 GESTIÓN DEL PROYECTO

### 7.1 Planificación del proyecto

Para crear cualquier tipo de proyecto, hay que tener en cuenta la planificación de este, así como el tiempo que se va a dedicar al desarrollo de dicho sistema. Este proceso consiste en establecer los objetivos del proyecto y elegir adecuadamente qué medios hay que usar para poder llegar a alcanzar dicha propuesta.

Es importante realizar un análisis en profundidad de la planificación del proyecto para evitar algunas situaciones no deseadas durante el desarrollo del trabajo que puedan afectar a su desenlace y de esta forma crear propuestas que puedan solucionar las mismas.

Para poder observar esta planificación de forma rápida y eficaz se ha hecho uso de un diagrama de Gantt, que gracias a este se puede ver el número de horas estimadas totales dedicadas a este proyecto desglosadas en las diferentes partes de las distintas etapas. Para esta estimación se ha tomado una media de unas 3 horas de trabajo diario.

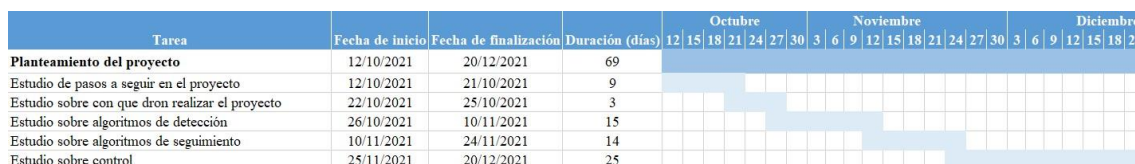


FIGURA 7.1 DIAGRAMA DE GANTT PLANTEAMIENTO

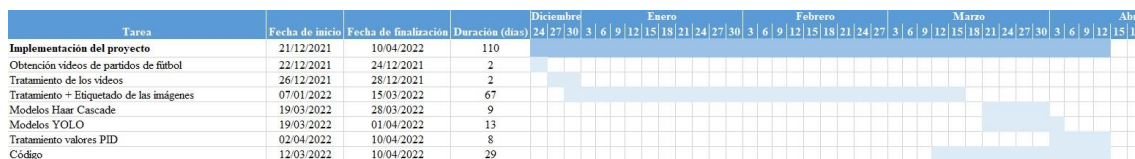


FIGURA 7.2 DIAGRAMA DE GANTT IMPLEMENTACIÓN

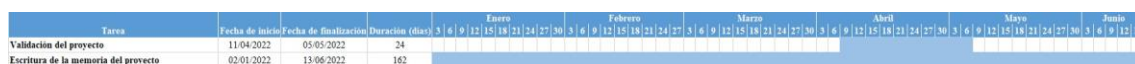


FIGURA 7.3 DIAGRAMA DE GANTT VALIDACIÓN Y ESCRITURA

Como se puede observar en las imágenes del diagrama de Gantt, el proceso más largo fue el de la implementación del proyecto, ya que la selección de las imágenes y su posterior etiquetado para poder crear dos modelos buenos con YOLOv3 y con Haar Cascade requirió muchas horas frente al ordenador. Finalmente, y después de más de medio millón de fotografías etiquetadas, podemos decir que los modelos finales son satisfactorios en gran medida.

Por otra parte, podemos ver que un gran número de horas fueron destinadas a la creación de un buen código, junto con la parte de experimentación, que después de realizar gran cantidad de vuelos se llega a la conclusión de que el sistema funciona de forma adecuada.

El proyecto ha tenido una duración total de 244 días contando tanto los días no laborables como laborables así como las vacaciones.

### 7.2 Presupuesto del proyecto

En cuanto a los presupuestos del proyecto, estos deben llevarse a cabo con cierta antelación para tener en cuenta los costes que va a conllevar su desarrollo. Estos costes



para poder ver mejor en que se tiene que gastar el dinero para poder llevarlo a cabo van a desglosarse en: costes de Hardware, costes de Software y costes de personal.

Los costes del hardware vienen principalmente de los componentes que han sido usados para la creación del sistema. Estos elementos están formados por un ordenador que es donde se ha realizado todo el proyecto, así como el dron que se ha usado para verificar el funcionamiento del sistema.

CONCEPTO	UNIDADES (ud)	PRECIO (€)	COSTE (€)
ORDENADOR			
Tarjeta Gráfica: NVIDIA RTX 2060 6GB	1	349,90	349,90
RAM: 8GB DDR4 3200Mhz	2	40,99	81,98
Procesador: AMD Ryzen 7 3800X	1	249,90	249,90
Webcam Logitech c270	1	24,99	24,99
DRON			
DJI Tello	1	100	100
Baterías	2	25	50
TOTAL HARDWARE			856,77

TABLA 7.1 DESGLOSE PRESUPUESTO HARDWARE

En la TABLA 7.1 solamente se han puesto los componentes principales del ordenador que ha sido usados, obviando por ejemplo que tipo de monitor, torre, fuente de alimentación y otros periféricos. También ha sido obviado otros costes indirectos como puede ser la electricidad y la conexión a internet.

Todo el software que ha sido usado para el desarrollo se puede ver desglosado en la TABLA 7.2, todo este software proviene de librerías y entornos de programación *OpenSource* por lo que como podemos observar en la tabla estos costes son nulos.

CONCEPTO	COSTE (€)
ENTORNO DE PROGRAMACIÓN	
PyCharm Profesional	0
OTROS	
Python	0
OpenCV	0
YOLOv3	0
Haar Cascade	0
DJITellopy	0
TOTAL SOFTWARE	0

TABLA 7.2 DESGLOSE PRESUPUESTO SOFTWARE

Finalmente, hay que tener en cuenta que para poder llevar a cabo este proyecto se necesita personal, entre este personal se va a tener un Ingeniero Informático Junior. El cual tiene de media un sueldo de 10,51 €/hora [59]. Y también se necesita un Ingeniero Senior que será el que vaya guiando al Ingeniero Junior a lo largo de todo el proceso. Este ingeniero tiene de media un salario de 17,95 €/h [60].

CONCEPTO	SUELDO BRUTO (€/h)	HORAS NECESARIAS	COSTE (€)
Ingeniero Informático Junior	10,51	609	6400,59
Ingeniero Informático Senior	17,95	304	5456.8
TOTAL RECURSOS HUMANOS			11857,39

*Tabla 7.3 DESGLOSE PRESUPUESTO RECURSOS HUMANOS*

Los costes totales de todo el proyecto se corresponden con la suma de los tres costes anteriores, hardware, software y recursos humanos. Esta suma se puede ver reflejada en la TABLA 7.4


CONCEPTO	COSTE (€)
Hardware	856,77
Software	0
Recursos Humanos	11857,39
COSTE TOTAL	12715,16

*TABLA 7.4 DESGLOSE COSTE TOTAL*

## 8 BIBLIOGRAFÍA

- [1] “Enaire registra un incremento del 370 en la demanda de operaciones con drones.” <https://www.infodron.es/texto-diario/mostrar/3529233/enaire-registra-incremento-370-demanda-operaciones-drones> (accessed Jun. 07, 2022).
- [2] “Campo de investigación Inteligencia Artificial | Festo ES.” [https://www.festo.com/es/es/e/sobre-festo/investigacion-y-desarrollo/campos-de-investigacion/campo-de-investigacion-de-inteligencia-artificial-id\\_1079428/](https://www.festo.com/es/es/e/sobre-festo/investigacion-y-desarrollo/campos-de-investigacion/campo-de-investigacion-de-inteligencia-artificial-id_1079428/) (accessed Jun. 07, 2022).
- [3] “Inicio | Fly-Fut.” <https://www.fly-fut.com/> (accessed Jun. 05, 2022).
- [4] “Veo | Lleva tu deporte al siguiente nivel.” <https://www.veo.co/es> (accessed Jun. 07, 2022).
- [5] “Pixellot You | Cámara deportiva portátil - Pixellot Air.” <https://you.pixellot.tv/es/cameras/air/> (accessed Jun. 07, 2022).
- [6] “PIX4TEAM | Cámara automática para los deportes colectivos.” <https://shop.movense.com/es/content/31-pix4team-camara-automatica-para-los-deportes-colectivos> (accessed Jun. 07, 2022).
- [7] “Write an Object Tracking Drone Application - Circuit Cellar.” <https://circuitcellar.com/research-design-hub/write-an-object-tracking-drone-application/> (accessed Jun. 07, 2022).
- [8] “DJI - Página oficial.” <https://www.dji.com/es> (accessed May 23, 2022).
- [9] “12 mejores drones con tecnología Follow me (Sígueme) - Guía Drones.” <https://guiadrones.com/reviews/mejores-drones-con-tecnologia-follow-me-sigueme/> (accessed Jun. 07, 2022).
- [10] “Sistema de control que permite el vuelo autónomo de drones - pt-PROTECMA.” <https://ptprotecma.es/ofertas/sistema-de-control-que-permite-el-vuelo-autonomo-de-drones/> (accessed Jun. 07, 2022).
- [11] “Esta es la magia detrás de la espectacular coreografía en la que un enjambre de 2.000 drones iluminó la noche de Shanghai.” <https://www.xataka.com/drones/esta-magia-detras-espectacular-coreografia-que-enjambre-2-000-drones-ilumino-noche-shanghai> (accessed Jun. 07, 2022).
- [12] “VIDEO | El futuro de la publicidad está en el cielo: Enjambre de drones creó gigantesco código QR escaneable.” <https://sonarfm.cl/tecnologia/video-publicidad-cielo-enjabre-drones-gigante-codigo-qr-28-04-2021> (accessed Jun. 07, 2022).

- [13] “Normativa de Drones en España 2022: Todo lo que debes saber | One Air.” <https://www.oneair.es/normativa-drones-espana-aesa/> (accessed Jun. 09, 2022).
- [14] “ENAIRe Drones.” <https://drones.enaire.es/> (accessed Jun. 09, 2022).
- [15] “RGPD | Reglamento Europeo de Protección de Datos | Qué es 【2022】 .” <https://ayudaleyprotecciondatos.es/guia-rgpd/> (accessed Jun. 09, 2022).
- [16] “¿Qué es un drone? - El Drone.” <http://eldrone.es/que-es-un-drone/> (accessed Mar. 21, 2022).
- [17] “PHANTOM-2-scaled.jpg (450×450).” <https://i0.wp.com/tienda-optidronmq.com/wp-content/uploads/2021/10/PHANTOM-2-scaled.jpg?resize=450%2C450&ssl=1> (accessed May 23, 2022).
- [18] “Cultura tecnológica: ¿Qué es un dron y cuáles son sus usos?” <https://edu.gcfglobal.org/es/cultura-tecnologica/que-es-un-dron-y-cuales-son-sus-usos/1/> (accessed Mar. 21, 2022).
- [19] E. M. Rodríguez, “Qué es un dron y para qué sirve - Blog de CIM Formación,” Jun. 09, 2016. <https://www.cimformacion.com/blog/aeronautica/que-es-un-dron-y-para-que-sirve/> (accessed Mar. 21, 2022).
- [20] “esquema-multirrotores.png (978×863).” <https://www.miprimerdron.com/wp-content/uploads/2016/06/esquema-multirrotores.png> (accessed May 23, 2022).
- [21] “Todas las Partes de los Drones. Explicadas al Detalle - Esenziale.” <https://esenziale.com/tecnologia/partes-drone/> (accessed Mar. 21, 2022).
- [22] “altitude-7e757661b6.png (504×403).” <https://f8c2641a47.clvaw-cdnwnd.com/3d8b8a52879c4c5a822d80db28ed3bb1/200000008-a53f8a640e/altitude-7e757661b6.png?ph=f8c2641a47> (accessed May 23, 2022).
- [23] Miguedronero, “Controladora de Vuelo (FC) ¿Que es? 🧠 | De Cero a Dronero,” Sep. 30, 2020. <https://deceroadronero.es/componentes-que-tenes-que-saber-de-tu-dron/controladora-de-vuelo-fc-que-es/> (accessed Apr. 04, 2022).
- [24] “Para qué sirven los GPS en los drones y en qué situaciones se utilizan | ¿Quieres ser piloto de drones?,” Oct. 23, 2021. <https://cursodedrones.es/para-que-sirven-los-gps-en-los-drones-y-en-que-situaciones-se-utilizan/> (accessed Apr. 04, 2022).
- [25] M. Contreras, “¿Que tipo de camara usan los drones – dronextremo.com.” <https://dronextremo.com/que-tipo-de-camaras-usan-los-drones/> (accessed Apr. 04, 2022).

- [26] “Paracaidas para drones  El Mejor Drone.” <https://elmejordrone.com/paracaidas-para-drones> (accessed Apr. 04, 2022).
- [27] J. Lopez, “Estaciones terrestres para drones de software libre – CreBots,” May 18, 2020. <https://crebots.com/estaciones-terrestres/> (accessed Mar. 18, 2022).
- [28] “Estaciones terrestres para drones de software libre – CreBots.” <https://crebots.com/estaciones-terrestres/> (accessed May 23, 2022).
- [29] “Conexiones e interfaces de comunicaciones en drones | Embention,” Sep. 14, 2018. <https://www.embention.com/es/news/conexion-e-interfaces-de-comunicacion-en-drones/> (accessed Apr. 04, 2022).
- [30] “Los protocolos TX RX de Radio control | Tienda y Tutoriales Arduino.” <https://www.prometec.net/protocolos-tx-rx-de-radio-control/> (accessed Apr. 04, 2022).
- [31] “Protocolos TCP y UDP: características, uso y diferencias,” Jun. 05, 2020. <https://www.redeszone.net/tutoriales/internet/tcp-udp-caracteristicas-uso-diferencias/> (accessed Mar. 23, 2022).
- [32] “UDP: ¿qué es UDP? - IONOS.” <https://www.ionos.es/digitalguide/servidores/know-how/udp-user-datagram-protocol/> (accessed Mar. 23, 2022).
- [33] “INTELIGENCIA ARTIFICIAL 101 COSAS QUE DEBES SABER HOY SOBRE NUESTRO FUTURO INTELIGENCIA ARTIFICIAL,” 2018, Accessed: May 20, 2022. [Online]. Available: [www.planetadelibros.com](http://www.planetadelibros.com)
- [34] “What is Artificial Intelligence (AI)? | Glossary | HPE España.” [https://www.hpe.com/es/es/what-is/artificial-intelligence.html?jumpid=ps\\_2678kiuhbz\\_aid-520061736&ef\\_id=Cj0KCQjw-JyUBhCuARIsANUqQ\\_KWAM2fnVL1\\_a-Eba2UtWe0TDKuWnt9sSoEK3Gb53xyQmf--oM7ZsUaAn6CEALw\\_wcB:G:s&s\\_kwid=AL!13472!3!569614956421!e!!g!!historia%20de%20la%20inteligencia%20artificial!14805571337!136796166891&](https://www.hpe.com/es/es/what-is/artificial-intelligence.html?jumpid=ps_2678kiuhbz_aid-520061736&ef_id=Cj0KCQjw-JyUBhCuARIsANUqQ_KWAM2fnVL1_a-Eba2UtWe0TDKuWnt9sSoEK3Gb53xyQmf--oM7ZsUaAn6CEALw_wcB:G:s&s_kwid=AL!13472!3!569614956421!e!!g!!historia%20de%20la%20inteligencia%20artificial!14805571337!136796166891&) (accessed May 20, 2022).
- [35] Na8, “Modelos de Detección de Objetos | Aprende Machine Learning,” 2020. <https://www.aprendemachinelearning.com/modelos-de-deteccion-de-objetos/> (accessed Apr. 06, 2022).
- [36] R. Morales Suero, “Implementaci´on de algoritmo de detecci´on de objetos en entornos abiertos en hardware embebido,” Santiago de Querétaro, 2017. Accessed: Apr. 06, 2022. [Online]. Available: <https://cidesi.repositorioinstitucional.mx/jspui/bitstream/1024/307/1/ETM-RMS-2017.pdf>
- [37] P. Viola and M. Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features,” 2001.

- [38] “Face Detection using Haar Cascades — OpenCV 3.0.0-dev documentation.” [https://docs.opencv.org/3.0-beta/doc/py\\_tutorials/py\\_objdetect/py\\_face\\_detection/py\\_face\\_detection.html](https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html) (accessed Mar. 23, 2022).
- [39] A. Mittal, “Haar Cascades, Explained. A brief introduction into Haar... | by Aditya Mittal | Analytics Vidhya | Medium,” Dec. 21, 2020. <https://medium.com/analytics-vidhya/haar-cascades-explained-38210e57970d> (accessed Mar. 24, 2022).
- [40] J. Garcia Cuchillero, “Seguimiento de vuelo complejo. Análisis de algoritmos avanzados para una aplicación en Quiroptología,” Colmenarejo, Sep. 2019.
- [41] Rohith Gandhi, “R-CNN, Fast R-CNN, Faster R-CNN, YOLOv3 — Object Detection Algorithms | by Rohith Gandhi | Towards Data Science,” 2018. <https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e> (accessed Apr. 06, 2022).
- [42] “Búsqueda selectiva para la detección de objetos | R-CNN – Acervo Lima.” <https://es.acervolima.com/busqueda-selectiva-para-la-deteccion-de-objetos-r-cnn/> (accessed Apr. 06, 2022).
- [43] “Algoritmo You Only Look Once (YOLOv3) - Wikipedia, la enciclopedia libre,” Nov. 26, 2020. [https://es.wikipedia.org/wiki/Algoritmo\\_You\\_Only\\_Look\\_Once\\_\(YOLOv3\)#Entrenamiento](https://es.wikipedia.org/wiki/Algoritmo_You_Only_Look_Once_(YOLOv3)#Entrenamiento) (accessed Mar. 23, 2022).
- [44] enrique a., “Detección de objetos con YOLOv3: implementaciones y como usarlas | by enrique a. | Medium,” May 12, 2018. <https://medium.com/@enriqueav/detecci%C3%B3n-de-objetos-con-yolo-implementaciones-y-como-usarlas-c73ca2489246> (accessed Mar. 23, 2022).
- [45] S. Mallick, “Object Tracking using OpenCV (C++/Python),” Feb. 13, 2017. <https://learnopencv.com/object-tracking-using-opencv-cpp-python/> (accessed Mar. 28, 2022).
- [46] M. Høfft, “Portfolio - Median flow,” Mar. 02, 2014. <https://mortenhofft.github.io/medianflow/> (accessed Mar. 28, 2022).
- [47] J. Barkey Wolf, “Barkey Wolf Consulting - Object tracking in video using a MOSSE tracker implemented in Rust,” 2022. <https://barkeywolf.consulting/posts/mosse-tracker/> (accessed Mar. 28, 2022).
- [48] “Controlador PID - Control Automático - Picuino.” <https://www.picuino.com/es/control-pid.html> (accessed Mar. 23, 2022).
- [49] A. Camarillo, “¿Qué es un control PID? - 330ohms,” Jun. 02, 2021. <https://blog.330ohms.com/2021/06/02/que-es-un-control-pid/> (accessed Mar. 23, 2022).

- [50] “¿Qué son los PIDs? Explicación y guía de ajuste para Drones - FpvMax.” <https://www.fpvmax.com/uncategorized/pids-ajuste-drones/> (accessed Jun. 08, 2022).
- [51] “Cómo ajustar un controlador PID en Arduino.” <https://www.luisllamas.es/como-ajustar-un-controlador-pid-en-arduino/> (accessed Jun. 08, 2022).
- [52] “Diseño de sistemas de control. Sintonía de PIDs,” [https://ocw.ehu.eus/file.php/83/capitulo10\\_html/capitulo10.html](https://ocw.ehu.eus/file.php/83/capitulo10_html/capitulo10.html).
- [53] “Tello.” <https://www.ryzerobotics.com/es/tello/specs> (accessed May 20, 2022).
- [54] “PyCharm: el IDE de Python para desarrolladores profesionales, por JetBrains.” <https://www.jetbrains.com/es-es/pycharm/> (accessed Jun. 05, 2022).
- [55] “Open Images V6.” <https://storage.googleapis.com/openimages/web/index.html> (accessed Jun. 05, 2022).
- [56] “The AI Guy - YouTube.” <https://www.youtube.com/c/TheAIGuy> (accessed Jun. 05, 2022).
- [57] “YOLOv3: Real-Time Object Detection.” <https://pjreddie.com/darknet/yolo/> (accessed Jun. 05, 2022).
- [58] “Make your own Haar Cascade on Windows | Quick & Simple | Face Detection - YouTube.” <https://www.youtube.com/watch?v=Dg-4MoABv4I> (accessed Jun. 06, 2022).
- [59] “Salario para Ingeniero Informático Junior en España - Salario Medio.” <https://es.talent.com/salary?job=ingeniero+inform%C3%A1tico+junior> (accessed Mar. 15, 2022).
- [60] “Salario para Ingeniero Informático Senior en España - Salario Medio.” <https://es.talent.com/salary?job=ingeniero+inform%C3%A1tico+senior> (accessed Mar. 15, 2022).