UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

"Sistema de seguimiento de rostros en un dron cuadricóptero"

TESIS

Que para obtener el título de

Ingeniero Mecatrónico

P R E S E N T A

Yocoyani Ehecatzin Pérez Ayala

DIRECTOR DE TESIS

M.C. Gabriel Castillo Hernández

Ciudad Universitaria, Cd. Mx., 2022

Índice

LISTA DE IMÁGENES 4
LISTA DE TABLAS 5
LISTA DE DIAGRAMAS 5
INTRODUCCIÓN 6
PROBLEMÁTICA 6
OBJETIVOS 7
OBJETIVOS ADICIONALES 7
MARCO TEÓRICO 8
Características de un dron cuadricóptero 8
Controladores PID 10
Control en tiempo discreto 12
Visión por computadoras 14
Detección de rostros. 15
Identificación de rostros. 17
DESARROLLO DEL ALGORITMO 19
Análisis del sistema 19
Conexión dron computadora para la transmisión de datos por Wi-Fi 20
Implementación del algoritmo de identificación de imágenes 21
Secuencia de despegue y búsqueda de rostros inicial 22
Desarrollo del algoritmo de control PID 23
Control del movimiento *yaw* 23
Control del movimiento sobre el eje Z 28
Control del movimiento YAW y sobre el eje Z en combinación 33
Implementación del algoritmo de selección de objetivos de seguimiento 34
Algoritmo de reconocimiento de rostros objetivos 34
Algoritmo de seguimiento de objetivos en YAW 35
Algoritmo de seguimiento de objetivos en el eje Z 37
Algoritmo de seguimiento de objetivos en YAW y en el eje Z 38
RESULTADOS 39
Prueba de seguimiento sobre el eje YAW 39
Prueba de seguimiento sobre el eje vertical 39
Prueba de seguimiento sobre el eje *yaw* y el eje 40

Prueba de seguimiento aislando un objetivo 41
CONCLUSIONES 42
REFERENCIAS 44
BIBLIOGRAFÍA 45
ANEXOS. 46

LISTA DE IMÁGENES

Figura 1. Ejemplo de un dron cuadricóptero

Figura 2. Configuración Quad + de un dron

Figura 3. Configuración Quad *x* de un dron

Figura 4. Representación de los movimientos del dron en un espacio tridimensional

Figura 5. Esquema clásico de un controlador PID

Figura 6. Respuesta de un sistema con un control proporcional y diferentes ganancias

Figura 7. Respuesta de un sistema con un control PI y diferentes ganancias ki

Figura 8. Respuesta de un sistema con un control PID y diferentes ganancias k

Figura 9. Ejemplo de una función continua con su equivalente discreto y una función continua promedio con base en el equivalente discreto

Figura 10. Interacción de las ramas de la inteligencia Artificial en la visión por computadora

Figura 11. Características del algoritmo Haar

Figura 12. Puntos de referencia en rostros

Figura 13. Centrado de las medidas del rostro en una foto

Figura 14. Dron Tello EDU de DJ

Figura 15. Esquema de transmisión de datos por Wi-Fi

Figura 16. Coordenadas y medidas obtenidas en la detección de rostros

Figura 17. Comportamiento del sistema con la primera configuración de ganancias, YAW

Figura 18. Pérdida del objetivo con la primera configuración de ganancias, YAW

Figura 19. Señal de control con la primera configuración de ganancias, YAW

Figura 20. Comportamiento del sistema con la segunda configuración de ganancias, YAW.

Figura 21. Pérdida del objetivo con la segunda configuración de ganancias, YAW.

Figura 22. Señal de control con la segunda configuración de ganancias, YAW

Figura 23. Comportamiento del sistema con la tercera configuración de ganancias, YA

Figura 24. Seguimiento del objetivo con la tercera configuración de ganancias, YAW

Figura 25. Señal de control con la tercera configuración de ganancias, YA

Figura 26. Comportamiento del sistema con la primera configuración de ganancias, eje Z

Figura 27. Pérdida del objetivo con la primera configuración de ganancias, eje Z

Figura 28. Señal de control con la primera configuración de ganancias, eje Z
Figura 29. Comportamiento del sistema con la segunda configuración de ganancias, eje Z
Figura 30. Señal de control con la segunda configuración de ganancias, eje Z
Figura 31. Comportamiento del sistema con la tercera configuración de ganancias, eje
Figura 32. Seguimiento del objetivo con la tercera configuración de ganancias, eje Z
Figura 33. Señal de control con la tercera configuración de ganancias, eje Z
Figura 34. Esquema de control PID de YAW y desplazamiento en Z
Figura 35. Modelo físico simplificado de captura de fotos (horizontal)
Figura 36. Modelo geométrico de la captura de imagen (horizontal)
Figura 37. Modelo físico simplificado de captura de fotos (vertical)
Figura 38. Modelo geométrico de la captura de imagen (vertical)
Figura 39. Seguimiento de rostros sobre el eje YAW
Figura 40. Seguimiento de rostros sobre el eje Z
Figura 41. Seguimiento de objetivos controlando los dos movimientos sobre el eje Z

LISTA DE TABLAS

Tabla 1. Valores de calibración de PID para el primer método de Ziegler-Nichols
Tabla 2. Valores de calibración de PID para el segundo método de Ziegler-Nichols
Tabla 3. Tabla de comandos del SDK para el Dron Tello EDU

LISTA DE DIAGRAMAS

Diagrama 1. Diagrama de flujo del reconocimiento de imagen
Diagrama 2. Diagrama de flujo de la secuencia de despegue
Diagrama 3. Diagrama de flujo de la implementación del seguimiento PID eje YAW y eje Z
Diagrama 4. Diagrama de flujo del reconocimiento de un rostro objetivo
Diagrama 5. Diagrama de flujo del seguimiento de objetos en eje Z con ambos desplazamientos

INTRODUCCIÓN

En el presente trabajo se explica el desarrollo e implementación de un sistema de control para un dron que le permita realizar el seguimiento de rostros. El desarrollo del sistema de control se divide en dos partes, el seguimiento de rostros y el seguimiento de un rostro seleccionado como objetivo.

La solución se plantea mediante el uso de técnicas de control automático, tales como el desarrollo y calibración de un controlador PID. Además, se implementan algunos conceptos básicos del funcionamiento de la captura de fotografías para el desarrollo de un controlador que por limitaciones técnicas el control automático no pueda realizar.

También, se implementan técnicas relacionadas con la visión por computadora y *Deep Learning* para poder realizar el reconocimiento de rostros y selección de un rostro específico. Para esta implementación se utilizan metodologías y algoritmos desarrollados dentro de la biblioteca Open CV.

El trabajo plantea el uso de herramientas de desarrollo de fácil acceso e implantación para la solución y experimentación de temas relacionados con análisis de imágenes y el control de drones.

Al final se muestran los resultados del desarrollo e implementación logrados, de igual manera se plantean los posibles puntos de mejora que se pueden agregar al sistema de control mostrado.

PROBLEMÁTICA

Dentro del campo de la Inteligencia Artificial, las áreas de reconocimiento de imágenes y *Deep Learning* han avanzado en el desarrollo e implementación de algoritmos para la solución de problemas y creación de aplicaciones. Muchos de estos avances se han producido gracias al aumento de la potencia de procesamiento de las computadoras que permite integrar algoritmos complejos en equipos de cómputo con distintas características, diversificando el rango de aplicación de la Inteligencia Artificial.

En los últimos años la implementación de la Inteligencia Artificial se ha enfocado en resolver problemas de la vida cotidiana. Se pueden mencionar la creación de mejores sistemas de seguridad por reconocimiento de datos biométricos, la creación de aplicaciones que utilicen el análisis de imágenes para Realidad Virtual o Realidad Aumentada e incluso la implementación de *Machine Learning* y *Deep Learning* para crear vehículos autónomos.

Los avances en análisis y reconocimiento de imágenes han permitido que los vehículos autónomos sean de mayor utilidad para la sociedad en diversos aspectos. Debido a ello, en los últimos años los vehículos autónomos se han vuelto cada vez más relevantes en el transporte particular y privado, en la industria y en la investigación. Incluso el desarrollo de robots de rescate y exploración se benefician en gran medida de los avances de los vehículos autónomos.

En la década pasada los vehículos aéreos no tripulados (UAV por sus siglas en inglés), popularmente conocidos como drones, tuvieron un gran auge en su uso en diversos campos de aplicación. Estos dispositivos han sido utilizados para proyectos fotográficos o cinematográficos, en estudios de suelo para cultivos o minería e incluso se utilizan para el transporte de productos por vía aérea a locaciones de difícil acceso.

Es evidente la necesidad actual para cualquier sociedad el poder aplicar la Inteligencia Artificial y los drones para solucionar necesidades y problemáticas. Uno de los campos fundamentales de desarrollo en estas tecnologías es el seguimiento y reconocimiento de objetos en imágenes que se obtienen durante el vuelo de un dron.

OBJETIVOS

- Implementar un algoritmo que permita a un dron cuadricóptero seguir un rostro moviéndose sobre un eje rotacional y sobre un eje lineal.

- A partir de un banco de pruebas con rostros distintos, el dron podrá realizar el seguimiento de uno de los rostros seleccionados, diferenciando los rostros reconocidos en la imagen del objetivo seleccionado.

OBJETIVOS ADICIONALES

- Solución de los objetivos generales planteados mediante la implementación de tecnologías existentes que sean de fácil acceso y consulta. Esto incluye plataformas de desarrollo de código abierto, así como plataformas de desarrollo ofrecidas en el mercado para el público general.

MARCO TEÓRICO

Características de un dron cuadricóptero

Un dron cuadricóptero posee cuatro rotores distribuidos de manera que permitan controlar el movimiento del vehículo en el espacio, un ejemplo de cuadricóptero se muestra en la figura 1. Existen dos configuraciones principales para la operación de los cuadricópteros, la configuración "Quad +" y la configuración "Quad x".

Figura 1. Ejemplo de un dron cuadricóptero

La configuración Quad + tiene la característica de que el frente del vehículo está alineado con uno de los ejes en los que están montados los rotores, el segundo eje del dron estará a 90° del primero. Los motores que están sobre un mismo eje deben girar en igual sentido, ya sea en el de las manecillas del reloj (CW del inglés *Clockwise*) o en el contrario a las manecillas del reloj (CCW del inglés *Counterclockwise*), la condición que debe mantenerse es que el sentido de giro de los motores de un eje sea el contrario al sentido de giro de los motores del otro eje. En la figura 2. se muestra un esquema de esta configuración.

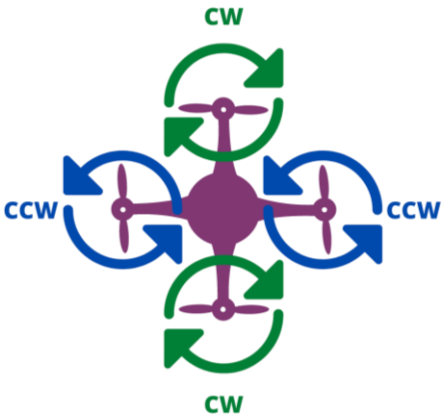

Figura 2. Configuración Quad + de un dron [1]

La configuración Quad *x* tiene la característica de tener sus ejes separados 45° con respecto al eje imaginario que atraviesa el lado del dron que sostiene la cámara del cuadricóptero. Con respecto a los rotores de la parte frontal, uno de ellos debe girar hacia adentro del dron, o en sentido horario (CW), y el otro rotor debe girar en hacia afuera del dron, o en sentido antihorario (CCW). Los rotores que están en el mismo eje deben girar en el mismo sentido para mantener equilibrada la rotación del dron. En la figura 3 se ejemplifica la configuración.

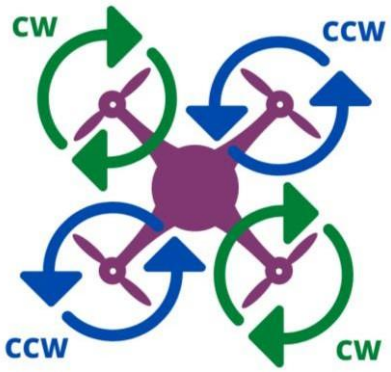

Figura 3. Configuración Quad *x* de un dron [1]

Debido a su configuración, un cuadricóptero tiene 4 grados de libertad mediante el cambio de orientación sobre los tres ejes del espacio cartesiano y de la elevación del dron. Con estas características el dron es capaz de desplazarse en un espacio cartesiano de 6 grados de libertad.

Basándonos en la definición tradicional de los ejes en el espacio cartesiano, al movimiento relacionado con la rotación sobre el eje vertical *z* del dron se le conoce como guiñada (*Yaw* en inglés) a la rotación sobre el eje *x* del dron se le conoce como balanceo (*Roll* en inglés) y a la rotación sobre el eje *y* se le llama cabeceo (*Pitch* en inglés), en el trabajo se utilizarán los términos en inglés de estos movimientos. En la figura 4 se muestra la configuración mencionada.

Figura 4. Representación de los movimientos del dron en un espacio tridimensional

Controladores PID

En la actualidad, la mayoría de los controladores implementados en la industria son controladores PID o controladores PID modificados. Su uso en la industria se ve justificado debido a que la aplicación de estos esquemas de control se puede realizar de manera experimental y sobre la marcha de los distintos procesos industriales. A su vez, presentan la facilidad de poderse implementar a través de distintas metodologías que no tienen la necesidad de utilizar el modelo matemático del sistema en el que se implementa, siendo una ventaja en la industria donde algunas veces no es práctico modelar un sistema y es conveniente medir únicamente las entradas y salidas de éste.

El esquema básico de los controladores PID se basa en tres partes esenciales: una parte proporcional, una integral y una diferencial. En conjunto las tres conforman la parte del controlador en lazo cerrado. El esquema de este modelo es el que se muestra en la figura 5.

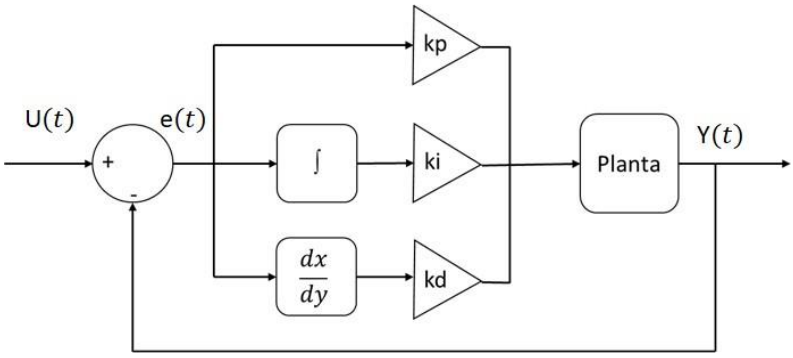

Figura 5. Esquema clásico de un controlador PID

Además de controlar el valor final del sistema, un controlador proporcional aumenta la señal de entrada a la planta cuando existe un error grande con respecto a la referencia y disminuye esa señal de control cuando el error va disminuyendo. La señal de salida de un sistema con un controlador proporcional se define como en la ecuación 1.

$$u(t) = k_p\, e(t) \quad \text{... (1)}$$

El principal problema de utilizar únicamente este controlador recae en que el sistema en estado permanente tendrá un error constante sin importar la función de transferencia del sistema. Además, al aumentar el valor de la constante proporcional el error se irá reduciendo, pero puede implicar que se generen señales de control de una magnitud mayor a la que el sistema puede generar. También al modificar la ganancia del controlador, dependiendo de la configuración de polos del sistema, puede provocar que el sistema se vuelva inestable. En la figura 6 muestra el comportamiento de un sistema controlado con un control proporcional con diferentes ganancias.

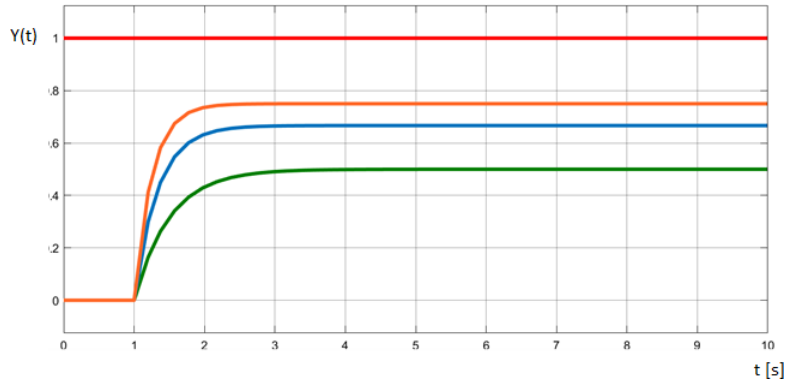

Figura 6. Respuesta de un sistema con un control proporcional y diferentes ganancias. Verde con ganancia igual a 1, azul con ganancia igual a 2 y naranja con ganancia igual a 3. Rojo señal de referencia.

El componente integral en el controlador PID se encarga de reducir a cero el error en estado permanente generado por el controlador proporcional. Para entender el funcionamiento del controlador es necesario observar su expresión matemática ecuación 2.

$$u(t) = k_i \int e(t)\, dt \quad \ldots (2)$$

Al analizar la expresión ecuación 2 observamos que el controlador se encarga de integrar los errores pasados del sistema por lo que realiza los ajustes necesarios con base en el pasado del sistema. Al juntar el control integral y el proporcional se obtiene un controlador PI con el que se pueden resolver muchas de las problemáticas de ingeniería. Con este controlador al aumentar la ganancia se reduce el tiempo del estado transitorio, aumenta el sobrepaso y aumenta la oscilación como se muestra en la figura 7.

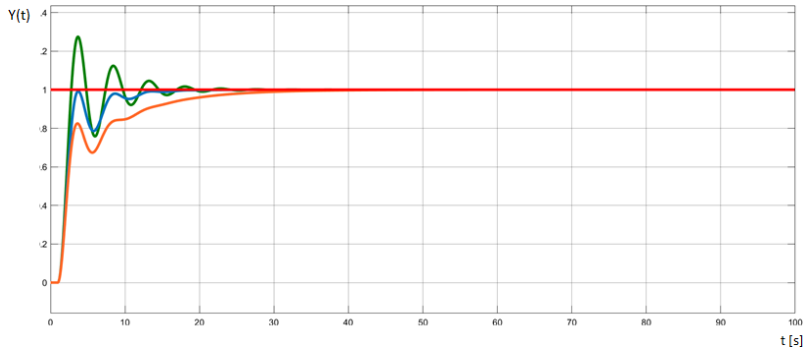

Figura 7. Respuesta de un sistema con un control PI y diferentes ganancias ki y una kp constante de 1. Verde con ganancia igual a 1, azul con ganancia igual a 0.5 y naranja con ganancia igual a 0.25. Rojo señal de referencia

La parte del derivador proporciona una mejor estabilidad en el sistema. Esto se debe a que el derivador permite por medio del concepto de la derivada tomar en cuenta el error futuro que tendrá el sistema como muestra su expresión ecuación 3.

$$u(t) = k_d \frac{d\, e(t)}{dt} \quad \ldots (3)$$

Al disminuir el valor de la constante del derivador se disminuye el sobrepaso y la oscilación de la salida del sistema como se muestra en la figura 8.

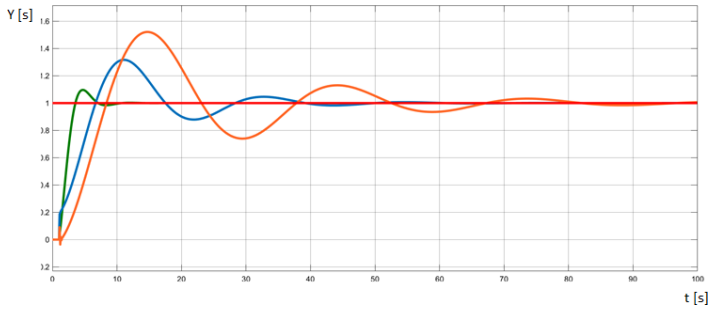

Figura 8. Respuesta de un sistema con un control PID y diferentes ganancias kd, con kp y kd constantes e iguales a 1. Verde con ganancia igual a 1, azul con ganancia igual a 10 y naranja con ganancia igual a 2. Rojo señal de referencia (salida en unidades adimensionales).

Para el diseño de un controlador PID existen varios métodos que permiten obtener de manera experimental los parámetros necesarios para la implementación del controlador.

Control en tiempo discreto

La teoría de control en tiempo continuo suele aplicarse cuando se utilizan dispositivos analógicos para el desarrollo del controlador. Sin embargo, en la actualidad es más común la implementación de controladores en dispositivos digitales debido a las facilidades que ofrecen.

La diferencia entre la implementación analógica y digital radica en el uso de convertidores ADC, siglas del inglés *Analogic to Digital Converter* (Convertidores analógico a digital) para la medición de la señal de salida de la planta y los convertidores DAC, siglas del inglés *Digital to Analogic Converter* (Convertidores digital a analógico). La conversión de las señales se lleva a cabo de manera constante durante periodos de tiempo definidos, al periodo de tiempo que toma analizar cada muestra se le conoce como periodo de muestreo.

Debido a que la salida y entrada del sistema se toman con base en los periodos de muestreo se obtiene una función discreta que depende de un término kT (donde T es el periodo de muestreo y k es un número entero que indica el paso actual), de igual manera la función de error se verá afectada por este término por lo que será una función discreta. Por la discretización del sistema, en lugar de utilizar una ecuación diferencial para su representación se utilizará una ecuación en diferencias como una aproximación al comportamiento del sistema dinámico continuo.

El impacto más significativo asociado con la implementación de un sistema de control digital es el *delay* asociado con el tiempo de mantenimiento del valor de la señal obtenida durante cada muestreo. El *delay* en sistemas con realimentación afecta la estabilidad y el factor de amortiguamiento del sistema.

Como el valor de la salida del sistema permanece constante hasta que se vuelva a tomar una nueva lectura la función de la salida queda dividida en una serie de pasos como se muestra en la figura 9.

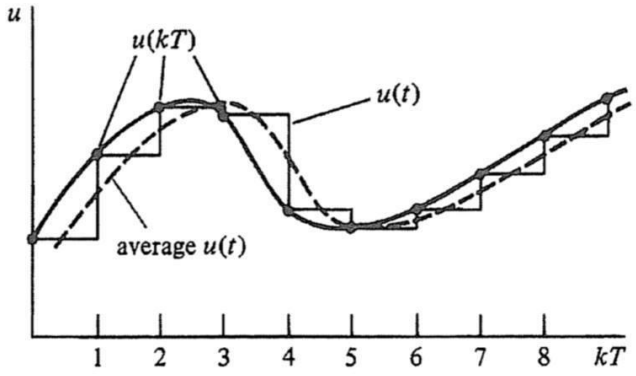

Figura 9. Ejemplo de una función continua con su equivalente discreto y una función continua promedio con base en el equivalente discreto [4]

Una de las formas más simples para realizar la discretización de una función continua es el método de Euler. El método parte de la definición de la derivada mostrada en la ecuación 4.

$$\dot{x} = \lim_{\delta t \to 0} \frac{\delta x}{\delta t} \ldots (4)$$

Mediante la ecuación 5 se puede obtener una buena aproximación. Entre más cercano sea el valor del periodo (T) a 0, mejor será la aproximación.

$$\dot{x}(k) \approx \frac{x(k)-x(k-1)}{T} \ldots (5)$$

Esta aproximación puede ser utilizada para sustituir las derivadas que aparezcan en una ecuación diferencial de un controlador, ya que al ser una ecuación algebraica recursiva puede ser resuelta fácilmente por sistemas computacionales. Esta aproximación posee buenos resultados en sistemas que tienen un ancho de banda del orden de 100 Hz [4].

De manera empírica, para que un sistema opere correctamente el rango de muestreo debe ser al menos 30 veces más rápido que el ancho de banda con el fin de que exista una relación aceptable entre la función continua y el sistema discreto.

Con base en el método de Euler para la discretización de funciones continuas se obtienen los siguientes equivalentes discretos para un controlador proporcional (ecuación 6), para un controlador integral (ecuación 7) y un controlador derivativo (ecuación 8).

$$u(k) = k_p\, e(k) \ldots (6)$$

$$u(k) = u(k-1) + k_I\, T\, e(k) \ldots (7)$$

$$u(k) = \frac{k_D}{T}\left[e(k) - e(k-1)\right] \ldots (8)$$

Visión por computadoras

Durante los últimos años la Inteligencia Artificial ha provocado avances en distintas áreas de la ciencia y la tecnología. Una de las ramas que ha tenido un gran desarrollo es el *Machine Learning*. *El Machine Learning* se encarga de crear metodologías y algoritmos para que los sistemas computacionales puedan llevar a cabo procesos de identificación de patrones y toma de decisiones mediante el aprendizaje adquirido a través de una serie de pruebas. Con cada prueba se espera que los modelos vayan mejorando su eficiencia conforme adquieran más información.

La visión por computadora es una rama de la Inteligencia Artificial y de *Machine Learning* que busca que las computadoras puedan entender el contenido de imágenes de una manera similar a la que el ser humano reconoce características dentro de imágenes. Para realizar está tarea existen una cantidad diversa de metodologías y algoritmos que permiten con mayor o menor eficiencia reconocer información específica dentro de las imágenes,

Otra rama de la Inteligencia Artificial y *Machine Learning* es el *Deep Learning*, que por medio de redes neuronales es capaz de producir modelos que reconozcan patrones específicos en imágenes, videos y audios, con un alto grado de efectividad. Se observa que una parte de las aplicaciones del *Deep Learning* guarda una estrecha relación con la visión por computadora.

A continuación, se detalla un esquema (figura 10) de la relación entre las áreas que interactúan en la visión por computadora.

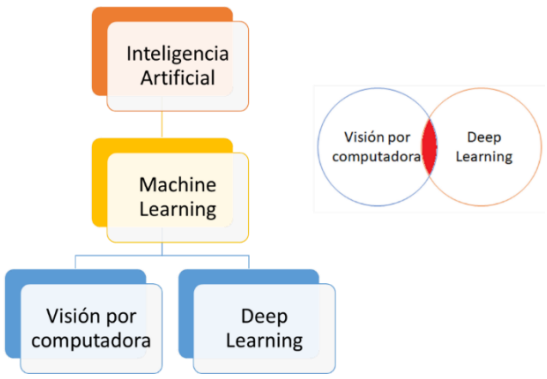

Figura 10. Interacción de las ramas de la inteligencia Artificial en la visión por computadora

Una de las ramas principales de la visión por computadora tiene la tarea de identificar rostros humanos mediante métodos de visión por computadora y *Deep Learning*.

Detección de rostros.

Un método comúnmente utilizado en el reconocimiento de imágenes es el algoritmo Haar. El algoritmo Haar fue propuesto por el matemático Alfréd Haar en el año de 1909 y fue modificado para su implementación en reconocimiento de imágenes por Paul Viola y Michael Jones en 2001 [5]. Antes de utilizar el algoritmo se debe realizar el entrenamiento de un modelo utilizando imágenes con rostros (positivas) e imágenes sin rostros (negativas). Cabe señalar que las imágenes con las que trabaja el algoritmo Haar deben estar en escala de grises para que puedan ser analizadas de manera correcta.

El algoritmo Haar utiliza 4 características o filtros básicos: el borde vertical, el borde horizontal, la línea vertical y la línea horizontal (figura 11). Dentro de la imágen por analizar se llevan a cabo una serie de barridos por secciones con un tamaño definido y por cada una de las características, y a su vez este proceso se repite variando el tamaño de las secciones que se analizan.

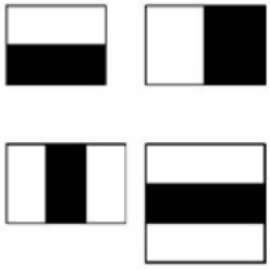

Figura 11. Características del algoritmo Haar. Las primeras dos imágenes representan los bordes vertical y horizontal y los segundos patrones representan las líneas vertical y horizontal [5]

Para empezar el proceso de análisis a los píxeles totalmente negros se les asigna el valor de 1, a los píxeles totalmente blancos se les asigna un valor de 0 y a los pixeles grises se les asigna un valor entre 0 y 1 dependiendo de su brillo. Cada vez que se evalúa una sección con una característica se suman los valores de los píxeles que existen en la imagen en la región correspondiente a la franja blanca de la característica superpuesta y también se suman los valores de los píxeles que existen en la imagen en la región correspondiente a la franja negra de la característica superpuesta. Posteriormente se procede a calcular el valor Δ con el valor promedio de la región de la imagen correspondiente a la franja negra de la característica y el valor promedio de la región que se corresponde a la franja blanca (ecuación 9). Este valor de Δ al acercarse a 1 indica que probablemente el sector analizado de la imagen corresponda con el patrón de la característica que se esté utilizando.

$$\Delta = negros - blancos = \frac{1}{n}\sum_{0}^{n_{negros}} I(x) - \frac{1}{n}\sum_{0}^{nblancos} I(x) \ldots (9)$$

En cada barrido de la imágen la característica con el mejor valor (cercano a 1) se utilizará como discriminante en la detección de rostros. A la mejor característica seleccionada de

manera individual se le conoce como característica débil, debido a que por sí sola no puede ser capaz de detectar si una imagen contiene o no un rostro; para que el algoritmo tenga un mejor grado de precisión es necesario juntar todas las características reconocidas en la imágen.

El proceso de selección de características se repite con cada una de las imágenes seleccionadas para entrenar el modelo y con todas las características recolectadas se crea el modelo para determinar si una imagen tiene o no un rostro. El trabajo *"Rapid Object Detection using a Boosted Cascade of Simple Features"* indica que con un aproximado de 200 características se puede obtener una precisión del 95% en la identificación de rostros [4].

Una vez completado el modelo se procede a su implementación para el reconocimiento de rostros. En la imagen que se desea encontrar un rostro se buscan una a una las características del algoritmo Haar registradas en el modelo. En promedio, cada imagen tiene alrededor de 6000 características por lo que demandaría una gran cantidad de procesamiento computacional. Así que para disminuir la cantidad de procesamiento se implementa el concepto de Clasificador de Cascadas (*Cascade Classifier*).

El Clasificador de Cascadas consiste en agrupar las características del modelo en distintas etapas. Cada etapa contendrá características específicas que discriminaran si la imagen analizada posee o no las características esperadas en un rostro. En caso de que la imagen cumpla con las características de un rostro se procede a ejecutar la siguiente etapa de análisis. Al manejar un limitado número de características por etapa, el clasificador permite que el procesamiento sea menor, ya que si las características buscadas en la etapa no se encuentran la imagen se descarta automáticamente.

Identificación de rostros.

Una vez que se tiene un rostro detectado se puede utilizar un algoritmo de identificación por medio de puntos de referencia en rostros. Con base en el método propuesto por Vahid Kazemi y Josephine Sullivan, se proponen 68 puntos de referencia aplicables a cualquier rostro [6], figura 12.

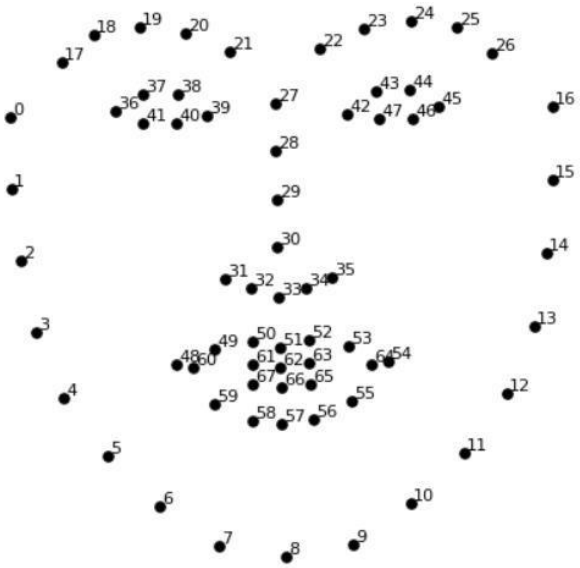

Figura 12. Puntos de referencia en rostros [6]

Utilizando los puntos de referencia se procede a implementar un modelo de *Machine Learning* capaz de identificar estos 68 puntos en cualquier rostro presentado.

Una vez obtenidas las ubicaciones de los puntos de referencia se procede a ajustar el rostro para que se ajuste de manera frontal. Estas acciones se llevan a cabo únicamente mediante transformaciones de rotación y escalamiento de manera que se preserven los paralelismos del rostro (figura 13).

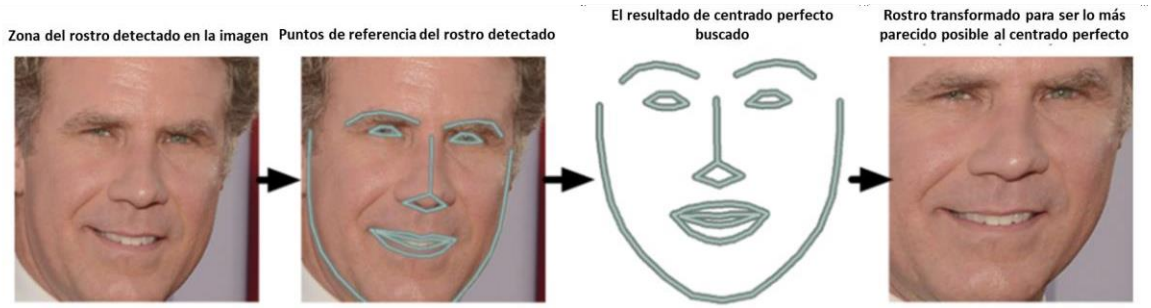

Figura 13. Centrado de las medidas del rostro en una foto [6]

Se deben realizar medidas entre los puntos de referencia para caracterizar al rostro. Posteriormente deberían compararse las medidas realizadas para determinar la similitud de los rostros. Sin embargo, debe definirse qué medidas son las mejores para realizar esa comparación. Para realizar la selección de medidas se utiliza una Red Neuronal Convolucional Profunda. Para entrenar está red se proporciona una imagen base de la que se extraen 128 medidas del rostro. Después se utiliza una imagen del rostro correcto y otra del incorrecto, obteniendo las 128 medidas correspondientes de cada una. Se procede a que la red neuronal identifique el rostro correcto con las medidas que considere relevantes. Una vez terminado el primer ciclo se repite el entrenamiento hasta que la red aprende a tomar las medidas correctas para identificar los rostros.

Con las 128 medidas seleccionadas por la red neuronal se genera un archivo embebido con la información recopilada. Este método de extracción de datos fue creado por Google [6].

Utilizando una tarjeta de video NVidia Tesla para obtener una red entrenada con un buen porcentaje de precisión puede llegar a tardar hasta 24 horas. Por ello, suelen utilizarse redes generadas con anterioridad y que se comparten en un repositorio en la nube, como el creado por Brandon Amos [6].

DESARROLLO DEL ALGORITMO

Análisis del sistema

Para llevar a cabo la solución del problema se utilizó el modelo Tello EDU de DJI (figura 14). Se decidió trabajar con este dron debido a que posee las siguientes características:

- Conexión Wi-Fi por medio del puerto UDP 8889.
- Biblioteca para control del dron por medio de Python.
- Transmisión de imagen por medio de la comunicación Wi-Fi.
- Cámara con resolución de 5 megapíxeles y grabación de vídeo HD.

Figura 14. Dron Tello EDU de DJI

El dron posee una serie de comandos predefinidos por el fabricante con los cuales podemos controlar el movimiento del dron y recibir imágenes por medio de la conexión Wi-Fi. Los comandos más relevantes son los mostrados en la tabla 3.

Comando	Descripción
take off	Orden de despegue del dron.
land	Orden de aterrizaje del dron.
stream on	Inicio de la transmisión de imagen.
up *x*	El dron sube la altura *x* en cm.
down *x*	El dron baja la altura *x* en cm.
cw *x*	El dron gira en sentido horario *x* grados.
cww *x*	El dron gira en sentido antihorario *x* grados.
rc *a b c d*	Se modifican las velocidades de: ● Movimiento izquierda/derecha (a) ● Movimiento atrás/adelante (b) ● Movimiento arriba/abajo (c) ● Movimiento *yaw* (d)
battery?	Regresa el porcentaje de batería del dron.

Tabla 3. Comandos más importantes del SDK para el Dron Tello EDU [7]

Es importante señalar que la comunicación con el dron se realiza por medio de un SDK proporcionado por Ryzen Robotics de Tello [7]. Por lo que no se tiene control del sistema

completo del dron, y únicamente se pueden controlar los movimientos explicados en la tabla de comandos.

Conexión dron computadora para la transmisión de datos por Wi-Fi

Como se mencionó, el dron está precargado con el software del Ryzen Robotics para controlar el movimiento y únicamente se tiene acceso a utilizar los comandos proporcionados por el SDK, sin embargo, el manejo de datos obtenido por la transmisión de imagen y el algoritmo de seguimiento se deben desarrollar de manera remota mediante un equipo que ejecute el software a desarrollar.

Para controlar los movimientos e información que corresponden al funcionamiento del dron se debe utilizar una conexión Wi-Fi por medio del puerto UDP 8889.

Las características de la conexión Wi-Fi son las siguientes:

- Los estándares más comunes IEEE 802.11b, IEEE 802.11g y IEEE 802.11n permiten tazas de transferencia de información de 11 $\frac{Mb}{s}$, 54 $\frac{Mb}{s}$ y 300 $\frac{Mb}{s}$ respectivamente.
- La mayoría de las conexiones Wi-Fi actuales permiten la transferencia de información a frecuencias de 5 GHz y 2.4 GHz.
- La mayoría de los dispositivos tecnológicos actuales son compatibles con esta conexión.

Es importante mencionar que el equipo utilizado es compatible con el estándar IEEE 802.11ac que permite frecuencias de transferencia de información de hasta 5 GHz.

Para realizar la conexión por Wi-Fi con el dron basta con presionar su botón de encendido y esperar a que este aparezca dentro de la zona de recepción del punto de acceso Wi-Fi de la computadora que se usará para analizar la información.

Una vez establecida la conexión Wi-Fi se ejecuta el programa elaborado en Python, ya que por default la biblioteca para usar el dron Tello incluye la función de apertura y conexión con el dron por el puerto UPD correspondiente. Es importante señalar que se debe recurrir al cierre de la conexión dentro del código siempre que se terminen las pruebas, ya que el no llevar a cabo esta operación provocará un bloqueo del puerto UDP que tendrá que ser reiniciado de manera manual desde el sistema.

Tomando en cuenta la conexión descrita el diagrama de conexión y transmisión de datos entre el dron y computadora sería el mostrado en la figura 15.

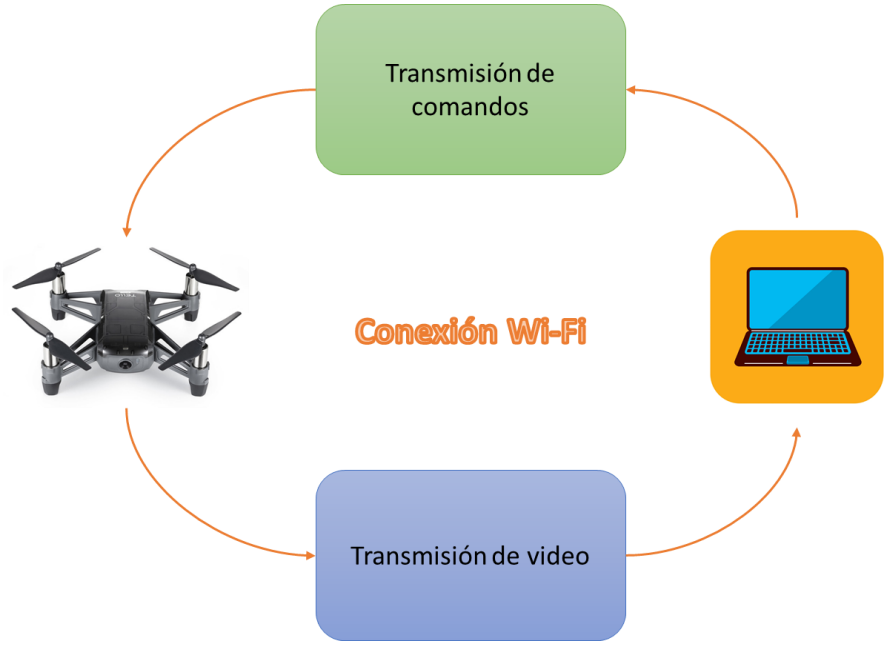

Figura 15. Esquema de transmisión de datos por Wi-Fi

Implementación del algoritmo de identificación de imágenes

Para el reconocimiento de rostros se implementó el algoritmo conocido como Haar Cascade utilizando la biblioteca de OpenCV en Python [5]. De acuerdo con la documentación ofrecida por la biblioteca el proceso para la detección de rostros es el siguiente.

- Cargar al clasificador de cascada el modelo con la clasificación Haar.
- Convertir la imagen a analizar en escala de grises para poder utilizar el procedimiento del algoritmo.
- Utilizar la función *detectMultiScale* de OpenCV para detectar los objetos que coinciden en la imagen con el modelo cargado. La función regresará las coordenadas donde se encuentra la imagen, coordenadas de la esquina superior izquierda donde la biblioteca define el área donde se encuentra el rostro, y el tamaño en pixeles de la imagen. La figura 16 muestra el sistema de coordenadas dentro de la foto y señala las principales coordenadas y medidas que regresa la función.

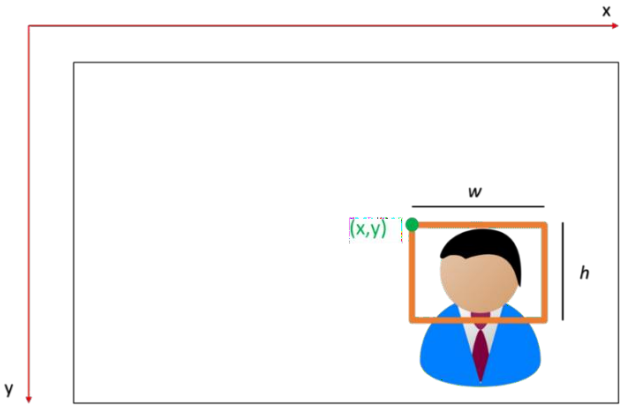

Figura 16. Coordenadas y medidas obtenidas en la detección de rostros

- Trazar un rectángulo sobre la imagen original en cada una de las caras detectadas utilizando las coordenadas obtenidas y el tamaño de la imagen.
- Con las coordenadas y el tamaño se guarda el centro y área de la imagen (ecuación 10 y ecuación 11), el centro servirá para el posterior seguimiento de imagen.

$$(cx, cy) = (x + \frac{w}{2}, y + \frac{h}{2}) \dots (10)$$

$$Área = w\ h \dots (11)$$

Una vez almacenados los datos de cada uno de los posibles rostros se seleccionó el rostro identificado cuya área haya sido la mayor, esto debido a que el rostro con mayor área es, probablemente, el rostro más cercano. Esta suposición se realiza debido a que la fotografía que analizamos es una proyección en dos dimensiones de un espacio tridimensional.

Cabe señalar que para el posterior seguimiento del rostro por parte del dron se tratará de alinear el centro del área de la imagen con el rostro detectado y centro de la imagen capturada por la cámara del dron.

En el diagrama 1 del anexo se observa el flujo de la implementación del algoritmo de reconocimiento de imagen.

Secuencia de despegue y búsqueda de rostros inicial

Como primer paso en la secuencia de ejecución del programa se planificó una secuencia de despegue en el que el dron fuese capaz de detectar rostros durante su ascenso a la altura operativa de 2 m.

La secuencia de despegue se planificó de la siguiente manera de la siguiente manera:

- Iniciar la transmisión de imagen entre el dron y la computadora.
- Ejecutar el comando de despegue del dron, elevándose 50 cm del suelo.
- Ejecutar la función de reconocimiento de rostros.

- Si un rostro es detectado en el campo de visión del dron, el programa pasa a la etapa de seguimiento de rostros mediante el control PID.
- En caso de no encontrar un rostro el dron se elevará 50 cm y repetirá la función de detección hasta detectar un rostro o alcanzar la altura máxima de operación.
- Si se alcanza la altura máxima de operación y no se ha detectado un rostro, el dron girará 10° en sentido horario e iniciará la detección de rostros. Este proceso se repetirá hasta que se encuentre un rostro o se haya completado una vuelta de 360°.
- En caso de no encontrar un rostro el dron permanecerá en vuelo por 15 segundos y después realizará un aterrizaje automático.

En el diagrama 2 del anexo se ejemplifica el funcionamiento con un diagrama de flujo.

Desarrollo del algoritmo de control PID

Control del movimiento *yaw*

Para desarrollar el algoritmo de control PID del sistema en el movimiento *yaw* se definió como variable a controlar la posición angular del dron y como variable controlada la velocidad *yaw* del dron. Debido a las limitaciones de control de actuadores por el SDK no fue posible seleccionar la velocidad de cada uno de los actuadores como variable controlada.

Teniendo en cuenta que se planea realizar el seguimiento del rostro centrando la imagen de la cámara con la cara detectada, la implementación por código del PID el algoritmo fue el siguiente:

- Por medio de las coordenadas recibidas de la identificación de imagen se lleva a cabo el cálculo del error en píxeles entre el centro de la imagen tomada y el centro del rostro detectado. En caso de no existir ningún rostro en la imagen, el control PID no realiza un cambio en la posición del dron.
- Se calcula el valor de la señal de control (velocidad *yaw*) por medio de la ecuación 12. En la ecuación se utiliza el error pasado para la parte diferencial ($error_p$) y el error acumulado para la parte integral ($error_{ac}$).

$$Velocidad_{YAW} = error * k_p + (error + error_{ac}) * k_i + (error - error_p) * k_d \ldots (12)$$

Para diseñar el sistema de control PID del dron se analizó la posibilidad de implementar los métodos de Ziegler-Nichols, pero se concluyó que no sería viable su uso. Para tener un método de partida se utilizó la configuración planteada por "*Murtaza's workshop – Robotics and AI*" [8], en donde se proponen como valores de calibración de PID iniciales:

- $k_p = 0.5$
- $k_i = 0.0$
- $k_d = 0.5$

Sin embargo, al realizar las pruebas correspondientes con estos valores de calibración se identificaron los siguientes problemas:

- Un gran tiempo de asentamiento en el posicionamiento del dron.
- Un sobrepaso que provoca que el dron pierda de su campo de visión el rostro detectado.
- Inestabilidad inherente a la implementación de un controlador PID con un valor uno en el componente integral.

En la figura 17 se muestra la gráfica correspondiente al posicionamiento del centro de la imagen objetivo sobre el eje x de la imagen. Se aprecia que durante los primeros 4 segundos el dron oscila sobre el eje correspondiente al movimiento *yaw*, posteriormente la posición se queda en un valor constante de 0 debido a la pérdida del objetivo de seguimiento (figura 18).

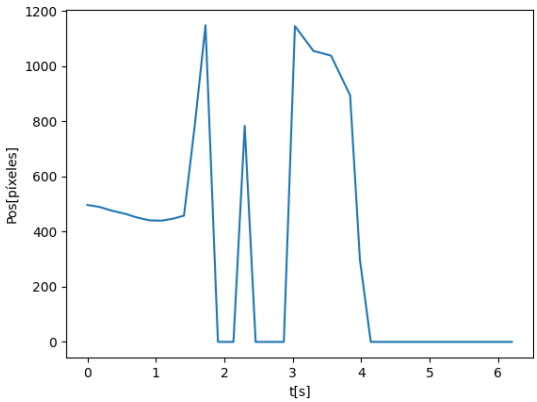

Figura 17. Comportamiento del sistema con la primera configuración de ganancias, *yaw*

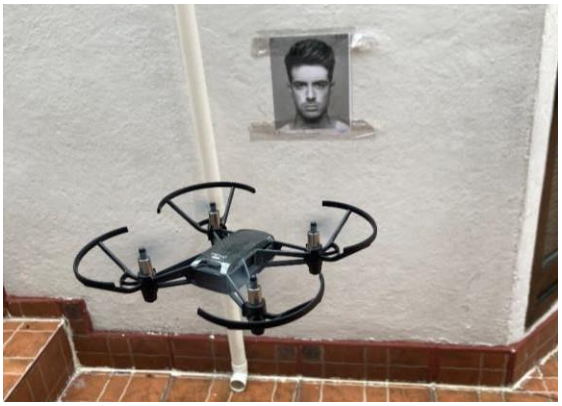

Figura 18. Pérdida del objetivo con la primera configuración de ganancias, *yaw*

En la figura 19 se muestra el comportamiento de la señal de control del sistema, que en este caso es la velocidad de giro sobre el eje correspondiente al movimiento *yaw* del dron. Inicialmente oscila entre valores negativos y positivos (valores negativos indican giro en sentido contrario a las manecillas del reloj y los valores positivos indican giro en sentido de las manecillas del reloj), lo que indica que hay un cambio constante en el sentido de giro.

Después del segundo 4 el dron se mantiene en una velocidad negativa que corresponde con la pérdida de objetivo en la posición.

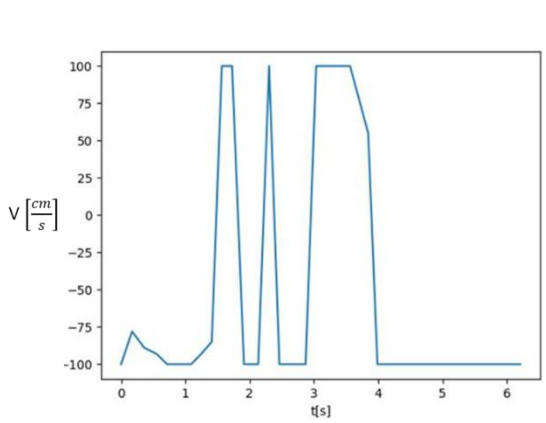

Figura 19. Señal de control con la primera configuración de ganancias, *yaw*

Mediante la realización de pruebas se ajustó de manera empírica el controlador y se llegó a la propuesta de los siguientes valores para el controlador PID.

- $k_p = 0.1$
- $k_i = 0.1$
- $k_d = 0.5$

Los valores propuestos se escogieron con base en la siguiente lógica:

- El valor de la ganancia k_i se escogió con un valor distinto de 0 para que se pudiera agregar un polo al sistema que permitiera estabilizarlo. Sin embargo, se propuso un valor menor al de las otras constantes, ya que un valor extremadamente grande de la ganancia podría provocar que la señal de control fuese sumamente grande y el sistema podría presentar errores.
- Se redujo el valor de la ganancia k_p para reducir el sobrepaso de la respuesta del dron y evitar la pérdida de la imagen de referencia.

Con las nuevas ganancias propuestas se observó un mejor desempeño en los siguientes aspectos.

- Menor sobrepaso en la respuesta del sistema.
- Menor tiempo de asentamiento en la respuesta del sistema.

A pesar de solucionar algunos de los problemas de la primera configuración se observó que el problema relacionado con el sobrepaso persistía y el dron seguía perdiendo la referencia de la imagen.

En la figura 20 se muestra la gráfica correspondiente al posicionamiento del centro de la imagen objetivo sobre el eje x de la imagen. Se aprecia que durante los primeros 8 segundos el dron oscila sobre el eje correspondiente al movimiento *yaw* con una frecuencia mayor que

en la primera configuración, posteriormente la posición se queda en un valor constante de 0 debido a la pérdida del objetivo de seguimiento (figura 21).

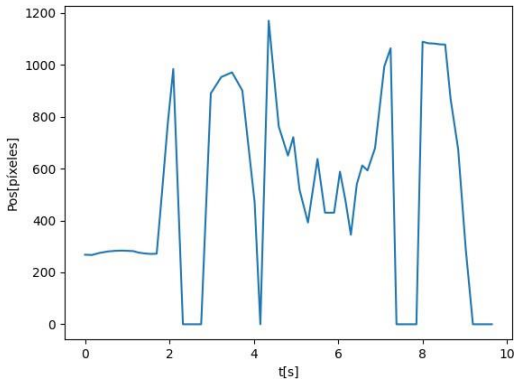

Figura 20. Comportamiento del sistema con la segunda configuración de ganancias, *yaw*

Figura 21. Pérdida del objetivo con la segunda configuración de ganancias, *yaw*

En la figura 22 se muestra el comportamiento de la señal de control del sistema, que en este caso es la velocidad de giro sobre el eje correspondiente al movimiento *yaw* del dron. Inicialmente oscila entre valores negativos y positivos, lo que indica que hay un cambio constante en el sentido de giro. Además, la frecuencia de esta señal es mayor en comparación con la primera configuración. Después del segundo 8 el dron se mantiene en una velocidad negativa que corresponde con la pérdida de objetivo en la posición.

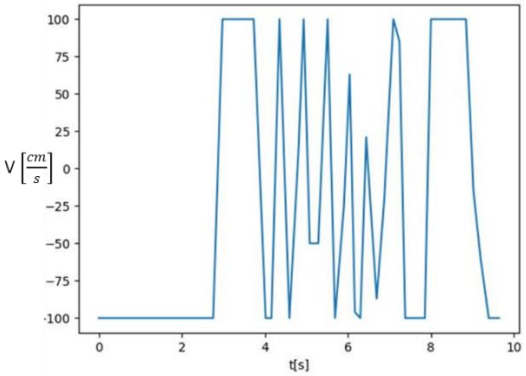

Figura 22. Señal de control con la segunda configuración de ganancias, YAW

Para solucionar los problemas del controlador anterior realizaron más pruebas ajustando los valores de calibración, al final se llegó a los siguientes valores para el controlador PID.

- $k_p = 0.1$
- $k_i = 0.001$
- $k_d = 0.3$

Los valores propuestos se escogieron con base en la siguiente lógica:

- El valor de la ganancia k_i se redujo para disminuir el sobrepaso y la señal de control del sistema.
- Se redujo el valor de la ganancia k_d para reducir el sobrepaso de la respuesta del dron y también para disminuir las oscilaciones del sistema. Está modificación implica el aumento del tiempo en estado transitorio.

Con las nuevas ganancias propuestas se observó un mejor desempeño en los siguientes aspectos.

- Un sobrepaso menor que conlleva a que el sistema no pierda la referencia al ejecutar su movimiento.
- A pesar del aumento del estado transitorio, se obtuvo una respuesta suficientemente rápida que permite seguir objetos en movimiento.

En la figura 23 se muestra la gráfica correspondiente al posicionamiento del centro de la imagen objetivo sobre el eje x de la imagen. Con esta configuración de valores PID se observa que tenemos un comportamiento amortiguado y que es capaz de seguir el objetivo propuesto. Como datos resaltables tenemos un tiempo de asentamiento aproximado de 5 segundos y un sobrepaso que evita la pérdida de la detección del objetivo. El enfoque del objetivo se muestra en la figura 24.

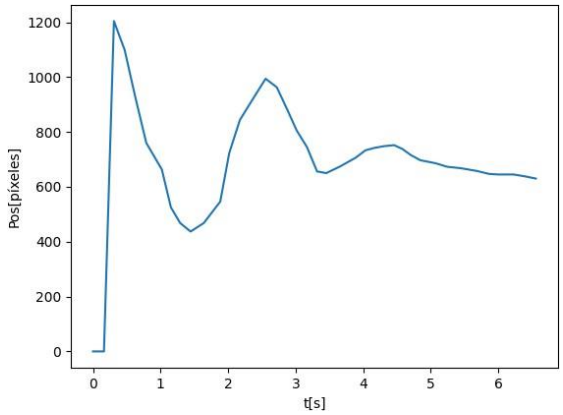

Figura 23. Comportamiento del sistema con la tercera configuración de ganancias, yaw

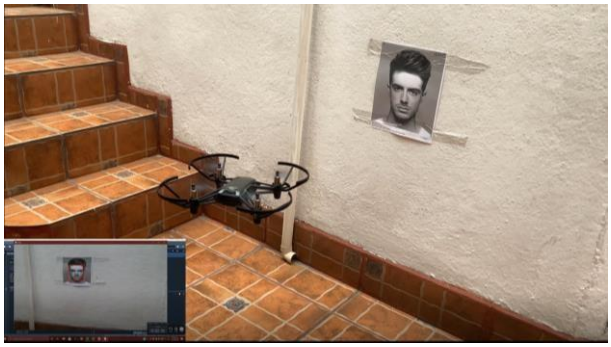

Figura 24. Seguimiento del objetivo con la tercera configuración de ganancias, yaw

En la figura 25 se muestra el comportamiento de la señal de control del sistema, que en este caso es la velocidad de giro sobre el eje del movimiento *yaw* del dron. Es resaltable señalar que en los momentos que se sobrepasa la posición de referencia la velocidad sufre un cambio de signo, implicando un cambio de sentido de giro, también se observa una reducción en la oscilación de la señal conforme se acerca al valor de referencia.

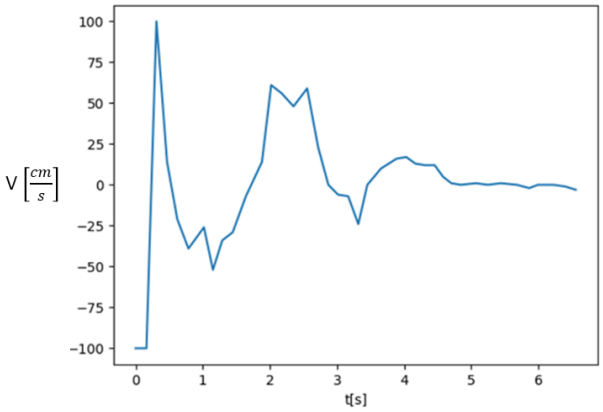

Figura 25. Señal de control con la tercera configuración de ganancias, YAW

Control del movimiento sobre el eje Z

Para desarrollar el algoritmo de control PID del sistema en el movimiento sobre el eje Z se definió como variable a controlar la posición del dron sobre el eje mencionado y como variable controlada la velocidad *UP/DOWN* del dron. Debido a las limitaciones de control de actuadores por el SDK no fue posible seleccionar la velocidad de cada uno de los actuadores como variable controlada.

Teniendo en cuenta que se planea realizar el seguimiento del rostro centrando la cámara con la cara detectada, la implementación por código del PID el algoritmo fue el siguiente:

- Por medio de las coordenadas recibidas de la identificación de imagen se lleva a cabo el cálculo del error en píxeles entre el centro de la imagen tomada y el centro del

rostro detectado. En caso de no existir ningún rostro en la imagen, el control PID no realiza un cambio en la posición del dron.

- Se calcula el valor de la señal de control (velocidad UP/DOWN) por medio de la siguiente expresión (ecuación 13). En la ecuación se utiliza el error pasado para la parte diferencial ($error_p$) y el error acumulado para la parte integral ($error_{ac}$).

$$Velocidad_{UP/DOWN} = error * k_p + (error + error_{ac}) * k_i + (error - error_p) * k_d \ldots (13)$$

Para la calibración del algoritmo PID se optó por empezar con la configuración final obtenida en la última configuración del movimiento *yaw*, siendo los valores de calibración de PID iniciales:

- $k_p = 0.1$
- $k_i = 0.001$
- $k_d = 0.3$

Sin embargo, al realizar las pruebas correspondientes con estos valores de calibración se identificaron los siguientes problemas:

- Un sobrepaso que provoca que el dron pierda de su campo de visión el rostro detectado.

En la figura 26 se muestra la gráfica correspondiente al posicionamiento del centro de la imagen objetivo sobre el eje *y* de la imagen. Se observa que la posición empieza a subir y llega a un punto donde el que el dron pierde el seguimiento del objetivo. La pérdida del objetivo se observa en la figura 27.

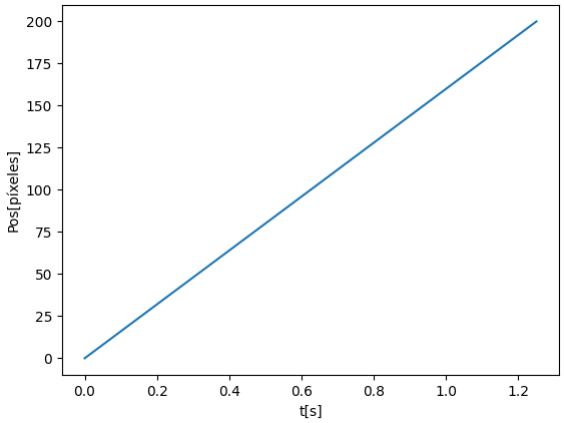

Figura 26. Comportamiento del sistema con la primera configuración de ganancias, eje Z

Figura 27. Pérdida del objetivo con la primera configuración de ganancias, eje Z

En la figura 28 se muestra el comportamiento de la señal de control del sistema, que en este caso es la velocidad de desplazamiento del dron sobre el eje Z. Se observa como en el primer segundo la señal de control disminuye desde el valor máximo de 100 $\left[\frac{cm}{s}\right]$ indicando que el dron inmediatamente recibe un impulso para subir rápidamente, pero al sobrepasar la referencia se reduce hasta 60 $\left[\frac{cm}{s}\right]$ su velocidad, sin embargo, esta disminución de la velocidad no es suficiente para lograr que el dron no pierda el objetivo de seguimiento.

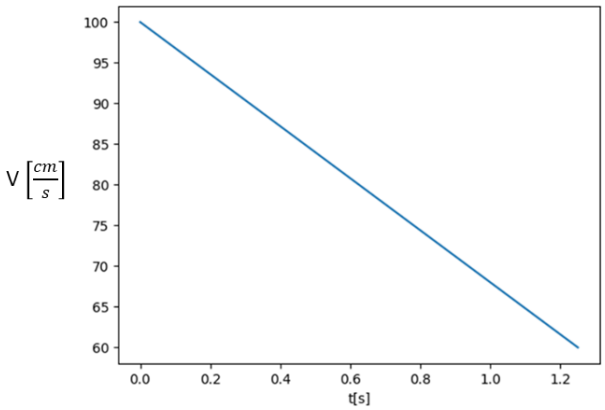

Figura 28. Señal de control con la primera configuración de ganancias, eje Z

Para solucionar los problemas del controlador anterior se propusieron los siguientes valores para el controlador PID después de realizar distintas pruebas.

- $k_p = 0.1$
- $k_i = 0.001$
- $k_d = 0.1$

Los valores propuestos se escogieron con base en la siguiente lógica:

- El valor de la ganancia k_d se escogió un valor menor para disminuir el sobrepaso del dron. También se redujo esta constante con el fin de reducir el tiempo del estado transitorio.

Con las nuevas ganancias propuestas se observó un mejor desempeño en los siguientes aspectos.

- Menor sobrepaso en la respuesta del sistema.
- Menor tiempo de asentamiento en la respuesta del sistema.

A pesar de solucionar algunos de los problemas de la primera configuración se observó que el problema relacionado con el sobrepaso persistía y el dron seguía perdiendo la referencia de la imagen.

En la figura 29 se muestra la gráfica correspondiente al posicionamiento del centro de la imagen objetivo sobre el eje *y* de la imagen. Se observa que la posición empieza a subir y lleva a un punto donde cambia drásticamente la dirección de la curva, este punto representa el momento en el que el dron pierde el seguimiento del objetivo. Esta pérdida de objetivo se observa en el primer segundo de ejecución debido a los valores de la señal de control.

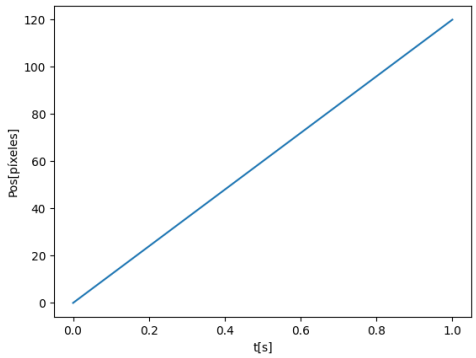

Figura 29. Comportamiento del sistema con la segunda configuración de ganancias, eje Z

En la figura 30 se muestra el comportamiento de la señal de control del sistema, que en este caso es la velocidad de desplazamiento del dron sobre el eje Z. Se observa como en el primer segundo la señal de control disminuye desde el valor máximo de 100 $\left[\frac{cm}{s}\right]$ indicando que el dron inmediatamente recibe un impulso para subir rápidamente, pero al sobrepasar la referencia se reduce hasta 45 $\left[\frac{cm}{s}\right]$ su velocidad, sin embargo, esta disminución de la velocidad tampoco es suficiente para lograr que el dron no pierda el objetivo de seguimiento.

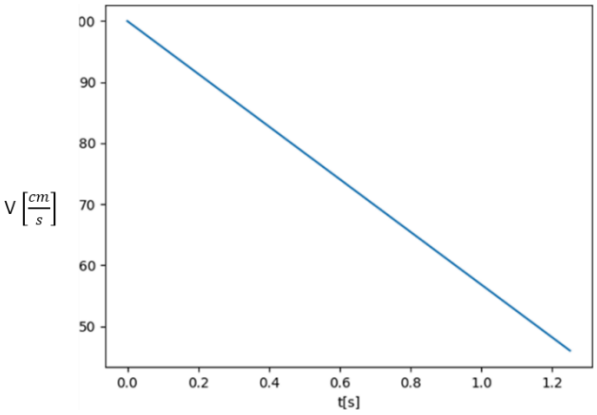

Figura 30. Señal de control con la segunda configuración de ganancias, eje Z

Para solucionar los problemas del controlador anterior se propuso una etapa adicional en la implementación del controlador. La etapa propuesta fue la implementación de un filtro de ventana que acota el valor máximo y mínimo de la señal de control. Se propusieron los siguientes límites de la señal de control en el filtro después de realizar algunas pruebas:

- Valor máximo de velocidad: $20\ \frac{cm}{s}$.
- Valor mínimo de velocidad: $-20\ \frac{cm}{s}$.

Estos parámetros propuestos se obtuvieron mediante una calibración experimental, seleccionando la configuración que a consideración cumplió de mejor manera el seguimiento de rostros.

Con las nuevas ganancias propuestas y el filtro implementado se observó un mejor desempeño en los siguientes aspectos.

- Un sobrepaso menor que conlleva a que el sistema no pierda la referencia al ejecutar su movimiento.
- Un estado transitorio que permite la identificación de rostros de manera adecuada.

En la figura 31 se muestra la gráfica correspondiente al posicionamiento del centro de la imagen objetivo sobre el eje *y* de la imagen. A diferencia de las configuraciones anteriormente propuestas la posición empieza a ajustarse durante un periodo transitorio de aproximadamente 20 segundos, después de ese tiempo el seguimiento del objetivo se completa. El resultado se observa en la figura 32.

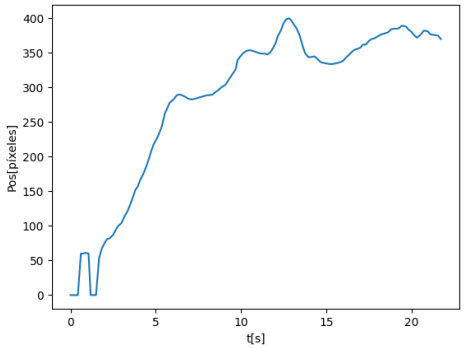

Figura 31. Comportamiento del sistema con la tercera configuración de ganancias, eje Z

Figura 32. Seguimiento del objetivo con la tercera configuración de ganancias, eje Z

En la figura 33 se muestra el comportamiento de la señal de control del sistema, que en este caso es la velocidad de desplazamiento del dron sobre el eje Z. Lo primero que debe analizarse es que la velocidad empieza en un valor menor debido al filtro de ventana, con el que se logra un valor máximo de 20 $\left[\frac{cm}{s}\right]$. Con este valor el dron se eleva hasta que sobrepasa el objetivo y empieza a disminuir la velocidad. Finalmente llega a un valor cercano a 0, este offset es provocado por las pequeñas corrientes de aire presentes en el lugar de las pruebas.

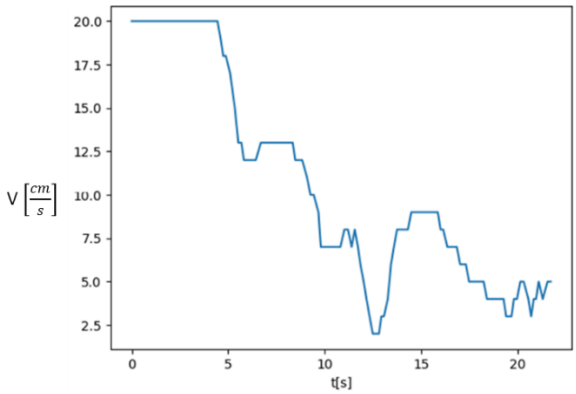

Figura 33. Señal de control con la tercera configuración de ganancias, eje Z

Control del movimiento YAW y sobre el eje Z en combinación

En la primera implementación del control de PID de ambos ejes de movimiento se siguió como base el esquema mostrado en la figura 34.

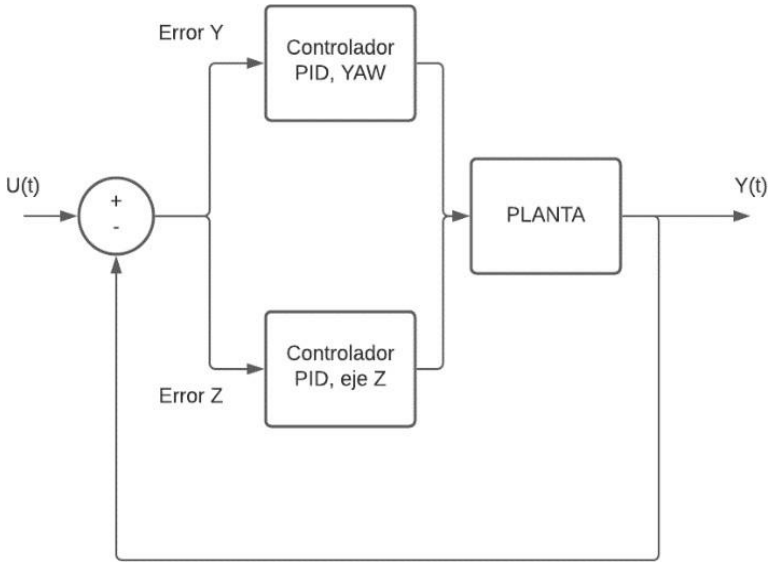

Figura 34. Esquema de control PID de YAW y desplazamiento en Z

Las ganancias utilizadas para la implementación de cada controlador son:

- Controlador PID YAW:
  - $k_p = 0.1$
  - $k_i = 0.001$
  - $k_d = 0.3$
- Controlador PID desplazamiento en eje Z:
  - $k_p = 0.1$
  - $k_i = 0.001$
  - $k_d = 0.1$

Adicionalmente en el control de desplazamiento sobre el eje Z se implementó el filtro de ventana con los siguientes valores máximos y mínimos.

- Valor máximo de velocidad: $20\ \frac{cm}{s}$.
- Valor mínimo de velocidad: $-20\ \frac{cm}{s}$.

Al realizar las pruebas con el controlador se observaron los siguientes problemas en la operación del sistema.

- Debido a que las instrucciones que utiliza el SDK del dron Tello para realizar el movimiento del dron mandan de manera conjunta los valores de velocidad a los actuadores se generaban movimientos que provocan interferencia en el movimiento final del dron.
- Dado que las dos señales de control interfieren entre sí, se obtenía una salida del sistema que no respondía de manera de manera adecuada a los controladores, evitando que la configuración planteada llevará a cabo el control del sistema.

Para solucionar los problemas presentados al combinar ambos controladores se optó por separar la ejecución de cada uno de los controladores. La implementación se ejemplifica en el diagrama 3 del anexo.

Mediante la implementación de este algoritmo se logró que el dron fuese capaz de seguir el objetivo propuesto de manera correcta sin que existiera interferencia en las señales.

Implementación del algoritmo de selección de objetivos de seguimiento

Algoritmo de reconocimiento de rostros objetivos

El algoritmo de reconocimiento de rostros utilizado se basa en los pasos sugeridos en la biblioteca *"Face Recognition"* de Adam Geitgey [9]. Los pasos del algoritmo son los siguientes:

- Detectar las caras que se encuentran en el *frame* analizado. Para esta acción se utiliza la función "*face_location*" que identifica los rostros dentro de una imagen.
- Codificar las 128 medidas necesarias para el algoritmo de identificación por *Deep Learning* en cada uno de los rostros detectados en la imagen. La función "*face_encodings*" se encarga de ejecutar este paso.
- Comparar cada uno de los rostros detectados en la imagen con el modelo del objetivo precargado en el programa. Para la evaluación se utiliza la función "*compare_faces*" que realiza un análisis comparativo de las 128 medidas y determina si existe semejanza entre los rostros comparados, también se obtiene el valor del porcentaje de error más grande entre las distancias comparadas con la función "*face_distance*".
- Se escogen como objetivos válidos los rostros que tengan un error menor al 50% en la distancia con mayor error.
- Se obtiene el área de cada uno de los rostros escogidos como válidos y se toma como objetivo a aquel rostro que tenga mayor área, el tener mayor área implica que es el rostro más cercano.

El diagrama 4 del anexo explica el proceso de detección de objetivos.

Algoritmo de seguimiento de objetivos en YAW

Como el seguimiento de objetivos se basa en identificar el rostro de una persona en específico para la identificación se requiere de un modelo de *Deep Learning* adicional para identificar el rostro buscado en la imagen, por lo que este proceso requiere de un mayor tiempo de ejecución con respecto al utilizado en los seguimientos anteriores que solo utilizaban el algoritmo Haar. Cuando se usa únicamente el algoritmo Haar para identificar rostros el periodo de muestreo es de 0.2 segundos y al agregar la comparación con el modelo de *Deep Learning* el periodo de muestreo se incrementa hasta los 0.9 segundos. Debido al incremento de tiempo de muestreo se determinó que el uso del control PID previamente diseñado no sería viable para el seguimiento de objetivos. Por lo anterior, en esta tesis propongo un método geométrico para realizar el seguimiento del objetivo sobre el eje correspondiente al movimiento *yaw*.

Para la solución geométrica, se parte de que al tomar una imagen con una cámara la luz se recolecta de manera que se forma un triángulo isósceles. Esta suposición se ejemplifica en la figura 35.

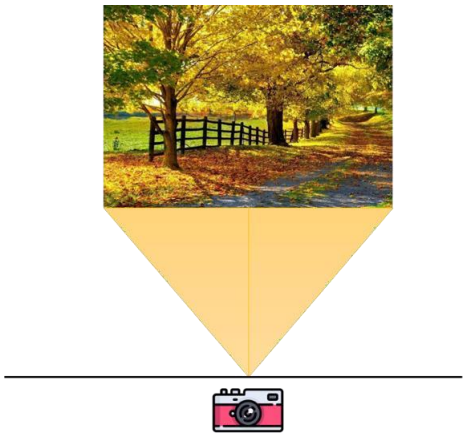

Figura 35. Modelo físico simplificado de captura de fotos (horizontal)

A partir de la simplificación del modelo de la figura 35 se puede plantear de manera geométrica una ecuación que permita conocer la distancia angular que habría que desplazarse para enfocar el centro de un objeto identificado. Para realizar el modelado se tomará únicamente uno de los triángulos rectángulos formados al partir el triángulo isósceles de la figura 35. A continuación se ejemplifica en la figura 36 el modelo utilizado para la obtención de la distancia angular deseada.

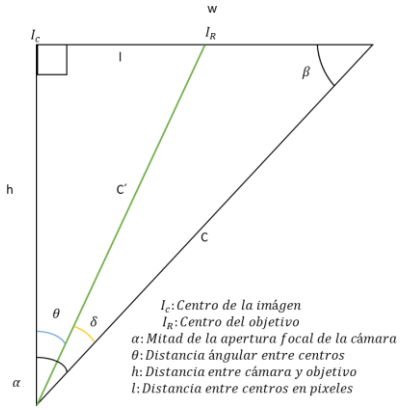

Figura 36. Modelo geométrico de la captura de imagen (horizontal)

A partir de las relaciones geométricas de cosenos, senos y triángulos equivalentes se definió la ecuación 14 para obtener el ángulo que indica la diferencia angular entre la imagen y el objetivo.

$$\theta = \alpha - angsen\left(\frac{w-l}{\sqrt{\left(\frac{\cos\alpha}{\sin\alpha}\right)^2 + w^2}}\sin\beta\right) \ldots (14)$$

Para poder aplicar la ecuación 14 es necesario conocer la apertura angular de la cámara porque α representa la mitad de ese parámetro. La cámara del dron Tello posee una apertura angular de 45°. Para definir al ángulo β se debe observar el modelo de la figura 36 donde se deduce que β es el ángulo complementario de α.

Algoritmo de seguimiento de objetivos en el eje Z

Se realizaron pruebas utilizando un algoritmo de control PID para el seguimiento de objetivos, dado que con esa implementación se obtuvo un periodo de muestreo de 0.9 segundos comparado con el tiempo de muestreo de 0.2 segundos sin el algoritmo de selección de objetivo se concluyó que con esa configuración de control no sería viable el seguimiento. Por lo anterior, en esta tesis propongo un método geométrico para realizar el seguimiento del objeto sobre el eje Z.

Para la solución geométrica, se parte de que al tomar una imagen con una cámara la luz se recolecta de manera que se forma un triángulo isósceles. Esta suposición se ejemplifica en la figura 37.

Figura 37. Modelo físico simplificado de captura de fotos (vertical)

A partir de la simplificación del modelo de la figura 37 se puede plantear de manera geométrica una ecuación que permita conocer la distancia que habría que desplazarse para enfocar el centro de un objeto identificado. Para realizar el modelado se tomará únicamente uno de los triángulos rectángulos formados al partir el triángulo isósceles de la figura 37. A continuación se ejemplifica en la figura 38. el modelo utilizado para la obtención de la distancia deseada.

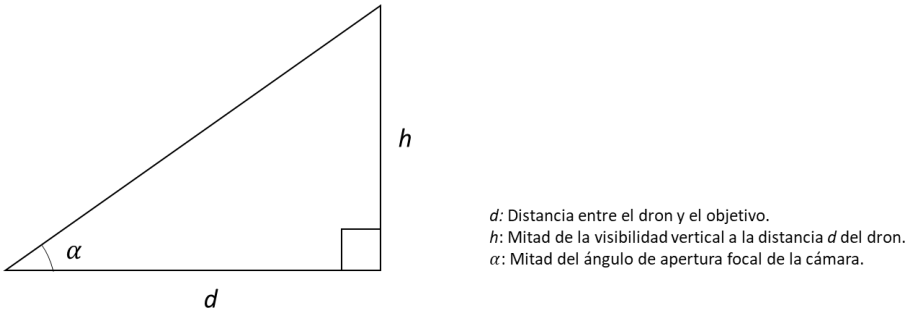

Figura 38. Modelo geométrico de la captura de imagen (vertical)

A partir del modelo de la figura 38 se puede obtener el valor de la altura real, ecuación 15, que tendría la visibilidad vertical ($2h$) a partir de una distancia al objetivo propuesta.

$$h = d\ tan\ \alpha \quad \text{... (15)}$$

Para poder definir un valor constante de la distancia al objetivo se propuso el valor de 2.5 m, el resultado de proponer este valor implica que si el objetivo se encuentra en la distancia propuesta el dron deberá alinearse perfectamente con el objetivo; en caso de que el objetivo se encuentre a una mayor distancia del objetivo el dron alinea la cámara con el centro del objeto de seguimiento, pero en la realidad estará por encima de él; en caso de que el objetivo se encuentre a una menor distancia el objetivo el dron alinea la cámara con el centro del objeto de seguimiento, pero en la realidad estará por debajo de él.

Dado que se conoce el tamaño real en píxeles de la imagen y la distancia en píxeles que hay entre el centro de la imagen y el del objetivo se puede aplicar semejanza de triángulos para obtener la ecuación 16 que permite determinar la distancia necesaria para moverse en cm a partir del error en pixeles.

$$z = error\ \frac{h}{h_{pixeles}} \quad \ldots (16)$$

Algoritmo de seguimiento de objetivos en YAW y en el eje Z

Para realizar el funcionamiento del algoritmo de seguimiento de objetivos en ambos ejes se combinaron los algoritmos de seguimiento de cada eje. La manera en que funciona el algoritmo es la siguiente:

- Detección del objetivo dentro de la imagen.
- Cálculo del error, vertical y horizontal, entre el centro de la imagen y del objetivo.
- Cálculo del movimiento vertical y horizontal necesario para centrar la imagen.
- Realizar el movimiento vertical para ajustar el seguimiento.
- Realizar la rotación horizontal (YAW) para ajustar el seguimiento.
- Repetir la detección del objetivo hasta que se indique el aterrizaje del dron.

El diagrama 5 del anexo ejemplifica el funcionamiento del algoritmo.

RESULTADOS

Prueba de seguimiento sobre el eje rotacional correspondiente al movimiento YAW

Como ya se mencionó, durante las pruebas de calibración del controlador los valores de las ganancias del controlador PID son:

- $k_p = 0.1$
- $k_i = 0.001$
- $k_d = 0.3$

Una vez determinados los valores de las ganancias del controlador se procedió a realizar pruebas con rostros en movimiento. En la figura 39 se muestra el resultado del seguimiento de rostros sobre el eje del movimiento YAW. En la siguiente liga se encuentra una muestra de la prueba https://youtu.be/d_HmoAPwP20.

Figura 39. Seguimiento de rostros sobre el eje YAW

Prueba de seguimiento sobre el eje vertical

En las pruebas de calibración del controlador se escogieron los siguientes valores de las ganancias del controlador PID:

- $k_p = 0.1$
- $k_i = 0.001$
- $k_d = 0.1$

Una vez determinados los valores de las ganancias del controlador se procedió a realizar pruebas con rostros en movimiento. En la figura 40 se muestra el resultado del seguimiento de rostros sobre el eje Z. En la siguiente liga se encuentra una muestra de la prueba https://youtu.be/I7DWS0Ggptg.

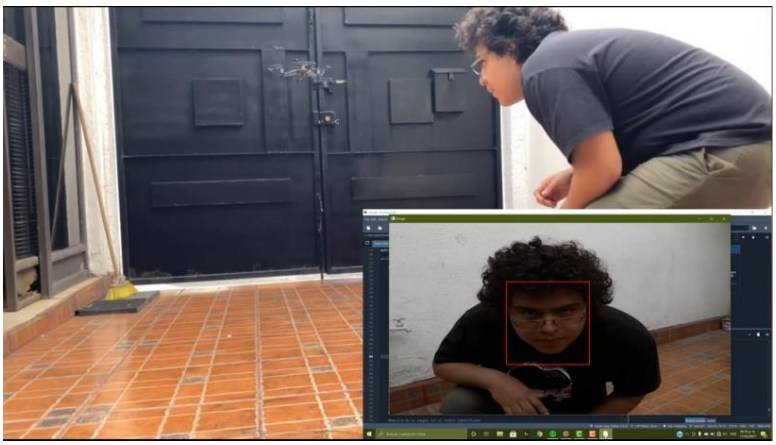

Figura 40. Seguimiento de rostros sobre el eje

Prueba de seguimiento sobre el eje vertical, desplazamiento lineal y rotacional.

Mediante la implementación del esquema de control correspondiente con los controladores PID de ambos desplazamientos combinados se obtuvo un programa capaz de controlar ambos movimientos sin que existiera interferencia entre los controladores.

Una vez implementados los controladores se procedió a realizar pruebas con rostros en movimiento. En la figura 41 se muestra el resultado del seguimiento de rostros controlando los dos movimientos posibles sobre el eje vertical. En la siguiente liga se encuentra una muestra de la prueba https://youtu.be/3UogYCdYcBM.

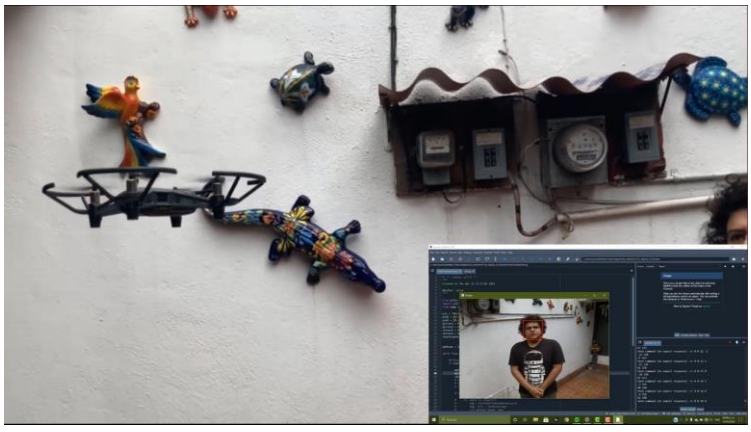

Figura 41. Seguimiento de rostros sobre el eje Z

Prueba de seguimiento aislando un objetivo

Una vez implementados los métodos de identificación de objetivos y los métodos geométricos para calcular las distancias de movimientos se procedió a realizar pruebas en movimiento con más de un rostro detectado en la imagen. En la figura 42 se muestra el resultado de estas pruebas. En la siguiente liga se encuentra una muestra de la prueba https://youtu.be/ezNLIfupn-4.

Figura 42. Seguimiento de objetivos controlando los dos movimientos posibles sobre el eje Z

CONCLUSIONES

Mediante el presente trabajo se demostró que, a través de la integración de distintas tecnologías y áreas, como la Inteligencia Artificial y control automático, se pudieron alcanzar los objetivos planteados. Además, para alcanzar los objetivos planteados se utilizaron herramientas de software libre como las bibliotecas de Tello y OpenCV, así como el dron Tello Edu.

En el diseño de los controladores PID para el movimiento *yaw* y el movimiento sobre el eje Z se obtuvieron los valores de ganancias de forma experimental; ya que debido a las limitaciones en cuanto al conocimiento de los parámetros del dispositivo y a la limitante de acceso al control de los actuadores que permite el SDK del fabricante del dron fue más útil diseñar un control PID de manera experimental, mediante la realización de pruebas para el ajuste empírico, que por algún otro método que requiriera el desarrollo de un modelo matemático. En consecuencia, no se puede garantizar que el controlador diseñado sea el óptimo para resolver el problema, sin embargo, se buscó llegar a una solución que permitiera cumplir los objetivos de una satisfactoria.

Se trató de implementar en el seguimiento de objetivos el controlador PID diseñado para el control de movimiento en ambos ejes en seguimiento de rostro, pero no fue posible su implementación debido a que el algoritmo utilizado para la detección de objetivos requiere de una gran capacidad de procesamiento de cómputo y el equipo que se utilizó para realizar las pruebas carecía de la suficiente capacidad para realizar un procesamiento rápido de la imagen. Esto demoraba el periodo de muestreo de la imagen, provocando que el sistema fuese incapaz de llevar un correcto seguimiento del objetivo por medio del controlador PID.

Para solucionar el seguimiento de un objetivo y con la imposibilidad de utilizar un controlador PID, se optó por utilizar métodos geométricos que permitieran llevar acabo un cálculo aproximado del movimiento que tendría que realizar el dron para encuadrar la imagen de manera correcta. Si bien este método satisface el poder realizar el seguimiento con el dron, debido a las simplificaciones para el modelad, se tiene un error al detectar objetivos que están muy alejados del rango de operación propuesto, sobre todo en el seguimiento de la altura.

En conclusión, se cumplieron los objetivos planteados en el presente trabajo; sin embargo, es importante reflexionar sobre los márgenes de mejora existentes para la implementación desarrollada.

Un punto importante de mejora se relaciona directamente con el SDK y el modo de conexión del dron. En primer lugar, se debería de buscar el acceso a poder controlar la totalidad de los actuadores de los que se disponen para que el desarrollo de un controlador óptimo sea factible. En segundo lugar, se sugiere buscar un dron o la manera de implementar el sistema de control dentro del dron, con ello se reduciría el tiempo de retardo que existe entre el dron y el sistema de control para realizar el análisis de la información de manera rápida y que permita una mejor respuesta del sistema.

Otro punto importante de mejora es el contar con un equipo de cómputo especializado para el reconocimiento de imágenes por medio de IA, para que la respuesta del dron al buscar un objetivo sea en un menor tiempo y permita la implementación de sistemas de control óptimos.

Además, para mejorar el rendimiento general del controlador PID conviene utilizar un modelado del sistema en espacio discreto que nos permita conocer de mejor manera el comportamiento dinámico del dron, de esta manera se podría implementar mejores valores de calibración para el controlador

Como observación adicional, en caso de seguir trabajando con la misma plataforma es recomendable utilizar algún método experimental definido para la calibración de controladores PID, de esta manera se garantizaría la obtención de un controlador óptimo a pesar de no contar el modelo matemático.

REFERENCIAS

[1] Romero L., Pozo D. & Rosales J. (2014). Quadcopter stabilization by using PID controllers. junio 2021, de Universidad de Cuenca Sitio web: https://publicaciones.ucuenca.edu.ec/ojs/index.php/maskana/article/view/585/509

[2] Lee K., Kim Y & Hong Y. (2018). Real-Time Swarm Search Method for Real-World Quadcopter Drones. MDPI, Julio 2018, 12.

[3] Ogata K. (2010). Modern Control Engineering. New Jersey: Prentice Hall.

[4] Franklin F., Powell J. & Workman M.. (1998). Digital Control of Dynamic Systems. Half Moon Bay, CA: Ellis-Kagle Press.

[5] Anónimo. (s.f.). Cascade Classifier. junio 2021, de OpenCV Sitio web: https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html

[6] Geitgey A. (2016). Machine Learning is Fun! Part 4: Modern Face Recognition with Deep Learning. junio 2021, de Medium Sitio web: https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78

[7] Anónimo (2018). Tello SDK 2.0 User Guide. junio 2021, de Ryzen Tello Sitio Web: https://dl-cdn.ryzerobotics.com/downloads/Tello/Tello%20SDK%202.0%20User%20Guide.pdf

[8] Murtaza´s Workshop Robotics and AI: https://www.murtazahassan.com/

[9] Geitgey A. (2020). Face Recognition 1.3.0. junio 2021, de Python Package Index Sitio web: https://pypi.org/project/face-recognition/

BIBLIOGRAFÍA

- Brawnlee J. (2019). A Gentle Introduction to Computer Vision. junio 2021, de Machine Learning Mastery Sitio web: https://machinelearningmastery.com/what-is-computer-vision/

- Anónimo. (s.f.). Target face detection + recognition (Python + OpenCV + Keras). junio2021, de Programmer Sought Sitio web: https://www.programmersought.com/article/48795442100/

- Anónimo. (s.f). Deep Learning Tres cosas que es necesario saber. junio 2021, deMathWorks Sitio web: https://la.mathworks.com/discovery/deep-learning.html

- Arrabales R. (2016). Deep Learning: qué es y por qué va a ser una tecnología clave en el futuro de la inteligencia artificial. junio 2021, de Xataka Sitio web: https://www.xataka.com/robotica-e-ia/deep-learning-que-es-y-por-que-va-a-ser-una-tecnologia-clave-en-el-futuro-de-la-inteligencia-artificial

- García C. (s.f). ¿Qué es el Deep Learning y para qué sirve?, junio 2021, de Indra Company Sitio web: https://www.indracompany.com/es/blogneo/deep-learning-sirve

- Fuentes D. (2018). DJI Tello Py. junio 2021, de GitHub Sitio web: https://github.com/damiafuentes/DJITelloPy

- Nise N. (2011). Control Systems Engineering. California, USA: John Wiley & Sons, Inc.

- Knospe C. (2006). PID Control. IEEE Control Systems Magazine, Febrero 2006, 2.

- Nieto E. & Vaca F. (2020). Desarrollo de un modelo matemático, cinemático y dinámico con la aplicación de software, para modificar el funcionamiento de un dron, para que este realice monitoreo automático. RECIMUNDO, marzo 2020, 332-343.

ANEXOS.

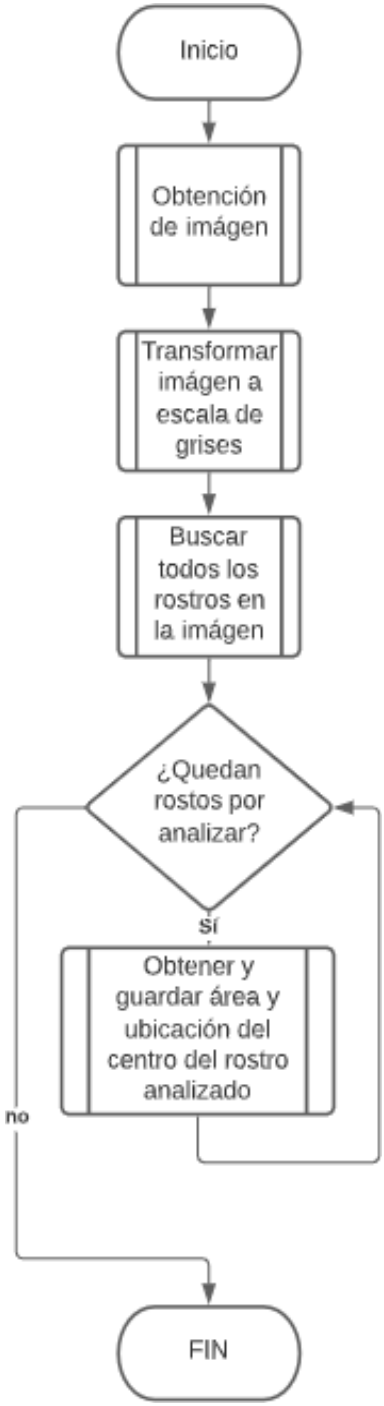

Diagrama 1. Diagrama de flujo del reconocimiento de imagen

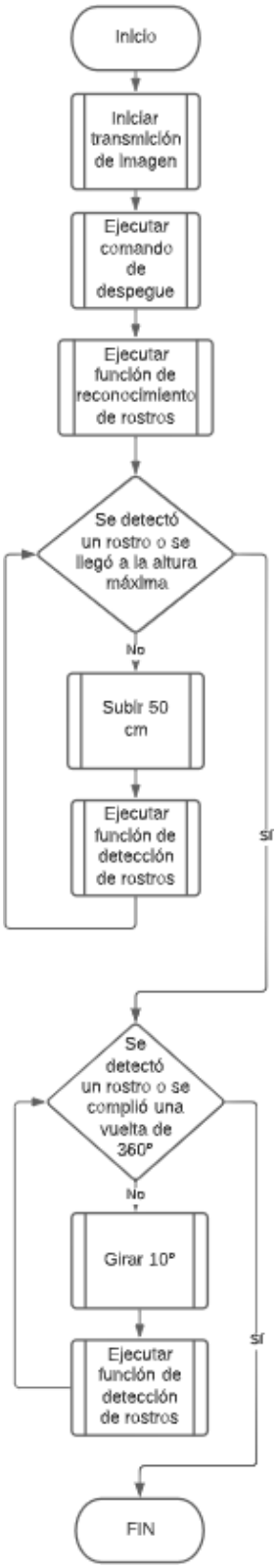

Diagrama 2. Diagrama de flujo de la secuencia de despegue

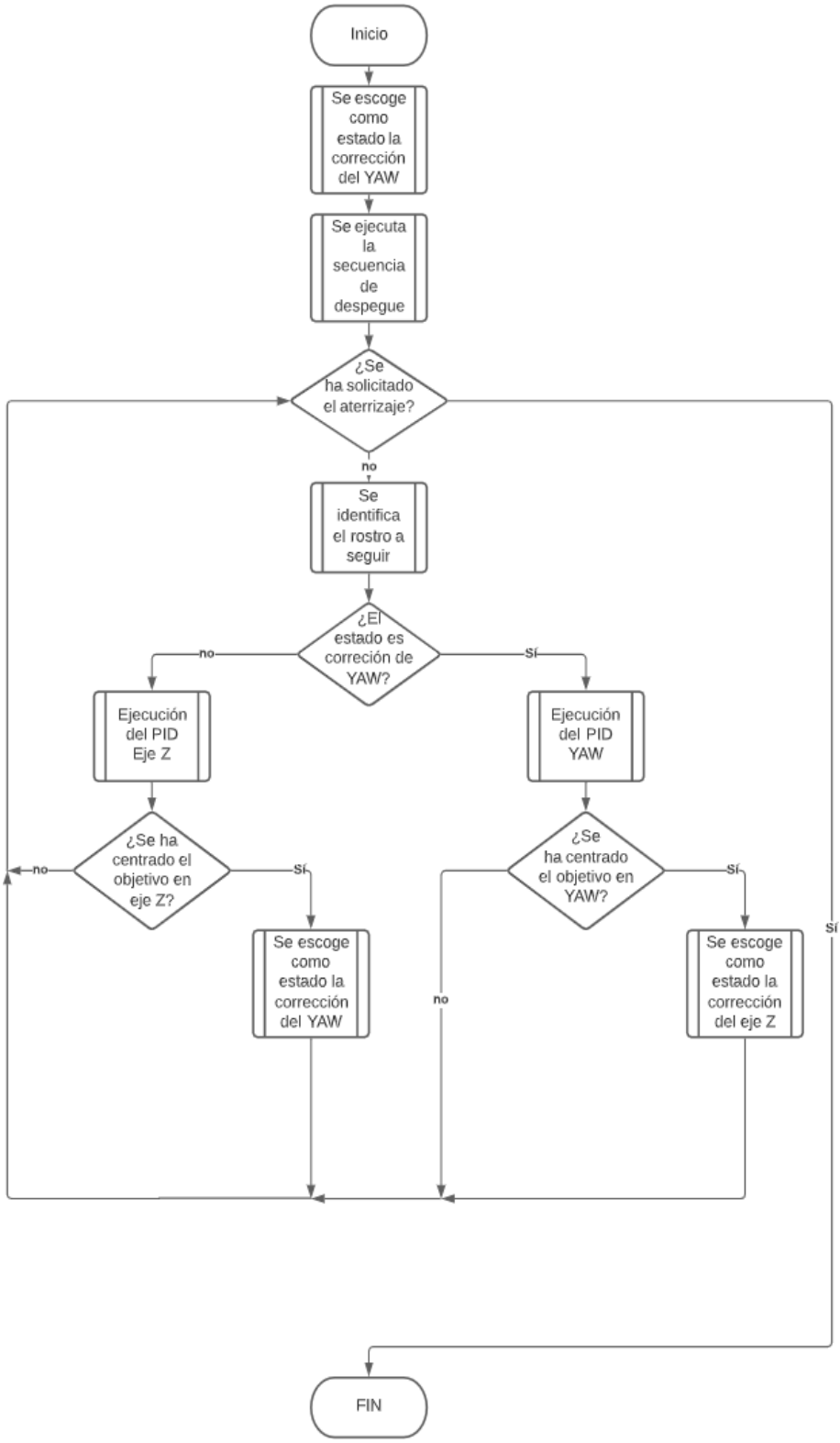

Diagrama 3. Diagrama de flujo de la implementación del seguimiento PID eje YAW y eje Z

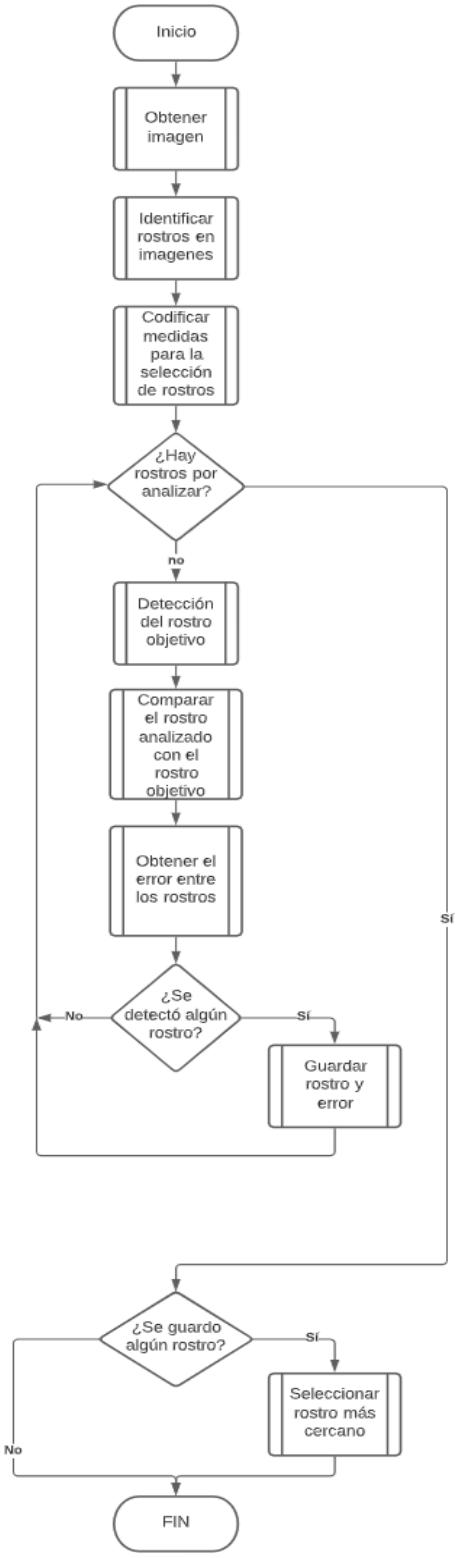

Diagrama 4. Diagrama de flujo del reconocimiento de un rostro objetivo

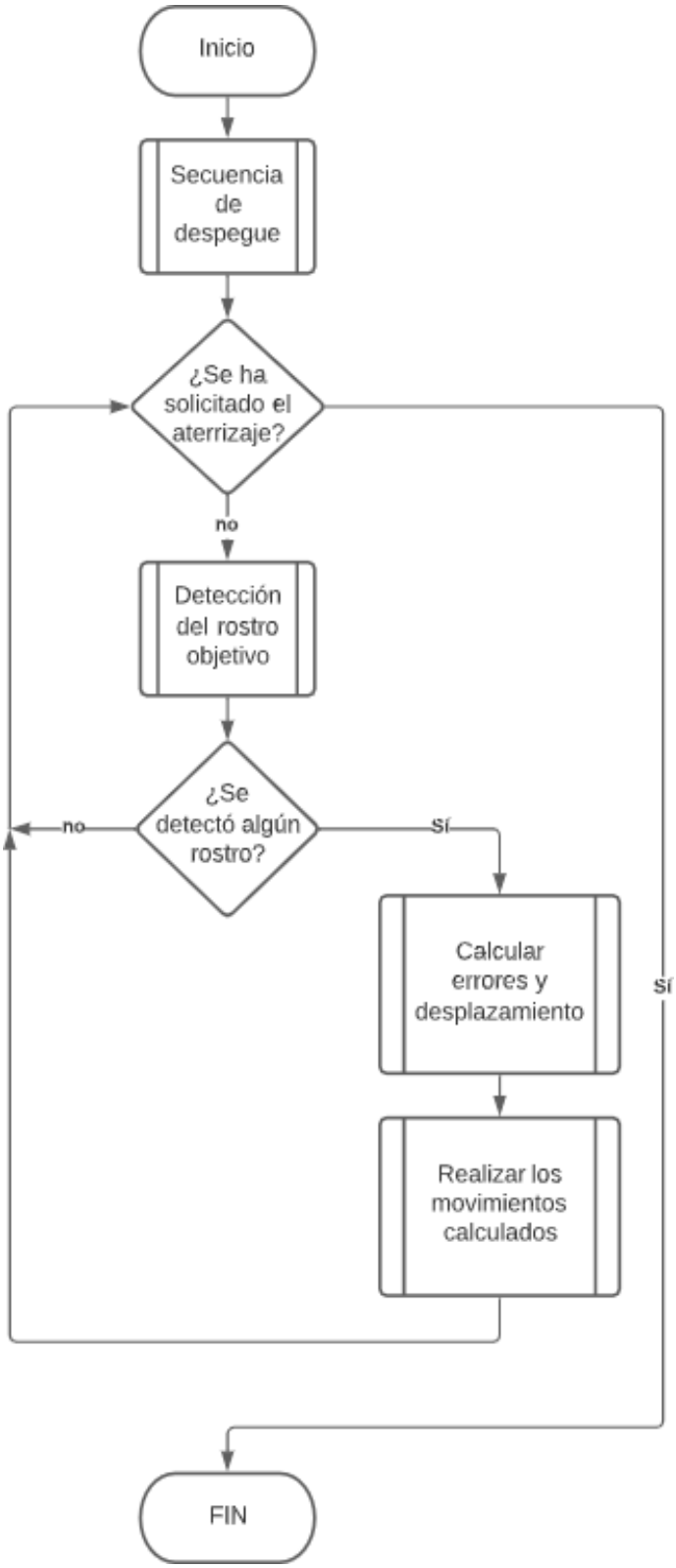

Diagrama 5. Diagrama de flujo del seguimiento de objetos en eje Z con ambos desplazamientos