Escuela Politécnica Superior de Jaén

Grado en Ingeniería Electrónica Industrial

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior

Trabajo Fin de Grado

EL DRON TELLO COMO HERRAMIENTA DOCENTE EN LA ASIGNATURA: VISIÓN POR COMPUTADOR

Alumno: Javier Latorre Lorite

Tutora: Prof. Dña. Silvia Satorres Martínez
Tutor: Prof. D. Diego Manuel Martínez Gila
Dpto: Ingeniería Electrónica y Automática

NOVIEMBRE, 2023

Universidad de Jaén
Escuela Politécnica Superior de Jaén
Departamento de Ingeniería Electrónica y Automática

Doña Silvia Satorres Martínez y Don Diego Manuel Martínez Gila, tutores del Trabajo Fin de Grado titulado: El dron Tello como herramienta docente en la asignatura: Visión por Computador, que presenta Javier Latorre Lorite, autoriza su presentación para defensa y evaluación en la Escuela Politécnica Superior de Jaén.

Jaén, Noviembre de 2023

El alumno:

LATORRE LORITE JAVIER - 77434479B

Firmado digitalmente por LATORRE LORITE JAVIER - 77434479B
Fecha: 2023.11.20 03:26:58 +01'00'

JAVIER LATORRE LORITE

Los tutores:

SILVIA SATORRES MARTÍNEZ

Firmado digitalmente por MARTINEZ GILA DIEGO MANUEL - 77338812R
Fecha: 2023.11.20 11:51:15 +01'00'

DIEGO MANUEL MARTÍNEZ GILA

Resumen

El objetivo de este trabajo es integrar el dron Tello como una herramienta docente en la asignatura de visión por computador. Para conseguir esto, primero se han explorado las posibilidades que el dron Tello puede ofrecer, diseñando un caso práctico basado en un dron seguidor de línea. Este proceso se ha llevado a cabo con el entorno de programación utilizado en la optativa de visión, que es Matlab, completado con App Designer para el desarrollo de interfaz y un paquete de soporte para el control del dron Tello con Matlab. El desarrollo del software ha sido dividido en procesamiento de imagen y control de movimiento, creando así un sistema de navegación autónomo. Tras finalizar el desarrollo, se han conseguido resultados satisfactorios, aunque con una ejecución poco fluida debido a ciertas limitaciones del paquete de soporte. Por último, se culmina con la creación de unas prácticas dedicadas a la asignatura de visión por computador, en las cuales se implementa el dron Tello como una herramienta innovadora.

Palabras clave: dron Tello, seguidor de línea, Matlab, App Designer, visión por computador.

Abstract

The aim of this work is to integrate the Tello drone as a teaching tool in the Computer Vision course. In order to achieve this goal, the possibilities offered by the Tello drone were first explored by designing a case study based on a line-following drone. This process was carried out using the programming environment used in the Computer Vision course, Matlab, complemented by App Designer for interface development and a support package for controlling the Tello drone with Matlab. The software development was divided into image processing and motion control to create an autonomous navigation system. At the end of the development, satisfactory results were obtained, although the execution has not been fluent due to certain limitations of the support package. Finally, it culminates with the creation of two practice sessions dedicated to the Computer Vision course, in which the Tello drone is implemented as an innovating tool.

Key words: Tello drone, line follower, Matlab, App Designer, computer visión.

Agradecimientos

Quiero dar las gracias a mis padres, hermanos y pareja, por el apoyo y ánimo recibido durante toda mi etapa universitaria. Gracias a los amigos que me han acompañado durante los años del grado y con los que he compartido muy buenos momentos. Gracias a la Universidad de Jaén que ha hecho posible realizar mis estudios de erasmus en Wroclaw, donde he conocido a gente maravillosa.

También quiero dar las gracias a mis tutores Silvia y Diego, por el apoyo recibido para llevar a cabo este trabajo de fin de grado, con el que finalizo esta etapa universitaria.

Índice

1. INTRODUCCIÓN 1
1.1. Motivación 1
1.2. Objetivos 2
1.3. Estructura de la memoria 2
2. ESTADO DEL ARTE 4
2.1. Historia y evolución de los drones 5
2.2. Clasificación de los drones 9
2.2.1. Aerostatos 9
2.2.2. Aerodinos 10
2.3. Aplicaciones 12
2.3.1. Sector industrial 12
2.3.2. Sector servicios 14
3. MATERIALES Y MÉTODOS 16
3.1. Dron Tello 16
3.1.1. Componentes 17
3.1.2. Accesorio de visión 21
3.1.3. Modo conexión de Tello 23
3.2. Matlab 24
3.2.1. App Designer 25
3.2.2. Extensión para dron Tello 29
4. DESARROLLO SOFTWARE: SEGUIDOR DE LÍNEA 32
4.1. Procesamiento de imagen 34
4.1.1. Adquisición 35
4.1.2. Preprocesamiento 35
4.1.3. Extracción de características 37
4.2. Sistema de navegación 38
4.2.1. Cálculo de errores y asignación de variables 39
4.2.1. Órdenes de movimiento 41
5. RESULTADOS 43
5.1. Interfaz: seguidor de líneas 43
5.2. Guiones de Practicas 47
5.2.1. Práctica 1: Interfaz de control del dron Tello 47
5.2.2. Practica 2: El dron Tello. Seguidor de línea 48

6. CONCLUSIONES 49
Anexo I. Práctica 1: Interfaz de control del dron Tello 51
Anexo II. Práctica 2: El dron Tello. Seguidor de línea 59
Bibliografía 64

Índice de Figuras

Figura 2.1. “The Pidgeon” [1]..... 5
Figura 2.2. Representación del ataque austríaco en Venecia [2]..... 6
Figura 2.3. Teleautomatón de Nikola Tesla..... 6
Figura 2.4. Kettering Bug 7
Figura 2.5. Línea temporal del uso militar y civil de los drones [5] 8
Figura 2.6. Estadísticas internacionales del sector de drones (2006-2016) [6] .. 8
Figura 2.7. Globo no tripulado 10
Figura 2.8. Ejemplos de UAVs de ala rotatoria..... 11
Figura 2.9. Ejemplos de UAVs de ala fija 11
Figura 2.10. Dron WingtraOne (VTOL) [9]..... 12
Figura 2.11. Uso de drones en el sector industrial [10]. 13
Figura 2.12. Dragonfly [11]. 14
Figura 3.1. Dron Tello..... 16
Figura 3.2. Estructura del dron Tello 17
Figura 3.3. Diagrama de la aeronave [15] 18
Figura 3.4. Chasis del dron Tello..... 18
Figura 3.5. Motores 19
Figura 3.6. Hélices y protectores..... 19
Figura 3.7. PCB con componentes electrónicos..... 20
Figura 3.8. Batería de dron Tello..... 20
Figura 3.9. PCB con sensores infrarrojos y VPS..... 21
Figura 3.10. Clip de espejo para dron Tello 22
Figura 3.11. Dron Tello con espejo montado..... 22
Figura 3.12. La ley de la reflexión [17] 23
Figura 3.13. Botón de Encendido..... 23
Figura 3.14. Indicador de estado de la aeronave 24
Figura 3.15. Red de conexión 24
Figura 3.16. Nueva plantilla en App Designer 26
Figura 3.17. Plantilla de App Designer con componentes..... 27
Figura 3.18. Vista de código en App Designer 27
Figura 3.19. Ejemplo de Callback para pulsar un botón..... 28
Figura 3.20. Conexión con el dron Tello en Matlab..... 30
Figura 4.1. a: Línea abierta; b: Línea cerrada 32
Figura 4.2. Flujograma principal del proceso..... 33
Figura 4.3. Flujograma del procesamiento de imagen 34
Figura 4.4. Imágenes obtenidas con el dron Tello..... 35
Figura 4.5. a: Original; b: Invertida 35
Figura 4.6. a: Original; b: Escala de grises..... 36
Figura 4.7. a: Original; b: Binarizada..... 36
Figura 4.8. a: Binarizada; b: Complementaria 37
Figura 4.9. Imagen con centroide representado..... 38
Figura 4.10. Representación de la orientación de un objeto 38
Figura 4.11. Flujograma del sistema de navegación..... 39

Figura 4.12. Ejes de coordenadas de la imagen 40
Figura 4.13. Calibración de la longitud real de la imagen..... 41
Figura 4.14. Mapa con regiones de movimiento..... 42
Figura 5.1. Interfaz de seguidor de línea..... 43
Figura 5.2. Interfaz conectada con el dron Tello 44
Figura 5.3. Interfaz con cámara activada 45
Figura 5.4. Proceso de calibración..... 45
Figura 5.5. Proceso de seguimiento de la línea 46
Figura 5.6. Seguimiento de circuito cerrado..... 46
Figura 5.7. Seguimiento de circuito abierto..... 47

Índice de Tablas

Tabla 3.1. Comandos de conexión 29
Tabla 3.2. Comandos de adquisición de imagen 30
Tabla 3.3. Comandos básicos de navegación 31
Tabla 3.4. Comandos avanzados de navegación 31
Tabla 3.5. Comandos de datos de vuelo 31

1. INTRODUCCIÓN

1.1. Motivación

La educación está experimentando una transformación sin precedentes en la era digital y la integración de tecnologías emergentes en el proceso de enseñanza-aprendizaje se ha vuelto inevitable. La visión por computador es un campo de estudio de gran relevancia dentro del campo de la informática, que busca dotar a los estudiantes de las habilidades necesarias para comprender y aplicar conceptos relacionados con el procesamiento de imágenes y la percepción visual. En este contexto, la integración de dispositivos como el dron Tello ofrece una oportunidad única para enriquecer la experiencia educativa de los estudiantes y promover con mayor fuerza un aprendizaje activo, práctico y motivador.

El dron Tello, desarrollado en colaboración entre las empresas Ryze Tech y DJI, es una plataforma asequible y versátil que combina la tecnología de vuelo con una cámara de alta calidad. Su facilidad de uso y programación lo convierten en una herramienta ideal para explorar los conceptos de visión por computador en un entorno real. Al aprovechar las capacidades de vuelo autónomo y la captura de imágenes, los estudiantes pueden experimentar de manera práctica cómo los algoritmos de procesamiento de imágenes pueden aplicarse en situaciones del mundo real.

La elección de utilizar el dron Tello como herramienta docente en la asignatura de Visión por Computador se basa en varios factores:

- Aprendizaje: los estudiantes pueden aplicar directamente los conceptos teóricos aprendidos en clase a través de la programación y control del dron Tello. Esto fomenta un aprendizaje activo y práctico.
- Interdisciplinariedad: la combinación de la visión por computador con la tecnología de drones aborda de manera interdisciplinaria temas relevantes en electrónica, robótica y más, lo que amplía la perspectiva de los estudiantes.

- Motivación y Compromiso: el uso de una tecnología emocionante como los drones captura la atención e interés de los estudiantes, lo que puede aumentar su motivación y compromiso con la materia.
- Preparación para el mundo real: el conocimiento y las habilidades adquiridas al trabajar con el dron Tello tienen aplicaciones prácticas en la industria, donde la visión por computador y la robótica están en constante crecimiento.

1.2. Objetivos

Los principales objetivos que se pretenden alcanzar con este trabajo de fin de grado son los siguientes:

- Analizar y evaluar el potencial del dron Tello como herramienta docente.
- Diseñar y desarrollar casos prácticos que aprovechen las capacidades del dron Tello para enseñar conceptos clave en visión por computador, con la correspondiente interfaz que facilite su uso.
- Proporcionar recomendaciones y pautas para la conexión del dron Tello en las prácticas utilizando el entorno de MATLAB.
- Redactar guías de prácticas adaptadas al caso realizado con el dron Tello.

En resumen, este trabajo tiene como objetivo explorar cómo el dron Tello puede enriquecer el proceso de enseñanza-aprendizaje en la asignatura de Visión por Computador. La combinación de teoría y práctica a través de esta herramienta promete una experiencia educativa más emocionante y efectiva para los estudiantes, preparándolos para futuros desafíos tecnológicos y fomentando un enfoque de aprendizaje novedoso que se alinea con las demandas de la sociedad digital actual.

1.3. Estructura de la memoria

La memoria de este Trabajo de Fin de Grado se inicia con una introducción que sienta las bases del estudio, abordando la relevancia del uso de drones en la educación y, en particular, la aplicación del dron Tello en la asignatura de visión por

computador. Se establecen los objetivos y se anticipa la organización del material dentro de esta estructura de la memoria.

Se continúa con una visión general del estado del arte, proporcionando un marco teórico que incluye la evolución histórica de los drones, su clasificación y aplicaciones civiles.

Posteriormente, se describe detalladamente los materiales empleados para la realización del proyecto. Se desglosan las especificaciones técnicas del dron Tello y las herramientas de software, con especial énfasis en Matlab y App Designer.

El desarrollo del caso práctico constituye la parte central de la memoria, donde se explora la capacidad del dron Tello como seguidor de línea. Se pretende integrar los conocimientos de visión por computador con el control de dron Tello. Este apartado detalla el proceso desde la captura y procesamiento de imágenes hasta la navegación y control del dron.

La culminación del trabajo se presenta en los resultados obtenidos, donde se muestran evidencias de la integración entre el dron Tello y Matlab, la interfaz de navegación creada para su control y los logros del dron en las pruebas prácticas, reflejando la viabilidad del sistema desarrollado. Por último, se presentan unos guiones de prácticas dedicados a la asignatura de visión por computador, que pretenden integrar el dron como una herramienta docente, basándose en el proceso de aprendizaje realizado para concluir el caso práctico.

2. ESTADO DEL ARTE

En la última década, hemos sido testigos de una auténtica revolución en la tecnología aérea, y los drones han sido protagonistas destacados en esta transformación. Los drones también conocidos como UAVs (Unmanned Aerial Vehicles) o RPAS (Remotely Piloted Aircraft Systems), han experimentado un crecimiento exponencial en su popularidad y utilidad en diversas esferas de la sociedad.

Este crecimiento vertiginoso no es casualidad, en gran medida, se debe a la incorporación de avanzados sensores y cámaras que han elevado la capacidad de estos dispositivos a niveles superiores nunca antes imaginados. Los drones se han convertido en herramientas versátiles capaces de capturar información detallada desde perspectivas aéreas lo que ha revolucionado numerosas industrias y sectores.

Uno de los campos donde los drones han dejado una huella profunda es en el ámbito laboral. Desde la agricultura de precisión hasta la inspección de infraestructuras, los drones han optimizado procesos, reducido costes y han minimizado riesgos. Ejemplos concretos incluyen la monitorización de líneas de transmisión eléctrica, la inspección de puentes y edificios de difícil acceso, la grabación de escenas para películas e incluso espectáculos de luces.

Sin lugar a dudas, uno de los avances tecnológicos más significativos que ha impulsado el amplio rango de aplicaciones de los drones es la visión por computador. La capacidad de estos dispositivos para procesar imágenes y datos visuales en tiempo real ha abierto una nueva dimensión en términos de aplicaciones y funcionalidades.

La visión por computador ha habilitado aplicaciones como la detección y seguimiento de objetos, la creación de mapas tridimensionales precisos, la identificación de patrones y anomalías. Esta capacidad no solo aumenta la eficiencia de las tareas realizadas por los drones, sino que también amplia su alcance en áreas como la seguridad, la búsqueda y rescate, la agricultura de precisión y la monitorización medioambiental. La combinación de drones y la visión por computador representa una sinergia tecnológica que ha transformado

profundamente la forma en que abordamos una amplia gama de desafíos y oportunidades en la sociedad moderna.

Adicionalmente, la presencia de los UAVs en la docencia es una innovación en el proceso educativo, y existen drones de diversas características y tamaños, que se adaptan a diferentes niveles de enseñanza, desde la básica hasta la investigación doctoral. Su aplicación en la educación abarca la enseñanza de conceptos científicos, el manejo del dron, e incluso cuestiones éticas y legales relacionadas con su uso.

2.1. Historia y evolución de los drones

A lo largo de la historia, la humanidad ha tratado de sobrepasar sus limitaciones, en concreto, la idea de poder volar como lo hacen las aves ha llevado a numerosos intentos por parte del ser humano de conseguir este objetivo.

Tenemos que remontarnos al año 425 a.C. cuando surgió lo que se conoce como la primera máquina voladora autónoma de la antigüedad, “The Pidgeon” construido por Arquitas de Tarento, en el sur de Italia [1]. Este invento mostrado en la Figura 2.1, fue muy avanzado a su época, estaba propulsado a vapor y se estima que consiguió volar alrededor de 200 metros.

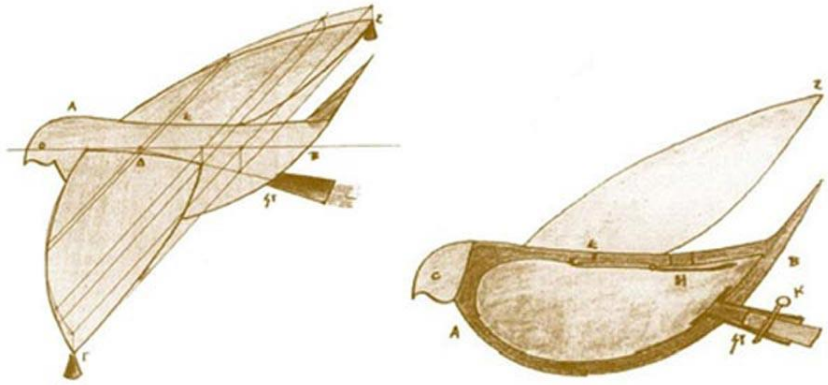


Figura 2.1. “The Pidgeon” [1]

En la Edad Contemporánea, los avances continuaron. En el año 1849, se documentó uno de los primeros intentos de usar vehículos aéreos no tripulados, los austriacos emplearon más de 200 globos aerostáticos cargados con bombas para atacar Venecia (Figura 2.2). Durante la Guerra Hispano-estadounidense en 1896,

utilizaron la práctica de vigilancia aérea equipando una cámara a un cometa, teniendo así la primera fotografía de reconocimiento aéreo.

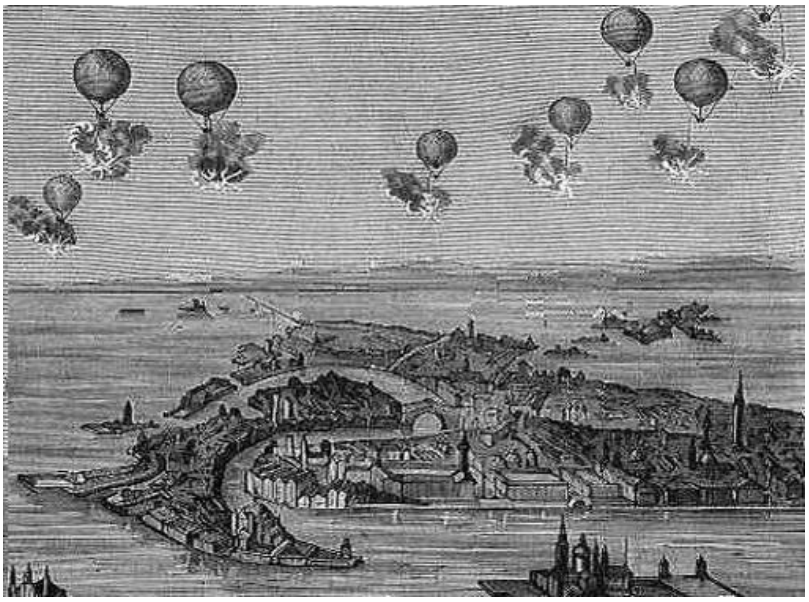


Figura 2.2. Representación del ataque austríaco en Venecia [2]

En 1898, Nikola Tesla inventó “El Teleautomatón” (Figura 2.3), un vehículo naval capaz de moverse y ser controlado mediante señales de radio, sentando las bases para futuras aplicaciones militares y robóticas [2].

Figura 2.3. Teleautomatón de Nikola Tesla

Los primeros drones, conocidos en aquel entonces como "aeronaves dirigidas por radio" o "torpedos aéreos", se originaron durante la Primera Guerra Mundial. Los ejércitos buscaban formas innovadoras de superar las trincheras y las líneas

defensivas enemigas. En 1917, una de estas innovaciones fue el desarrollo del "Kettering Bug", una de las primeras aeronaves no tripuladas diseñada para transportar una carga explosiva hasta un objetivo predeterminado (Figura 2.4) [3]. Aunque este dispositivo tenía un alcance limitado y una precisión variable, sentó las bases para el desarrollo futuro de vehículos aéreos no tripulados.

Figura 2.4. Kettering Bug

Sin embargo, fue en la Segunda Guerra Mundial cuando los drones experimentaron un desarrollo significativo. Se utilizaron como blancos móviles para entrenar a artilleros antiaéreos y como vehículos de reconocimiento. Los Estados Unidos, por ejemplo, adaptó aviones no tripulados como el "Radioplane OQ-3" para fines de entrenamiento.

El término “drone” del inglés, usado como “avión o barco no tripulado que puede navegar de forma autónoma”, fue documentado por primera vez en 1946 en una revista estadounidense llamada “Popular Science”. El significado original de esta palabra es “zángano” (macho de la abeja reina) y fue aplicado a las aeronaves no tripuladas debido al ruido que producían muy similar al de uno de estos insectos [4].

La Guerra Fría aceleró el desarrollo y la adopción de drones, especialmente en misiones de reconocimiento. Frente a la amenaza nuclear y la necesidad de obtener información sin poner en riesgo a las tripulaciones, las superpotencias vieron el valor estratégico de estas máquinas. Aeronaves como el "Ryan Firebee", utilizado por los Estados Unidos, realizaron misiones de reconocimiento en territorio hostil, recolectando datos valiosos sin arriesgar vidas humanas.

Con la disminución de las tensiones de la Guerra Fría y el avance tecnológico en miniaturización y capacidad de procesamiento, los drones comenzaron a diversificar sus aplicaciones. En la década de 1990 y principios del 2000 (Figura 2.5), surgieron drones más pequeños y versátiles, capaces de ser utilizados en misiones civiles y comerciales [5]. Ahora, no solo son capaces de volar por sí mismos, sino también de percibir el entorno, analizarlo y tomar decisiones.

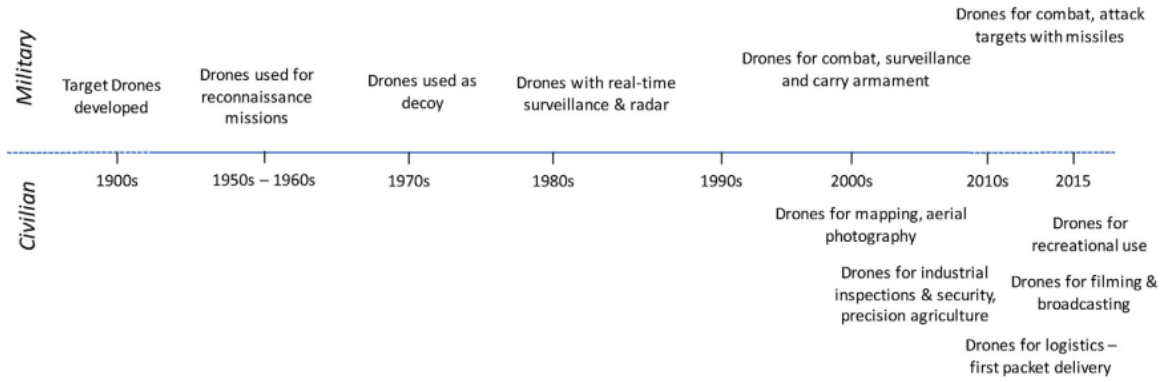


Figura 2.5. Línea temporal del uso militar y civil de los drones [5]

Esta industria ha experimentado un gran crecimiento en la última década debido a la reducción de costes y la facilidad de manejo de las tecnologías usadas. Esta evolución se ve reflejada en el aumento de fabricantes, que se vio triplicado entre los años 2006 y 2016 (Figura 2.6). En 2016, Estados Unidos y China lideraban la escena global en cuanto a la cantidad de entidades dedicadas al diseño y manufacturación de drones civiles de alto calibre, conformando aproximadamente una tercera parte del total mundial. Por su parte, la Unión Europea, con Francia, Alemania y el Reino Unido en posiciones destacadas, contribuyen con más del 30% a nivel global [6].

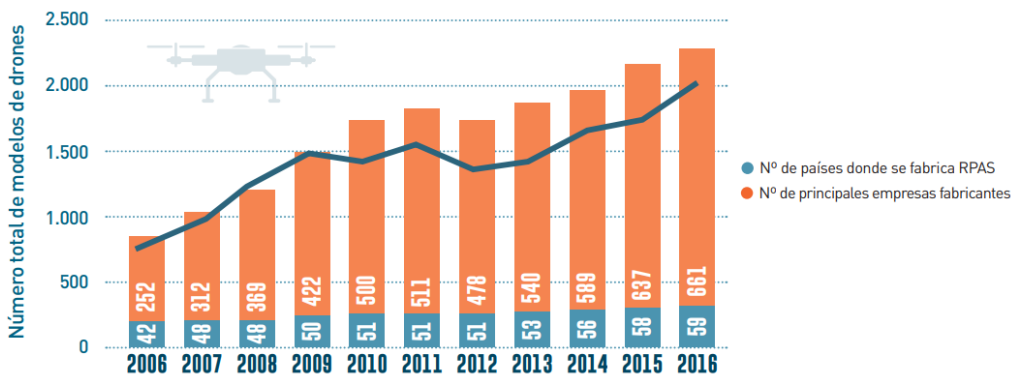


Figura 2.6. Estadísticas internacionales del sector de drones (2006-2016) [6].

En cuanto a la variedad de modelos, el año 2016 presenció la creación y producción de más de 2.000 tipos distintos de drones para aplicaciones civiles y militares. En términos de producción, el año cerró con 2,1 millones de unidades fabricadas, representando un aumento del 60.3% en comparación con el año anterior [6]. Si bien la mayoría de los drones fabricados en 2016 se destinaron al entretenimiento, se ha visto que el sector profesional ha adquirido mayor relevancia con el paso de los años. Se proyecta que la producción de drones experimentará un crecimiento exponencial en los próximos años, impulsado por la diversificación de aplicaciones civiles y comerciales.

2.2. Clasificación de los drones

No existe una clasificación exacta para los drones, ya que aspectos como tamaño, peso, velocidad, alcance, tipo de despegue, tipo de sustentación e incluso tipo de aplicaciones son algunos de los factores que los diferencian. En este caso, el criterio usado es el tipo de sustentación, que clasifica a los drones en dos grandes grupos: aerostatos y aerodinos [7] [8] [3].

2.2.1. Aerostatos

Estas aeronaves se mantienen sustentadas debido a que el gas con el que están llenos es más ligero que el aire ambiente, basándose en el *Principio de Arquímedes* [7]. No requieren de un movimiento continuo a través del aire para mantenerse elevados. Existen dos tipos principales de aerostatos basados en este método de flotación:

- Globos: Son los más sencillos en términos de diseño, se mantienen en el aire gracias al aire caliente que generan, disminuyendo su densidad y permitiendo que el globo se eleve, o con el uso de otros gases menos densos que el aire como helio. Su ventaja principal es que se pueden mantener en el aire durante largos periodos sin gastar energía en la propulsión. Sin embargo, tienen limitada maniobrabilidad (Figura 2.7).

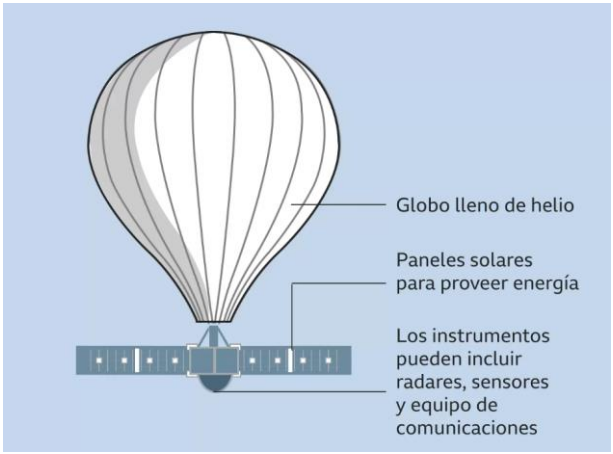


Figura 2.7. Globo no tripulado

- Dirigibles: Son vehículos aéreos más grandes y controlables que los globos. Pueden tener hélices para maniobrar y cambiar su dirección, aunque si flotación principal proviene de grandes bolsas llenas de gas.

Son ideales para tareas de monitoreo, vigilancia, investigación atmosférica y publicidad aérea, gracias a su capacidad para mantenerse estacionarios en un lugar específico por largos períodos y alcanzar una gran altitud durante el vuelo.

2.2.2. Aerodinos

Estas aeronaves, que son más densas que el aire, obtienen su sustentación mediante fuerzas aerodinámicas por el movimiento relativo entre el aire y sus superficies sustentadoras, basándose en el *Principio de Bernoulli* [7]. Podemos distinguir entre los siguientes tipos:

- Ala rotatoria: Estos drones utilizan hélices impulsadas por motores o rotores para generar la sustentación necesaria, aprovechando las fuerzas aerodinámicas de las palas giratorias (Figura 2.8). Existen varios tipos que se clasifican en función del número de hélices: helicóptero, tricóptero, cuadricóptero, hexacóptero, octocóptero, etc. Los más comunes en el mercado de los drones son los cuadricópteros (cuatro hélices), y a mayor número de hélices, mayor estabilidad y capacidad de carga.

Figura 2.8. Ejemplos de UAVs de ala rotatoria

- Ala fija: Estos drones tienen un diseño similar al de los aviones tradicionales, con alas estáticas (Figura 2.9). Son más eficientes en términos de alcance, autonomía y velocidad que los multirotores, pero requieren de grandes pistas para despegar y aterrizar.

Figura 2.9. Ejemplos de UAVs de ala fija

- Híbridos: Combina características de multirotores y drones de ala fija, ofreciendo la versatilidad de despegues y aterrizajes verticales junto con un vuelo eficiente en línea recta. Los UAVs híbridos son también conocidos como VTOL, del inglés “Vertical Take Off and Land” (Figura 2.10).

Figura 2.10. Dron WingtraOne (VTOL) [9]

2.3. Aplicaciones

Aunque los drones tuvieron su origen en misiones militares, su importancia ha crecido exponencialmente en el sector civil en los últimos años. Este crecimiento y expansión no solo ha llevado a un auge en la innovación y el desarrollo de nuevas tecnologías, sino que también ha creado una diversidad de aplicaciones prácticas. Estas aplicaciones se pueden clasificar en dos categorías amplias dentro del sector civil: las que están integradas dentro de distintas industrias [6] y las que quedan fuera de las industrias, desde usos recreativos hasta iniciativas de investigación y desarrollo.

2.3.1. Sector industrial

A continuación, veremos como las industrias están aprovechando los drones para optimizar procesos, mejorar la seguridad y aumentar la eficiencia (Figura 2.11):

- Agricultura: seguimiento del crecimiento de los cultivos, gestión del riego, detección de características del terreno y mejora de trabajos con maquinaria (fumigación).
- Obra civil: levantamientos topográficos 3D, control de excavaciones, seguimiento del avance de la construcción y vigilancia ambiental.
- Mantenimiento: mantenimiento preventivo y correctivo, inspección de zonas poco accesibles en infraestructuras (torres de

telecomunicaciones, líneas eléctricas, parques eólicos, puentes, edificios, etc.), inventario de activos.

- Minería: levantamientos topográficos 3D, control de apertura de frentes de excavación, zonas poco accesibles y monitorización de acopios.
- Logística: transportes, entrega rápida de paquetería y gestión de inventarios.
- Seguridad: prevención de accidentes, seguridad pública, respuesta a emergencias, búsqueda y rescate.
- Cinematografía y periodismo: filmación desde un amplio rango de altura, posibilidad de varias perspectivas, más silenciosos y versátiles que otros medios.
- Seguros: Inventario de activos a asegurar, seguimiento en la prevención de fraudes, caracterización de entorno para medición de riesgo potencial y evaluación de daños sufridos.

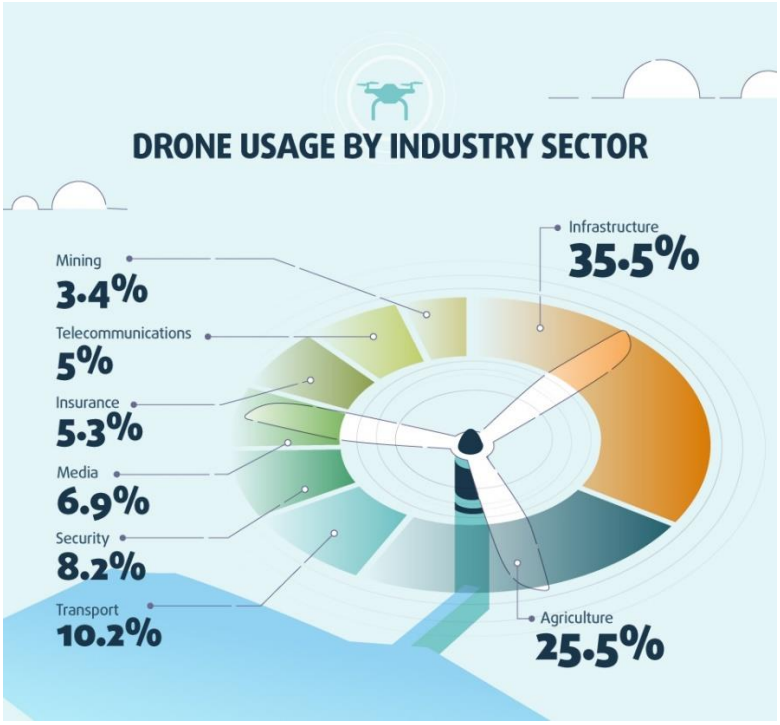


Figura 2.11. Uso de drones en el sector industrial [10].

El sector de los drones está en constante evolución, impulsado por la investigación y el desarrollo de nuevas tecnologías. Este campo dinámico no solo está perfeccionando los modelos existentes, sino que también abre la puerta a aplicaciones sin precedentes que podrían cambiar nuestro futuro. Un claro ejemplo

es el caso del dron Drangonfly (Figura 2.12), que será enviado en 2027 por la NASA para explorar Titan, la luna más grande de Saturno [11].

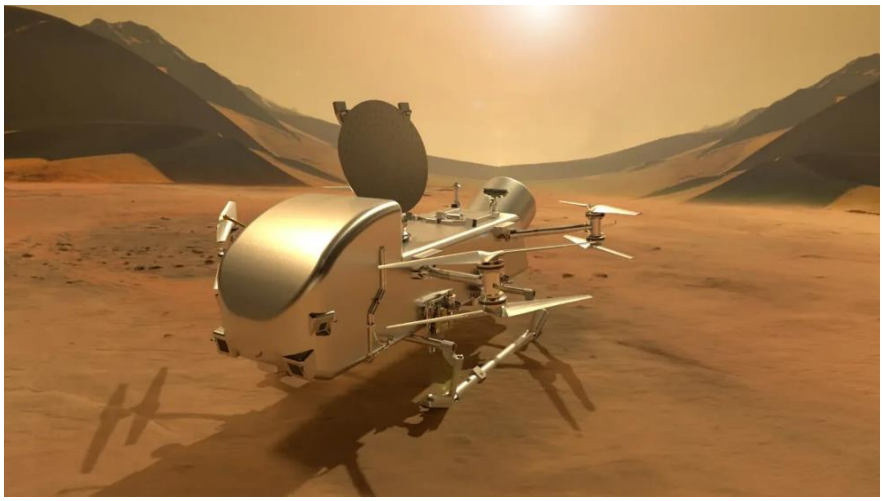


Figura 2.12. Dragonfly [11].

2.3.2. Sector servicios

Además de su indiscutible utilidad en el sector industrial, los drones han encontrado un lugar en nuestra vida cotidiana, abriendo posibilidades antes inimaginables. Veamos cómo los drones enriquecen nuestras actividades fuera del entorno industrial:

- Ámbito Recreativo: fotografía aérea, filmación, carreras de drones y espectáculos de luces.
- Investigación: recopilación de datos en estudios ambientales y arqueológicos, y búsqueda de nuevos desarrollos en los drones.
- Medioambiente: observación de plantas y animales, detección de vertidos, control de actividades no permitidas, cartografía de masas forestales, extinción y evaluación de incendios forestales.
- Educación: herramientas didácticas innovadoras y como parte de programas de formación tecnológica.
- Desastres naturales y no naturales: prevención de desastres. servicios humanitarios, entrega de suministros médicos, mapeo de áreas afectadas por desastres.

La docencia es una de las áreas donde poco a poco se están introduciendo estos dispositivos, demostrando ser poderosas herramientas educativas. Su facilidad

de uso y programación los convierte en herramientas ideales para la exploración práctica de campos como la robótica, la visión por computador y la programación en entornos del mundo real, lo que permite a los estudiantes ver inmediatamente los resultados de sus trabajos.

En la educación primaria y secundaria, los drones son una excelente manera de despertar el interés por la ciencia, la tecnología, la ingeniería y las matemáticas (STEM) [12]. Los estudiantes pueden aprender sobre aerodinámica, control remoto y programación básica de una manera interactiva y atractiva. Además, los drones promueven el desarrollo de habilidades importantes como la resolución de problemas, el trabajo en equipo y la creatividad.

A nivel universitario y en los centros de investigación, los drones se utilizan para experimentos y proyectos más avanzados. Por ejemplo, los estudiantes de ingeniería pueden diseñar y construir sus propios modelos de drones, implementar y probar sus propios sistemas de control y algoritmos de navegación. Las clases de visión por computador pueden utilizar drones para recopilar datos y aplicar técnicas de procesamiento de imágenes en proyectos que van desde el mapeo hasta el reconocimiento de patrones.

Como se ha visto, las aplicaciones de los drones hoy en día son amplias y poder trabajar con ellos en un contexto educativo permite poner en práctica conceptos teóricos en el mundo real [12]. La integración de los drones en la educación no solo enriquece la experiencia de aprendizaje, sino que también prepara a los estudiantes para enfrentar futuros desafíos tecnológicos y contribuir de manera significativa en diversas industrias que ya están aprovechando estas herramientas.

3. MATERIALES Y MÉTODOS

Este capítulo describe en detalle los materiales y métodos utilizados en el desarrollo del proyecto, centrándose en el dron Tello y su integración con las herramientas de programación de Matlab. Se describirá la estructura y funcionamiento del dron, así como la implementación del paquete de soporte necesario para su control con Matlab.

3.1. Dron Tello

El dron Tello, desarrollado por Ryze Tech en colaboración con DJI, es un cuadricóptero versátil y accesible, especialmente útil en el ámbito educativo (Figura 3.1). Con un diseño ligero y compacto de apenas 80 gramos, es ideal para la enseñanza de principios básicos y avanzados de vuelo en entornos cerrados, como aulas y laboratorios [13].

Figura 3.1. Dron Tello

Este dron está equipado con una cámara que captura fotos de 5 megapíxeles y graba a una resolución de 720p. con buena calidad de imagen para su tamaño y precio, lo que la convierte en una herramienta viable para la experimentación con visión por computador. Además, el dron incorpora un procesador de imagen de Intel que estabiliza el video capturado (Figura 3.2), un aspecto crucial para análisis de imagen precisos.

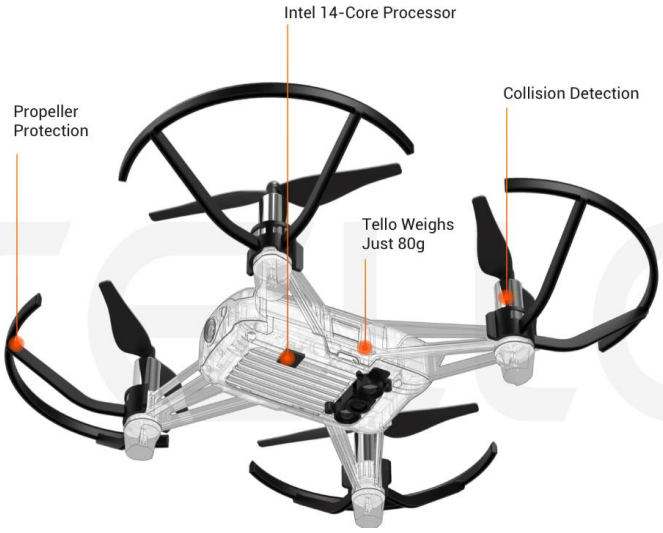


Figura 3.2. Estructura del dron Tello

El Tello SDK, proporcionado por Ryze, abre un abanico de posibilidades para la programación y control del dron mediante comandos de texto [14]. Este kit de desarrollo incluye documentación detallada, librerías, tutoriales y guías, que simplifican el proceso de desarrollo de software. Otro aspecto a destacar es su compatibilidad con Scratch, que es un lenguaje de programación visual e intuitivo, ideal para introducir a los principiantes en la programación. Estos servicios son muy útiles en un contexto educativo, donde los estudiantes pueden aprender sobre programación mientras interactúan directamente con el hardware del dron.

3.1.1. Componentes

A continuación, se detallan los componentes principales del dron Tello [15], incluyendo tanto los externos (Figura 3.3), que se aprecian en la figura, como los internos [16]. A pesar de que algunos elementos, como ciertos sensores, pueden variar dependiendo de la aplicación específica para la que se diseñe el dron, existen componentes clave que son comunes y fundamentales en la estructura de cualquier cuadricóptero. Los componentes principales son los siguientes:

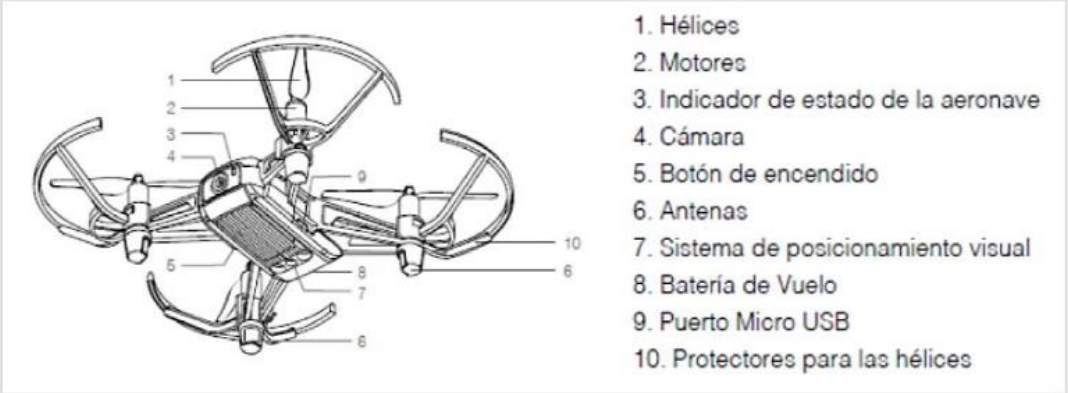


Figura 3.3. Diagrama de la aeronave [15]

- Chasis: El chasis actúa como el esqueleto del dron (Figura 3.4), proporcionando la estructura necesaria para soportar todos los componentes y asegurar un buen desempeño aerodinámico. También protege a los elementos internos sensibles de los posibles daños externos.

Figura 3.4. Chasis del dron Tello

- Motores: Son la fuente de potencia para las hélices y permiten el movimiento del dron (Figura 3.5). Para garantizar la estabilidad, los motores se dividen en pares alineados diagonalmente: un par gira en sentido horario y el otro en sentido antihorario. Si todos girasen en la misma dirección, el dron tendería a seguir ese giro perdiendo la estabilidad. Ajustando la potencia de los motores necesarios, el dron puede maniobrar en cualquier dirección y rotar sobre su eje, lo que demuestra la gran importancia de los motores y las hélices en la mecánica de vuelo.

Figura 3.5. Motores

- Hélices: Son esenciales para la sustentación, ya que los motores por sí solos no son capaces de elevar al dron. En el caso del dron Tello, las hélices cuentan con una pequeña marca que indica sobre que motor debe colocarse (Figura 3.6), asegurando que giren adecuadamente en sentido horario o antihorario. Además, la aeronave está equipada con protectores de hélices, que son cruciales para la seguridad, minimizando el riesgo de daños durante el vuelo o en caso de contacto accidental con objetos o personas.

Figura 3.6. Hélices y protectores

- Placa controladora de vuelo: Actúa como el cerebro del dron, coordinando la recepción y el procesamiento de la información proveniente del mando de control y de los sensores, así como la gestión de la alimentación eléctrica. Ejecuta los algoritmos de control de vuelo, indispensables para la estabilización y navegación del dron. Sobre esta placa (Figura 3.7) se montan también otros componentes cruciales, como los sensores, y cuenta con un gran disipador térmico adherido a los elementos que generan más calor, como el procesador, para su adecuada refrigeración.

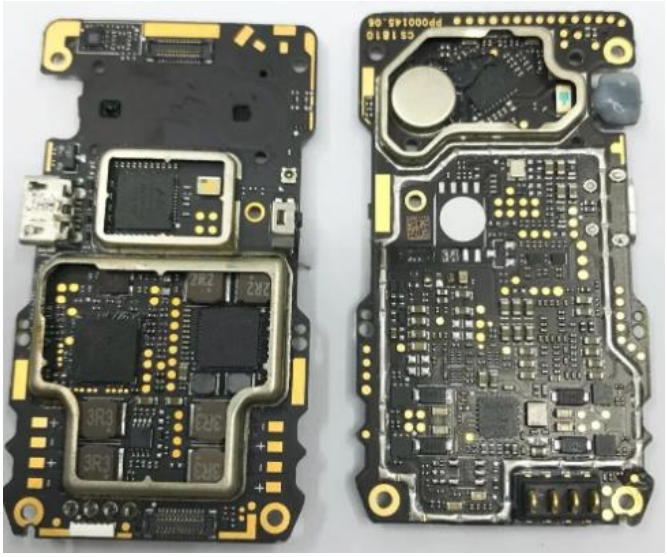


Figura 3.7. PCB con componentes electrónicos

- Controlador ESC (Electronic Speed Controller): Este componente es el encargado de modular la señal que va a los motores desde la placa principal de control.
- Antenas: permiten la comunicación inalámbrica entre el dron y el dispositivo de control, haciendo posible la transmisión de comandos y la recepción de datos de vuelo, así como la transmisión de video en tiempo real. Las antenas se encuentran en las dos patas traseras del dron.
- Batería: Proporciona la energía eléctrica necesaria para que todos los componentes electrónicos y motores funcionen. La batería de polímero de litio (Figura 3.8) ofrece una duración aproximada de 13 minutos de vuelo para el dron Tello.

Figura 3.8. Batería de dron Tello

- Sensores: el conjunto de sensores es crucial para la orientación y el vuelo autónomo del dron. Componentes como el giroscopio y el acelerómetro son esenciales para la estabilización y la orientación. Los sensores infrarrojos del dron Tello (Figura 3.9), con un alcance inferior a un metro, se utilizan para evitar obstáculos por debajo y para facilitar aterrizajes y despegues precisos. El sistema de posicionamiento visual (VPS) y el barómetro ayudan al dron a mantener su posición relativa al suelo y controlar la altitud. Por último, cuenta con la cámara principal capaz de capturar imágenes y videos con claridad.

Figura 3.9. PCB con sensores infrarrojos y VPS

3.1.2. Accesorio de visión

El Dron Tello cuenta con una cámara capaz de proporcionar una visión frontal con un ángulo de visión de 82,6°. Sin embargo, para algunas aplicaciones específicas, como en el caso del seguidor de líneas que se va a realizar en este proyecto, es necesario adaptar el campo de visión del dron para que pueda observar directamente hacia el suelo.

Para lograr esta visión cenital, se ha implementado una solución ingeniosa que consiste en colocar un espejo frente a la cámara del dron, con el objetivo de reflejar y dirigir la vista hacia el suelo (Figura 3.10). Este espejo se mantiene pegado bajo un clip especial para el dron Tello, que ha sido fabricado con impresora 3D para sostener el espejo en la posición adecuada (Figura 3.11) [16].

Figura 3.10. Clip de espejo para dron Tello

Figura 3.11. Dron Tello con espejo montado

El espejo empleado debe cumplir con requisitos ópticos precisos, siendo crucial que mantenga un ángulo de reflexión de 45°. Esto se debe a la ley de la reflexión [17], que establece que el ángulo de incidencia es igual al ángulo de reflexión (Figura 3.12). Al posicionar el espejo a 45°, los rayos de luz que inciden desde la cámara son reflejados en un ángulo de 90°, permitiendo así que la cámara capture imágenes directamente del suelo.

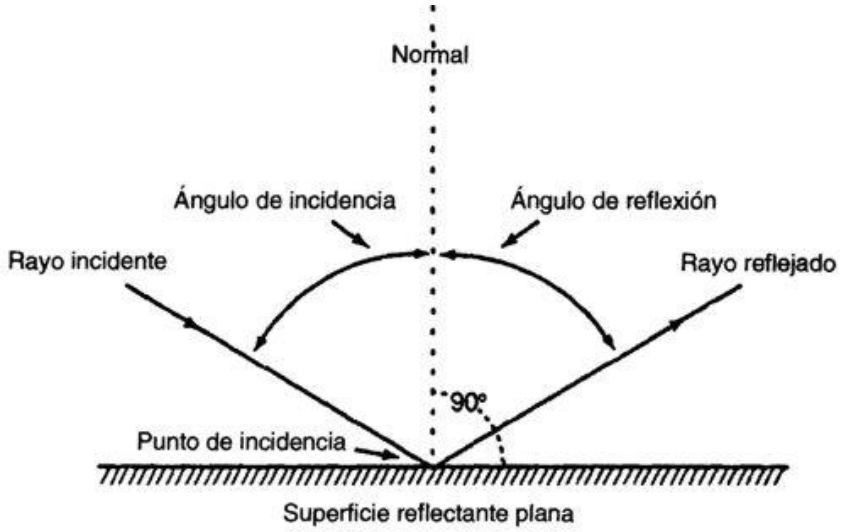


Figura 3.12. La ley de la reflexión [17]

Un inconveniente dado en este método es la inversión de la imagen captada por la cámara. En la visualización, la parte superior de la imagen corresponderá a la parte trasera del suelo observado, mientras que la parte inferior mostrará la parte frontal [17]. Este efecto es una consecuencia natural de la reflexión en el espejo, y es importante tenerlo en cuenta en la interpretación de las imágenes.

3.1.3. Modo conexión de Tello

El dron Tello es capaz de crear su propia red Wi-Fi permitiendo que cualquier dispositivo con capacidad pueda conectarse. Para que la aeronave entre en el modo conexión, solo será necesario encender el dispositivo pulsando una vez el botón que se encuentra en uno de los laterales del dron (Figura 3.13) [15].

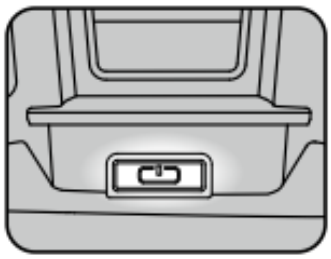


Figura 3.13. Botón de Encendido

Una vez pulsado el botón de encendido se podrá ver que el indicador de luz (Figura 3.14) comienza a encenderse en muchos colores y tras pasar unos segundos se mantiene parpadeando en color amarillo.

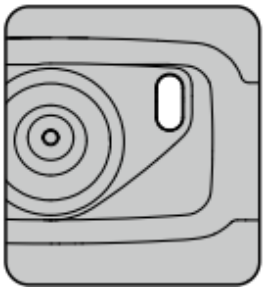


Figura 3.14. Indicador de estado de la aeronave

En este momento es cuando se podrá buscar la señal de red en los dispositivos (ordenador o smartphone) (Figura 3.15). Una vez establecida la conexión Wi-Fi con el dron, el indicador seguirá parpadeando en amarillo. A continuación, es imprescindible disponer de un entorno adecuado, como una aplicación o software específico, para enviar el comando de conexión al dron. Tras este paso, el indicador cambiará a un color verde constante, señal de que el dron está listo para ser controlado.

Figura 3.15. Red de conexión

Es importante recordar que, durante todo el proceso, mantener el dron y el dispositivo de control en un rango de distancia adecuado asegurará una conexión más estable y efectiva. Además, es recomendable usar el dron en un área libre de interferencias y con buena iluminación para optimizar la experiencia de vuelo.

3.2. Matlab

Matlab, abreviatura de “Matrix Laboratory” (Laboratorio de Matrices) es un entorno de desarrollo y lenguaje de programación creado por MathWorks, ampliamente utilizado para cálculos numéricos, análisis de datos, algoritmos y visualización gráfica. Una de las principales ventajas de Matlab es su capacidad para manejar matrices y arrays sin necesidad de dimensionarlos previamente, lo que facilita la programación y reduce el tiempo de desarrollo de algoritmos, especialmente en comparación con lenguajes como C o Fortran [18].

En entornos universitarios, es la herramienta computacional estándar para cursos introductorios y avanzados en matemáticas, ingeniería y ciencia. En la

industria, es la herramienta computacional de elección para investigación, desarrollo y análisis. Gracias a su naturaleza de alto nivel y a la gran cantidad de funciones incorporadas, permite realizar complejos cálculos numéricos y visualizar resultados con pocas líneas de código. Además, Matlab se integra fácilmente con otros lenguajes y herramientas, permitiendo el uso de bibliotecas de funciones externas.

Este entorno se complementa con una gran biblioteca de herramientas denominadas “Toolboxes”. A modo de ejemplo, “The Image Processing Toolbox” (IPT) es una colección de funciones de Matlab (llamadas M-functions o M-files) que extienden la capacidad del entorno de Matlab para la solución de problemas de procesamiento de imágenes digitales.

Matlab también ofrece la posibilidad de diseñar interfaces gráficas de usuario (GUIs) de manera intuitiva, a través de un entorno conocido como App Designer. Este entorno permite a los usuarios, incluso aquellos sin experiencia en diseño gráfico, desarrollar GUIs de manera sencilla y accesible. Es especialmente útil para hacer que herramientas y análisis complejos sean fácilmente manejables por usuarios que no son programadores.

A continuación, se exploran en detalle las herramientas que Matlab ofrece para el desarrollo del proyecto. Para ello hay que centrarse especialmente en App Designer, explorando su funcionalidad y aplicaciones prácticas, y en el paquete de soporte para el dron Tello.

3.2.1. App Designer

App Designer es un entorno de desarrollo integrado (IDE) diseñado para simplificar la creación de GUIs, arrastrando y soltando componentes sobre la interfaz [19]. La gran ventaja que presenta este entorno es que permite la edición visual del GUI, sin necesidad de programar su código. El código es generado automáticamente mediante programación orientada a objetos, reflejando el diseño de cada componente visual de acuerdo a como lo haya caracterizado el usuario.

Al iniciar un nuevo proyecto en App Designer (Figura 3.16), el usuario se encuentra con la vista de diseño, dedicada a crear la interfaz. Está vista se estructura en tres áreas principales:

- A la izquierda, se dispone de una amplia biblioteca de componentes visuales que pueden ser empleados para ensamblar la interfaz de usuario: gráficos, diversos tipos de botones, etiquetas y tablas entre muchos otros componentes.
- El área central de la pantalla está dedicada a la vista de la interfaz. Los componentes seleccionados se arrastran a esta zona, en la que se puede editar su tamaño y posición.
- A la derecha, se sitúa un navegador de componentes que permite a los usuarios visualizar todos los elementos añadidos a la interfaz, acceder a opciones de configuración más avanzadas y asignar funciones específicas a cada componente.

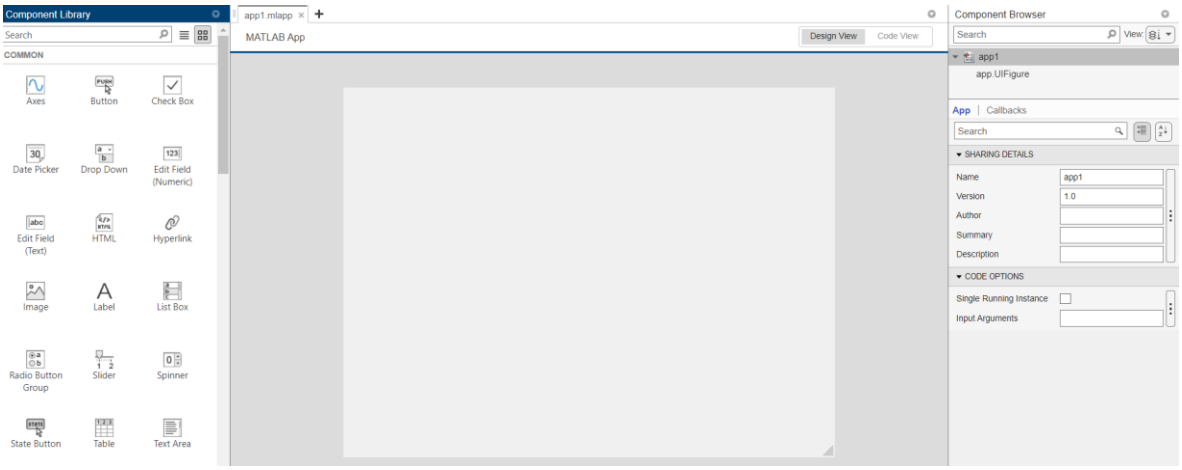


Figura 3.16. Nueva plantilla en App Designer

En la siguiente Figura 3.17 se muestra de nuevo el entorno tras añadir varios componentes a la interfaz arrastrándolos desde la biblioteca. Ahora se puede observar en el navegador como se muestran los cuatro componentes añadidos. En este caso se encuentra marcado el botón y en la parte inferior del navegador aparecen todas las características que se pueden modificar para este elemento, como fuente, el color de la letra, la interacción, la posición, funciones asociadas al botón o identificadores.

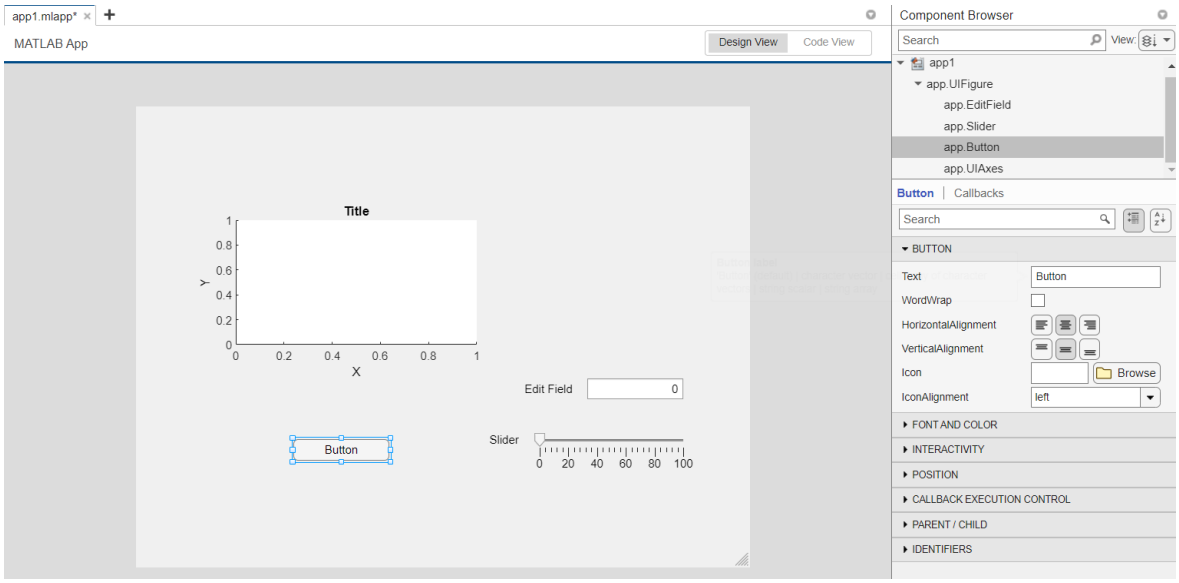


Figura 3.17. Plantilla de App Designer con componentes

La vista de código (Figura 3.18), visible en otra sección, muestra todo el código que el entorno genera automáticamente tras crear y modificar cualquier componente en la vista de diseño. Este código sobre el diseño de los componentes se encuentra bloqueado para modificaciones directas, solo se permite crear nuevas variables y funciones dedicadas los eventos que estos componentes puedan realizar, por ejemplo, pulsar un botón. De esta forma, la programación es mucho más intuitiva, incluso para usuarios que no tengan nociones básicas de programación orientada a objetos.

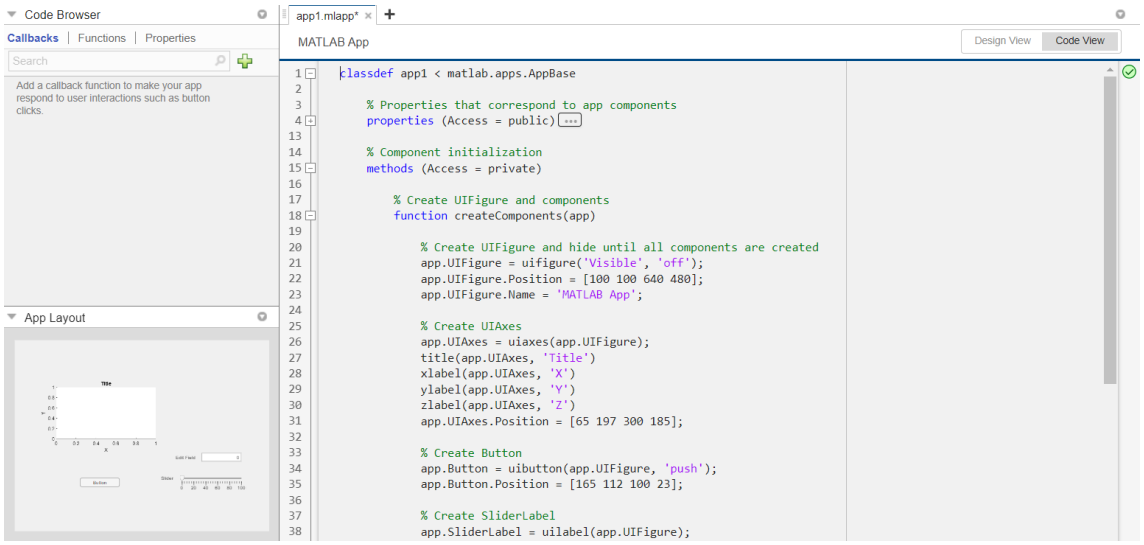


Figura 3.18. Vista de código en App Designer

Desde el navegador de código es posible crear los siguientes elementos, que permiten la programación de la interfaz:

- Callbacks: son funciones que se ejecutan cuando ocurre un evento determinado. Existen muchos tipos de eventos dentro de una aplicación, por ejemplo, iniciar o cerrar la interfaz. Sin embargo, la gran mayoría se asignan a los componentes, como pulsar un botón, editar un campo de texto, marcar una casilla o mover un deslizador.
- Funciones: son bloques de código que realizan alguna operación. Al contrario que los Callbacks, estas no son llamadas tras ocurrir un evento, sino que se utilizan para organizar código de los callbacks o como funciones auxiliares.
- Propiedades: son variables globales de la aplicación accesibles en cualquier función, sin necesidad de que sean variables de entrada. Las propiedades, al igual que las funciones, pueden ser privadas (para uso interno de la aplicación) o públicas (para compartir datos fuera de la aplicación).

Una vez creados estos elementos, el entorno de App Designer habilita la modificación de código únicamente dentro de estas funciones. En la Figura 3.19 se muestra cómo se visualizaría la vista de código al crear un callback para el evento de pulsar un botón.

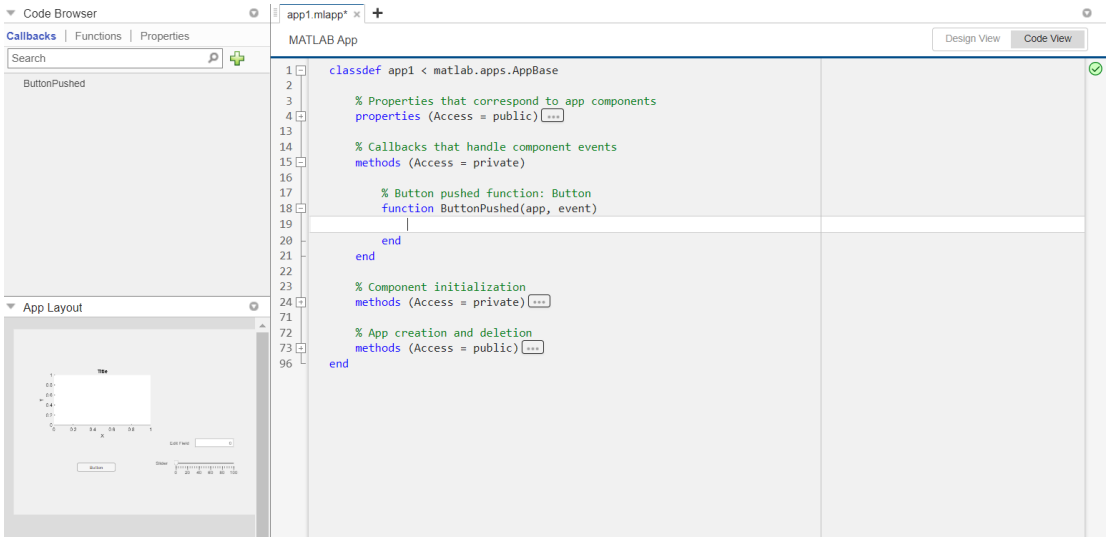


Figura 3.19. Ejemplo de Callback para pulsar un botón

Para referenciar cualquier variable externa a una función, como puede ser el caso de una propiedad o de una característica de un componente, es necesario utilizar el prefijo “app.”. Por ejemplo, en el caso de querer utilizar el valor de un

campo numérico dentro de un callback, se haría referencia a esa variable como “app.EditField.Value”.

La capacidad de App Designer para integrar funcionalidades, librerías y paquetes de soporte de Matlab lo convierte en una herramienta muy potente y versátil para el desarrollo de aplicaciones. También es posible exportar las aplicaciones creadas y compilarlas sin el uso de Matlab.

Este entorno de desarrollo se presenta como una solución ideal para el desarrollo de proyectos y prácticas en muchas asignaturas. App Designer, junto con las funciones de procesamiento de imagen y el paquete de soporte del dron Tello, traen nuevas propuestas al desarrollo de la asignatura de visión por computador.

3.2.2. Extensión para dron Tello

El "Support Package for Ryze Tello Drone" es una extensión diseñada para Matlab y Simulink que facilita la integración y control del dron Tello dentro de estos entornos de programación. Este paquete proporciona bloques y funciones que permiten una conexión directa con el dron, la transmisión y recepción de datos de vuelo, y el acceso a la transmisión de video en tiempo real [18].

Este paquete de soporte ha sido desarrollado y testeado por el equipo de hardware de Mathworks Matlab, gracias al soporte del Tello SDK. Es compatible con cualquier versión de Matlab superior a R2020a. Para utilizar las funciones que proporciona, solo es necesario descargar e instalar la extensión, disponible en el enlace.

Los comandos que ofrece esta extensión para Tello pueden agruparse en cuatro categorías: conexión y configuración, adquisición de imagen, navegación y datos de vuelo.

En primer lugar, los comandos dedicados a la conexión y configuración se muestran en la siguiente Tabla 3.1:

Función	Descripción
ryze	Conexión con el dron Tello
ryzelist	Lista de drones conectados al sistema
ryzesetup	Iniciar la configuración para trabajar con varios drones

Tabla 3.1. Comandos de conexión

Para poder realizar la conexión de Matlab con el dron, es necesario que se encuentre enlazado el ordenador a la red Wi-Fi que la aeronave proporciona, como se ha explicado en el capítulo anterior (**¡Error! No se encuentra el origen de l a referencia.**). Una vez conectada la red, se utiliza el comando "ryze" para establecer la conexión con el entorno y tomar el control del dron. Es necesario asignar una variable al ejecutar este comando, ya que hay que identificar al dron cuando se envíen los demás comandos de movimiento. Como se puede observar en la Figura 3.20, se ha utilizado la variable "dron" y contiene los identificadores y propiedades que se muestran.

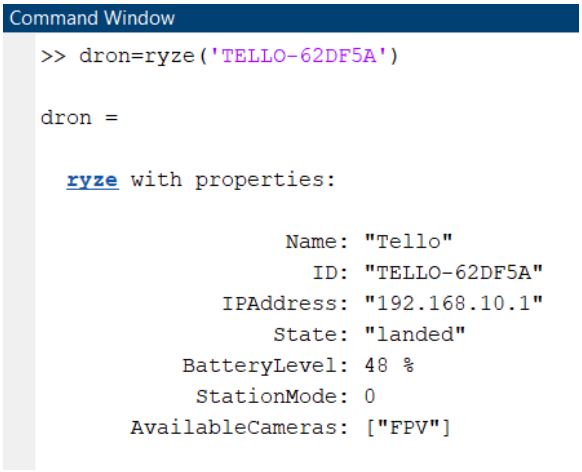


Figura 3.20. Conexión con el dron Tello en Matlab

En segundo lugar, se encuentran los comandos de adquisición de imagen. Estos incluyen la conexión con la cámara, la captura de imágenes y la transmisión de video (Tabla 3.2). El comando "camera" activa la cámara del dron, guardando sus propiedades en una variable, que será utilizada a la hora de realizar capturas o visualización de video.

Función	Descripción
camera	Conexión con la cámara
preview	Vista de video en vivo
snapshot	Adquiera una captura de imagen de la cámara

Tabla 3.2. Comandos de adquisición de imagen

Las funciones de navegación, pueden dividirse en comandos básicos (despegue y aterrizaje) y en comandos avanzados (movimientos en todas direcciones, rotaciones e incluso voltear el dron). Los comandos básicos de vuelo solo necesitan como argumento de entrada la variable que identifique al dron (Tabla 3.3).

Función	Descripción
takeoff	Iniciar el despegue del dron Ryze
land	Iniciar el aterrizaje del dron Ryze
abort	Fin de vuelo del dron Ryze

Tabla 3.3. Comandos básicos de navegación

Las funciones avanzadas (Tabla 3.4) añaden otros argumentos de entrada para definir el movimiento. En caso de los desplazamientos, por defecto, se asigna una distancia de 0.2 m y una velocidad de 0.4 m/s, pero es posible indicar como argumentos de entrada la velocidad (entre 0.1 m/s y 1 m/s), la distancia (entre 0.2 m y 5 m) o solo indicar la duración del movimiento. Por otro lado, en el caso del giro se indicará el ángulo como argumento de entrada y para voltear el dron se indicará la dirección.

Función	Descripción
move	Mover el dron Ryze en los tres ejes
moveforward	Mover el dron Ryze hacia adelante
moveback	Mover el dron Ryze hacia atrás
movedown	Mover el dron Ryze hacia abajo
moveup	Mover el dron Ryze hacia arriba
moveleft	Mover el dron Ryze a la izquierda
moveright	Mover el dron Ryze a la derecha
turn	Girar el dron Ryze en algún ángulo específico
flip	Voltear el dron Ryze en una dirección específica

Tabla 3.4. Comandos avanzados de navegación

Por último, se encuentran los comandos utilizados para obtener datos del vuelo. Los tres comandos que se muestran en la Tabla 3.5 miden: la altura, orientación y velocidad del dron.

Función	Descripción
readHeight	Leer la altura actual del dron Ryze respecto a la superficie de despegue
readOrientation	Leer la orientación actual del dron Ryze en términos de ángulos de Euler
readSpeed	Leer la velocidad actual del dron Ryze

Tabla 3.5. Comandos de datos de vuelo

4. DESARROLLO SOFTWARE: SEGUIDOR DE LÍNEA

El objetivo de esta sección es desarrollar un proyecto que integre las capacidades del dron Tello con conceptos de visión por computador. La idea elegida para el desarrollo es la de un dron seguidor de línea. En este caso, se pretende que el dron logre una navegación autónoma detectando en todo momento los parámetros de la línea que le ayuden a realizar el seguimiento. Por lo tanto, la visión del dron toma un papel esencial dentro de este proyecto.

En primer lugar, es necesario entender como se ha planteado el escenario, para explicar los diferentes procesos. Se parte de una línea gruesa formada por cartulinas negras pegadas en el suelo sin un diseño definido, es decir, puede conformar o no un circuito cerrado, con curvas no muy pronunciadas (Figura 4.1). La navegación autónoma del dron sobre esta línea implica que debe mantenerse siempre lo más centrado posible y rotar cuando sea necesario, como puede ser en el caso de una curva o desviación de la línea.

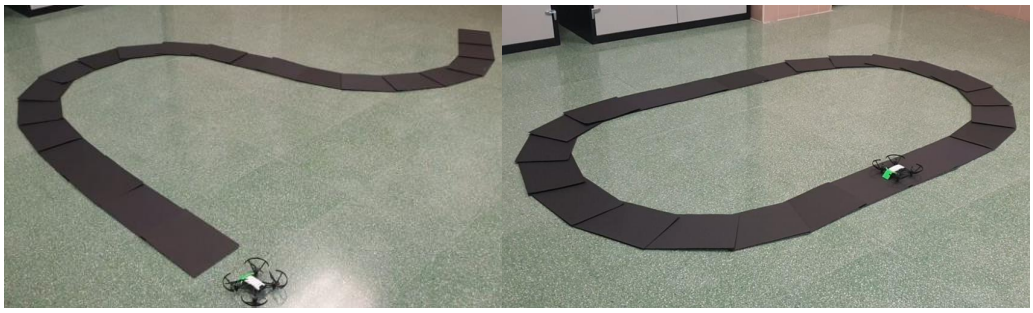


Figura 4.1. a: Línea abierta; b: Línea cerrada

Otro aspecto crucial en el escenario es mantener una iluminación adecuada. En primer lugar, el dron Tello utiliza un sistema de posicionamiento visual que requiere una iluminación óptima para mantener su posición en el aire sin desviaciones. En segundo lugar, para la captura de imágenes de la línea, es esencial contar con una iluminación uniforme y sin sombras ni reflejos, ya que esto mejora el contraste entre los colores del suelo y la línea.

Durante el vuelo, se realizan dos procesos principales: el procesamiento de la imagen y el sistema de navegación. A través de este desarrollo, se busca proporcionar al dron la capacidad de seguir una línea, utilizando la información visual adquirida para realizar ajustes de posición y orientación en tiempo real. Estos

procesos se ejecutan de manera continua en bucle, de forma que el dron toma la imagen, la procesa y a continuación se indican las ordenes de movimiento (Figura 4.2).

Figura 4.2. Flujograma principal del proceso

En el caso de una línea abierta, con un punto de inicio y final, el dron comienza siguiendo la línea y finaliza el proceso al llegar al final de la misma. Sin embargo, en el caso de un circuito cerrado, es necesario establecer una orden externa de parada para el dron ya que realizaría el circuito continuamente hasta agotar la batería.

Para llevar a cabo el procesamiento de la imagen es necesario determinar qué datos se necesitan extraer de la línea para trabajar en los movimientos. Inicialmente se consideró centrarse únicamente en la obtención del centroide de la línea. Sin embargo, este planteamiento tenía ciertas limitaciones al enfrentarse a curvas, ya que resulta difícil determinar el ángulo de rotación efectivo. Finalmente,

se adoptó un planteamiento para el sistema de navegación basado en el uso del centroide y la orientación de la línea. Este enfoque permite realizar un desplazamiento lateral en caso de que el error entre el centro de la imagen y el centroide sea significativo. Además, se considera la orientación de la línea con respecto a los ejes de la imagen para corregir el ángulo, ya sea durante un giro o desviación de la línea.

4.1. Procesamiento de imagen

El proceso consta de tres etapas fundamentales (Figura 4.3): adquisición de imagen, preprocesamiento y extracción de características [19]. En la etapa de adquisición, se captura la imagen desde la cámara del dron. En la etapa de preprocesamiento, la imagen se procesa para obtener únicamente la línea en una representación binarizada. A partir de esta nueva imagen, se lleva a cabo la extracción de las características necesarias de la línea, las cuales se envían al sistema de navegación para que tome decisiones sobre el movimiento del dron.

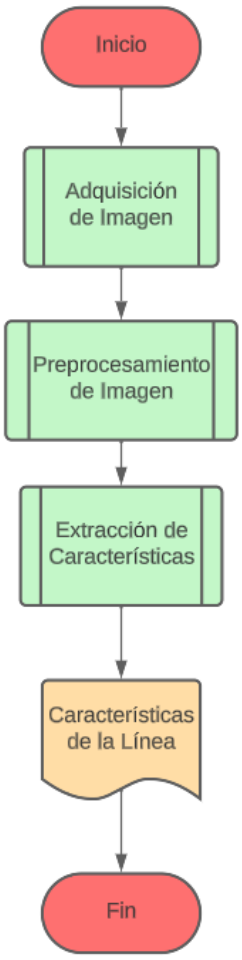


Figura 4.3. Flujograma del procesamiento de imagen

4.1.1. Adquisición

Para este proceso se utiliza la cámara principal del dron Tello, que es la cámara frontal. Aunque existe otra cámara orientada hacia el suelo, esta tiene menor resolución y se destina principalmente al sistema de posicionamiento visual. Para que la cámara frontal pueda dirigir la vista hacia el suelo, se utiliza el accesorio mencionado en el capítulo anterior, el clip de espejo.

La captura de imagen se realiza con la función “snapshot” de la extensión del dron Tello para Matlab. En la Figura 4.4 se muestran algunos ejemplos de las imágenes captadas por el dron utilizando el espejo.

Figura 4.4. Imágenes obtenidas con el dron Tello

4.1.2. Preprocesamiento

Una vez obtenida la captura se procede a realizar transformaciones en la imagen para la obtención de la línea. Durante este proceso, se aplican funciones que pertenecen a “Image Processing Toolbox” de Matlab, una herramienta muy utilizada en visión por computador [20].

En primer lugar, se considera que la imagen que se ha obtenido se encuentra invertida verticalmente debido al uso del espejo. Para ello se aplica la función “flip”, que permite la visualización real de la línea (Figura 4.5).

Figura 4.5. a: Original; b: Invertida

El siguiente paso implica la transformación de la imagen en formato RGB a escala de grises mediante la función “rgb2gray” (Figura 4.6).

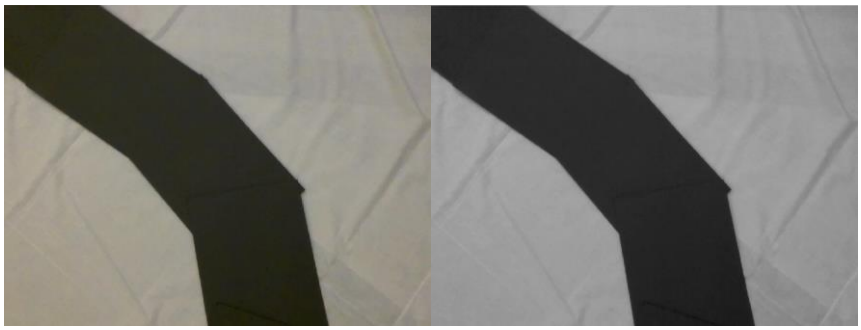


Figura 4.6. a: Original; b: Escala de grises

El siguiente paso consiste en binarizar la imagen, es decir, a cada pixel de la imagen se le asigna el valor de 0 (negro) o 1 (blanco) (Figura 4.7). Esta operación se realiza con la función “im2bw”, para la cual hay que aportar un umbral que determine a partir de qué nivel de gris se asigna el valor 1 o 0. Al utilizar el color negro en la línea, se simplifica este proceso, ya que, escogiendo un umbral bajo de gris, se logra obtener toda la línea. Sin embargo, cabe la posibilidad de que se incluyan otros píxeles negros fuera de la línea debido a sombras. En caso de optar por una línea de color blanco, podrían surgir otras complicaciones como posibles reflejos en el suelo que dificulten este proceso.

Figura 4.7. a: Original; b: Binarizada

Una vez realizada la binarización, se debe tener en cuenta que en visión por computador se trabaja siempre con los píxeles blancos. Por ello, se aplicará una función llamada “imcomplement” que obtiene la imagen complementaria, cambiando los 0 por 1 y viceversa (Figura 4.8).

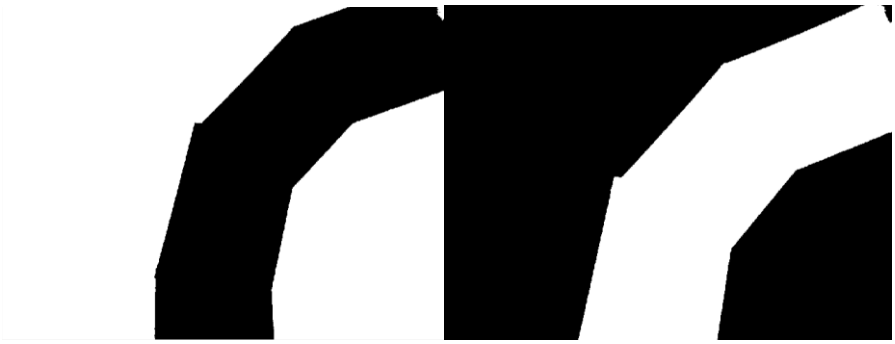


Figura 4.8. a: Binarizada; b: Complementaria

Es posible que, tras la binarización, queden píxeles sueltos que no formen parte de la línea, lo cual podría complicar considerablemente el proceso, ya que estos píxeles sueltos podrían interpretarse erróneamente como otros objetos dentro de la imagen. Para abordar este problema, se emplea la función “bwareaopen”, que elimina los objetos dentro de una imagen en función de la cantidad de píxeles conectados, ayudando a eliminar los posibles grupos de píxeles detectados fuera de la línea, como aquellos creados por sombras. De esta manera, se mejora la precisión del proceso al eliminar elementos no deseados que podrían interferir con la detección adecuada de la línea.

Finalmente, para completar la obtención de la línea se realiza un proceso de relleno de píxeles utilizando la función “imfill”. A pesar de que la línea es completamente negra, este paso asegura que cualquier zona de píxeles que podría haberse definido incorrectamente como negra debido a reflejos o una iluminación mayor se rellene correctamente para la extracción de características.

4.1.3. Extracción de características

Una vez completado el preprocesamiento de la imagen y obtenida una representación donde solo está presente la línea, es el momento de extraer las características necesarias para controlar la posición del dron. Como se mencionó al inicio de este capítulo, las dos características clave que se buscan son el centroide y el ángulo de orientación de la línea.

En Matlab, existe una función denominada "regionprops" que, dada una imagen binarizada con diversas regiones, puede extraer las propiedades de cada región. Esta función ofrece una extensa lista de propiedades, y entre ellas se

encuentra la propiedad "Centroid", que devuelve el centro de masas de la región en forma de un vector con las coordenadas en píxeles de la imagen (Figura 4.9).

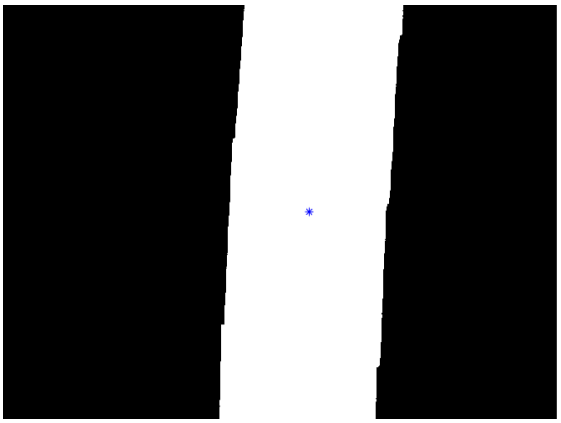


Figura 4.9. Imagen con centroide representado

Por otro lado, existe una propiedad definida para esta función llamada “Orientation”, que devuelve el ángulo de orientación de la región. Se define el ángulo que forma el eje mayor de la elipse que contiene los mismos segundos momentos que la región definida, con respecto al eje X. Como se observa en la Figura 4.10, mediante la elipse definida, se obtiene el ángulo utilizando su eje mayor.

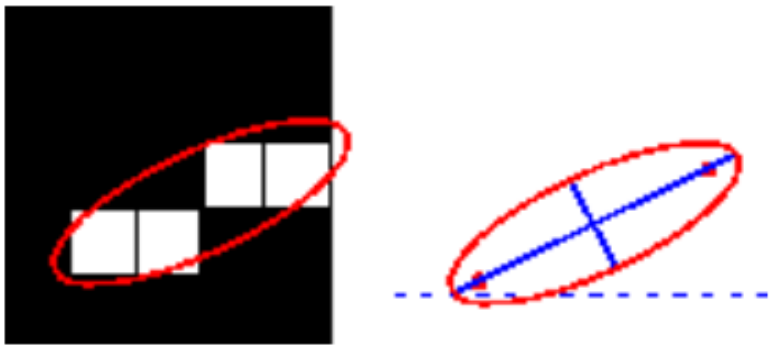


Figura 4.10. Representación de la orientación de un objeto

4.2. Sistema de navegación

El segundo proceso, el sistema de navegación, utiliza la información obtenida en el procesamiento de la imagen para realizar ajustes en tiempo real en la posición y orientación del dron. En una primera etapa, se interpreta la información extraída de las características de la línea para determinar la posición relativa de la línea con respecto al centro de la imagen del dron. Una vez calculados estos errores, los valores resultantes se asignan a variables que actuarán como argumentos de entrada para las órdenes de movimiento del dron (Figura 4.11).

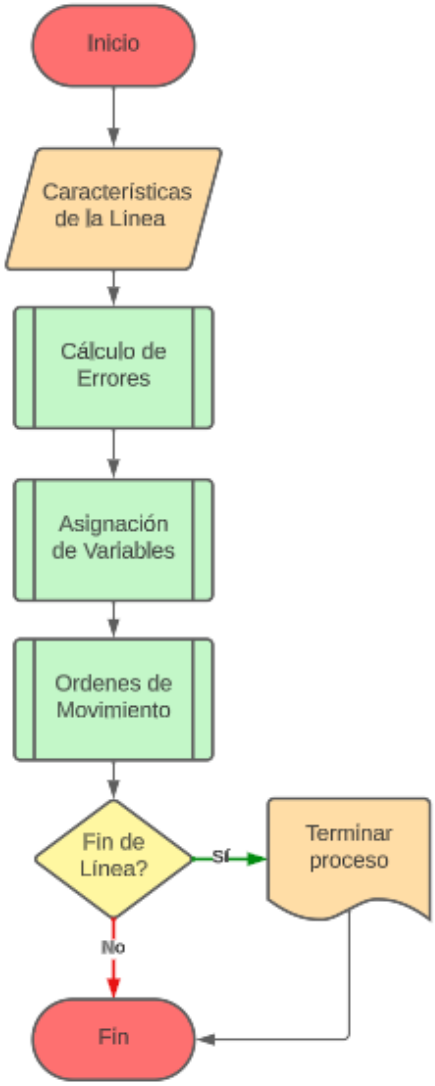


Figura 4.11. Flujograma del sistema de navegación

4.2.1. Cálculo de errores y asignación de variables

En este proceso se realizan los cálculos necesarios para determinar la posición relativa del centroide de la línea con respecto al centro de la imagen y el ángulo con respecto al eje Y del dron.

Para calcular el error de posición, es fundamental conocer las coordenadas del centro de la imagen. El origen de coordenadas de los píxeles se encuentra en la esquina superior izquierda de la imagen y por tanto las coordenadas del centro corresponderán a los valores medios del ancho y largo de la imagen (Figura 4.12). La resolución de la imagen puede obtenerse con la función “size”. Una vez conocidos estos valores se podrá calcular el error de posición en los ejes X e Y.

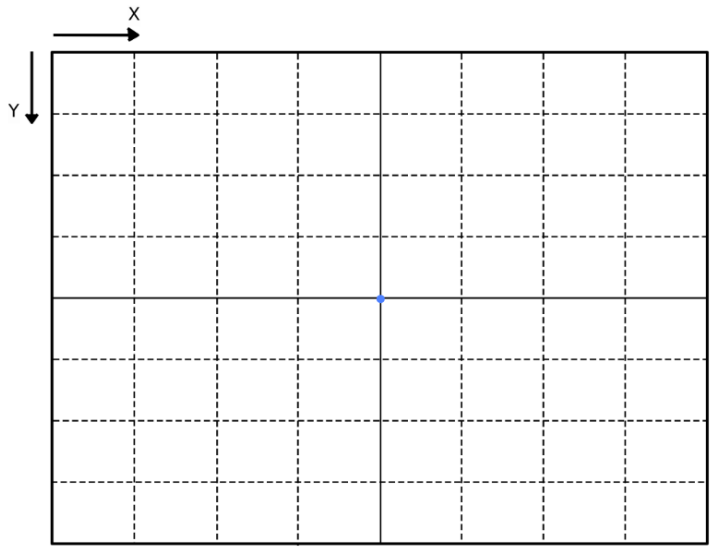


Figura 4.12. Ejes de coordenadas de la imagen

El objetivo es que el dron siempre se desplace en el eje Y mientras se detecte la línea, y para lograr esto, se asigna la distancia mínima requerida para realizar un movimiento con el comando “move”, que es de 0.2 metros. Por tanto, la distancia no dependerá del error en el eje Y, este error de posición se utiliza más adelante para conocer si comienza o termina la línea.

En cuanto al desplazamiento lateral (eje X), el error indica el número de píxeles que separa al centroide de la línea del centro de la imagen. Sin embargo, no se puede asignar con certeza una distancia para rectificar el error en el eje X, debido a que no existe una relación entre el ancho de píxeles de la imagen y la distancia real. Esta relación podría calcularse con la ecuación de la lente [21], conociendo la distancia focal de la cámara y la distancia entre la lente de la cámara y el objeto, sin embargo, esto no es posible debido al desconocimiento del último dato. Como solución, se han realizado diferentes mediciones con la altura media a la que el dron realiza el vuelo, estimando que la distancia real observada por el dron en el eje X es de 0.8 metros de longitud (Figura 4.13). Con esta medición, junto con la longitud de la imagen en píxeles, se obtiene un factor de conversión para poder indicar la distancia que se debe corregir, siempre que supere el mínimo de 0.2 metros.

Figura 4.13. Calibración de la longitud real de la imagen

Además de corregir la posición, se ajusta la orientación con respecto a la línea. El ángulo obtenido inicialmente, es el que conforma la línea con el eje X, sin embargo, es necesario calcular el ángulo con respecto al eje Y, es decir, el ángulo complementario. Este ángulo será asignado a la variable que se use como argumento en la función que realiza el giro, “turn”.

4.2.1. Órdenes de movimiento

En este apartado se define como se ejecutan los movimientos del sistema de navegación, utilizando la función “move” para los desplazamientos y la función “turn” para los giros.

El paquete de soporte del dron Tello para Matlab presenta diversas restricciones en sus movimientos. En primer lugar, se debe definir la distancia que se quiere realizar en los movimientos, lo cual puede limitar la capacidad de respuesta rápida ante un cambio de la línea. Además, la distancia mínima permitida para realizar un movimiento es de 0.2 metros. La última restricción es que no se pueden ejecutar otros comandos de movimiento mientras está en curso uno actual, lo que hace el proceso menos fluido al tener que esperar a que finalice.

Para desarrollar el sistema de ordenes de movimiento hay que tener en cuenta todas las posibilidades relacionadas con la línea. Esto incluye el inicio o y final de la línea en caso de que no forme un circuito cerrado. También hay que tener en cuenta posibles desplazamientos laterales inesperados del dron, ya sea por corrientes de viento o algún reajuste del sistema de posicionamiento visual.

Finalmente, se debe contemplar el propio seguimiento de la línea incluyendo las curvas.

En la Figura 4.14 se muestra un mapa de los movimientos en función de la posición del centroide de la línea en la imagen. Estas son las regiones definidas:

- Inicio de la línea (amarillo): El comienzo de la línea, aparecerá en la parte superior de la imagen, es decir, el centroide se encontrará dentro de la zona amarilla. En este caso se indicará al dron que se desplace hasta llegar a la zona verde donde continua el seguimiento.
- Seguimiento de la línea (verde): Cuando el centroide se encuentra en esta zona de la imagen, quiere decir que el dron se encuentra encima de la línea. En primer lugar, se verifica que el ángulo de orientación no sea mayor a un mínimo establecido, en caso de que lo sea, se realiza el giro para corregir la orientación. En la siguiente vuelta del bucle se comprueba de nuevo el ángulo, que ya debe haberse corregido, y realiza el desplazamiento.
- Desviación lateral (azul): Si la línea se encuentra en un lateral de la imagen, se realizará un desplazamiento para llevar la línea al centro y corregir la desviación.
- Final de la línea (rojo): Al llegar al final de la línea, el centroide se sitúa en la parte inferior de la imagen. Entonces se detecta que se ha llegado al final de la misma y se finaliza el proceso.

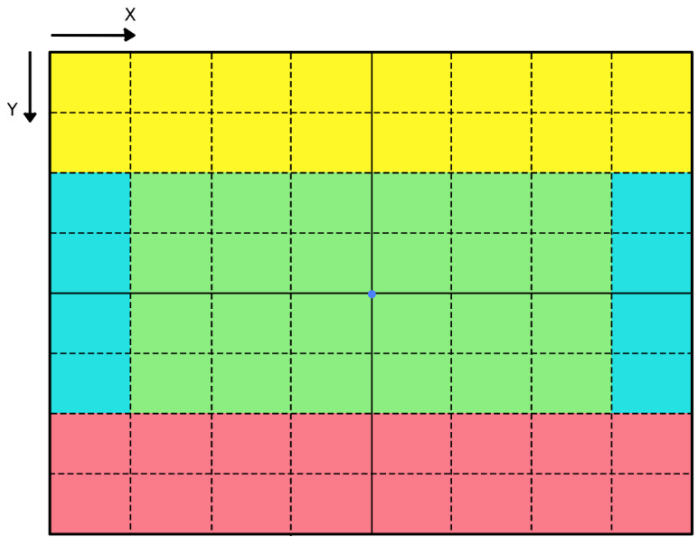


Figura 4.14. Mapa con regiones de movimiento

5. RESULTADOS

En este capítulo, se exponen los resultados obtenidos tras llevar a cabo el caso de estudio del dron seguidor de líneas. Se presenta su integración en Matlab con una interfaz de usuario desarrollada con el entorno de App Designer. La interfaz presenta los elementos necesarios para conectar con el dron, iniciar y parar el proceso, la representación de las imágenes captadas y procesadas en tiempo real, y algunos componentes extras.

Por otro lado, a partir del proceso de aprendizaje del desarrollo de este caso, se han diseñado dos guiones de prácticas, que adaptan algunos de los conceptos de visión por computador, presentados en las prácticas de la asignatura, junto con el dron Tello y el entorno de App Designer.

5.1. Interfaz: seguidor de líneas

Una vez definidas las etapas de procesamiento de la imagen y navegación del dron, es necesario unificarlo todo en un mismo punto. El uso del entorno de App Designer de Matlab ha facilitado la creación de una interfaz con la que poder desarrollar este caso de dron seguidor de líneas. En esta interfaz que se presentará a continuación (Figura 5.1), se han añadido varias funcionalidades que permiten entender de una forma clara el proceso que lleva a cabo el dron.

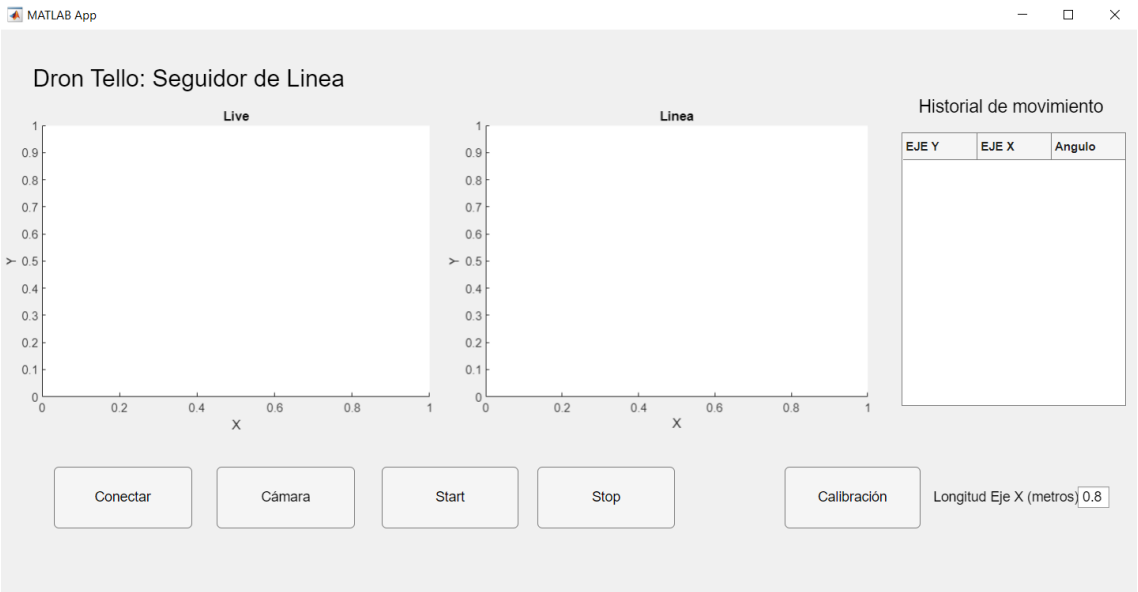


Figura 5.1. Interfaz de seguidor de línea

La interfaz contiene un conjunto de elementos que permiten la ejecución del proceso principal, que es el seguimiento de línea, además de otras funciones como la conexión con el dron, la visualización de las imágenes captadas por el dron, el proceso de calibración para la longitud real de la imagen y un historial de los movimientos realizados durante el proceso.

Al iniciar la aplicación, se establece la conexión del dispositivo con la red del dron Tello y se conecta el dron a la interfaz mediante el botón “Conectar”. Durante este proceso, se inicializan todas las variables necesarias para el funcionamiento de la aplicación y la conexión con el dron y su cámara. La conexión se ha realizado correctamente cuando en la interfaz aparecen los elementos mostrados en la parte superior derecha de la Figura 5.2, indicando la conexión y el estado de la batería en el momento de la conexión.

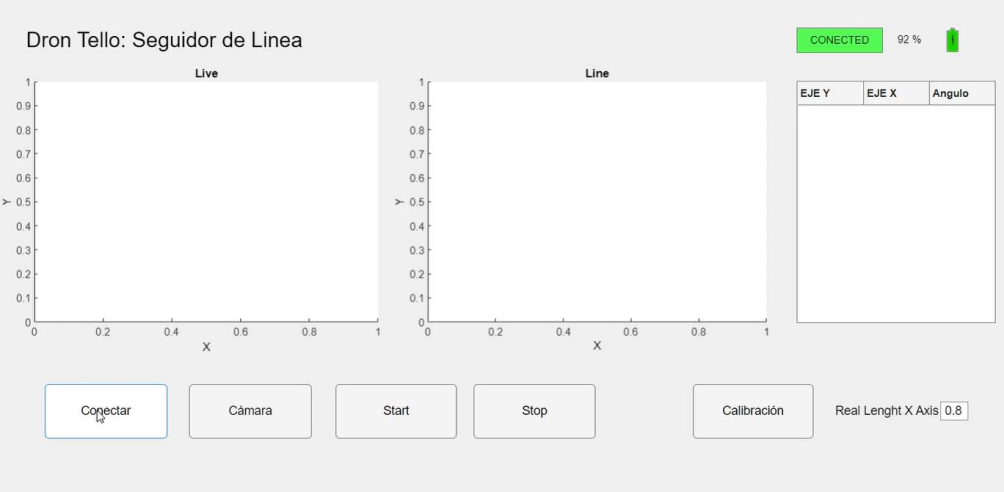


Figura 5.2. Interfaz conectada con el dron Tello

En la parte central se presentan los dos ejes donde se mostrarán las imágenes: a la izquierda, la imagen en vivo sin procesar y a la derecha, la imagen procesada mostrando la línea.

El botón “Cámara”, permite activar la visión en vivo del dron para comprobar la correcta visualización una vez colocado el accesorio del espejo (Figura 5.3).

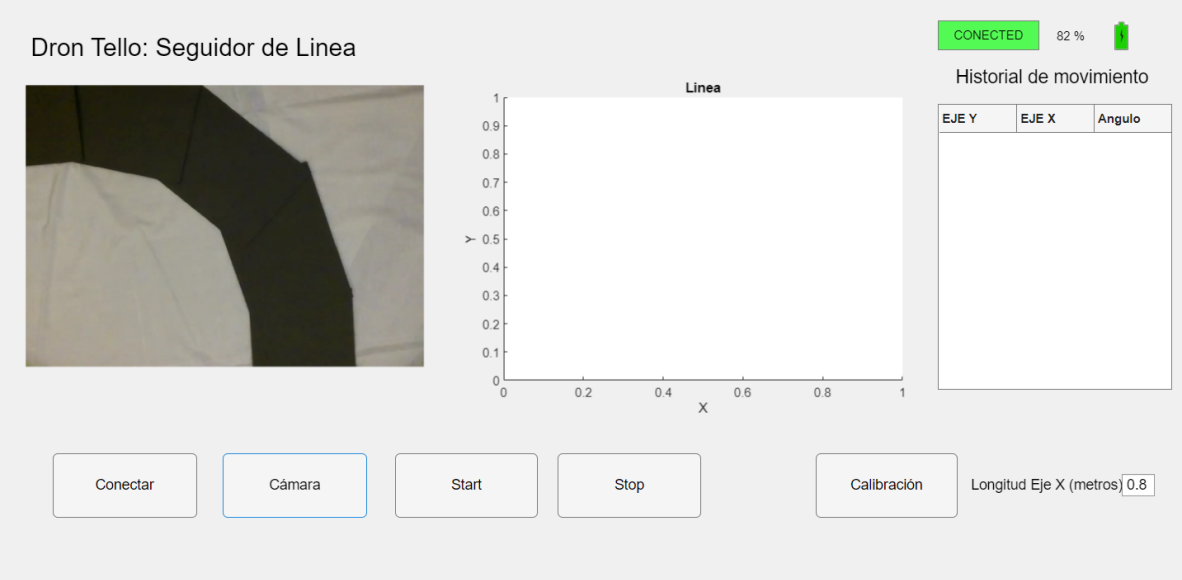


Figura 5.3. Interfaz con cámara activada

La siguiente función añadida durante el proceso de desarrollo es la calibración de la longitud de la imagen. En la parte inferior derecha de la Figura 5.4, se ha dedicado un botón para realizar un proceso de despegue y aterrizaje mostrando la visión de la cámara para comprobar la longitud real de la imagen a lo largo del eje X. En este proceso se utiliza una cinta métrica para realizar la medición. Finalmente, a la derecha del botón, hay un cuadro editable para añadir la longitud medida.

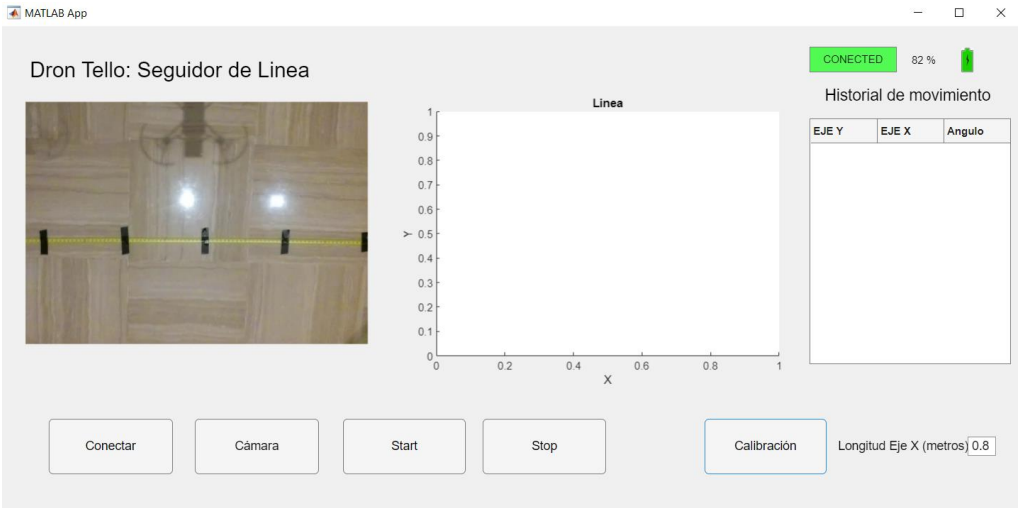


Figura 5.4. Proceso de calibración

Una vez conectado el dron y comprobada la cámara, el espejo del dron y la calibración, es momento de comenzar el proceso de seguimiento de la línea. Para ello, se han destinado los botones de “Start” y “Stop” para iniciar y detener el proceso si es necesario. Se posiciona el dron cerca de la línea y pulsando el botón

de inicio, el dron despega y comienza a realizar el proceso. Mientras tanto, se muestran en los ejes las imágenes obtenidas y procesadas por el dron (Figura 5.5).

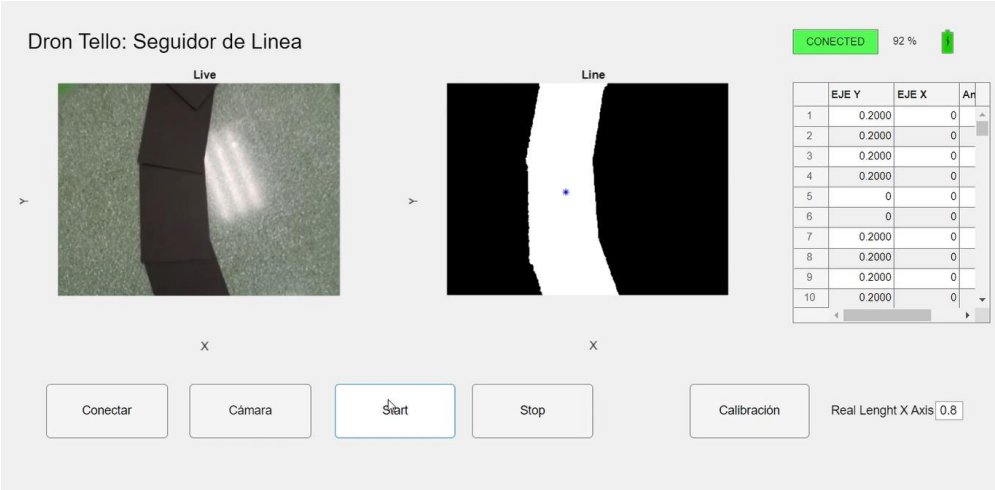


Figura 5.5. Proceso de seguimiento de la línea

En la parte derecha de la interfaz se encuentra el historial de movimientos, es una tabla que almacena los datos de cada movimiento realizado por el dron durante el vuelo.

Pruebas de funcionamiento

A continuación, se presentan las pruebas realizadas con el dron y la interfaz durante la realización de dos circuitos: uno abierto y otro cerrado (Figura 5.6 y Figura 5.7). Los videos correspondientes se encuentran disponibles en el siguiente enlace de Google Drive.

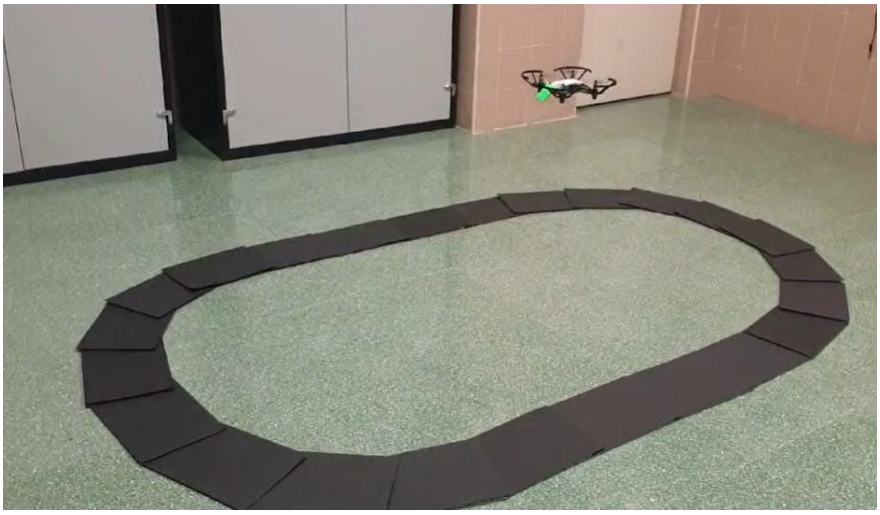


Figura 5.6. Seguimiento de circuito cerrado

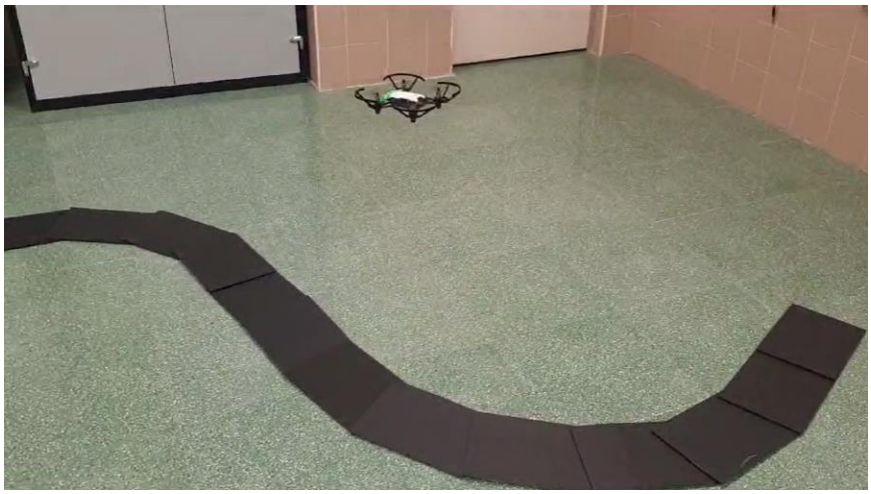


Figura 5.7. Seguimiento de circuito abierto

Es importante destacar que, durante el vuelo, se observa que el proceso es relativamente lento debido a las restricciones mencionadas durante el desarrollo del software. En los comandos de movimiento debe definirse la distancia a desplazar y hasta que el proceso no termine no se ejecutarán los siguientes comandos. Sin embargo, se ha conseguido que el dron cumpla la función realizando una navegación autónoma, integrando la visión por computador en el proceso.

5.2. Guiones de Practicas

En el marco de este estudio, se han desarrollado dos guiones de prácticas con el objetivo de integrar el dron Tello en la asignatura de visión por computador, aprovechando el entorno de desarrollo App Designer en Matlab. Cada guion de práctica se estructura para abordar aspectos clave del desarrollo, comenzando con la familiarización de los estudiantes con App Designer y el dron Tello.

Con estos guiones se busca ofrecer una idea de desarrollo de práctica básica y avanzada con el dron Tello. Estas prácticas pueden ser adaptadas a gusto del profesor y según la cantidad de alumnos y el nivel del grupo.

5.2.1. Práctica 1: Interfaz de control del dron Tello

Esta es una práctica de introducción en la que se enseñan los conceptos básicos para el control del dron Tello y el uso de App Designer. Durante el guion de la práctica se presentan varios ejemplos de uso del entorno desarrollando la adquisición de imagen con una cámara cualquiera y la creación de una interfaz

sencilla. Y por otro lado se presenta la conexión con el dron Tello. Una vez explicados estos conceptos, con sus respectivos ejemplos, se pide la creación de una interfaz que integre los movimientos básicos del dron, junto con la adquisición de imagen con la cámara del dron.

Al final del guion se presenta el resultado que se espera obtener en la práctica. El guion se encuentra detallado en el Anexo I.

5.2.2. Practica 2: El dron Tello. Seguidor de línea

En el segundo guion de prácticas presentado, se pretende que el alumno desarrolle un programa que permita el seguimiento de una línea, similar al caso de estudio. Sin embargo, en este caso, se adapta para que se pueda desarrollar en el transcurso de las horas de la práctica. Para ello, se proporciona las funciones básicas necesarias para llevar a cabo el proceso, y se guía con pequeños pasos hasta llegar a la interfaz final.

Al igual que con el primer guion, al final del documento se presenta la resolución propuesta. El guion se encuentra en el Anexo II.

6. CONCLUSIONES

En este trabajo, se ha llevado a cabo la integración del dron Tello como una herramienta docente en la asignatura de visión por computador, desarrollando un caso de estudio que establece bases sólidas para su posterior adaptación al contenido de la asignatura.

En una primera etapa, se ha realizado el desarrollo de un seguidor de línea que implementa un sistema de visión utilizando las herramientas y conceptos enseñados en la docencia. Matlab ha sido seleccionado como el entorno de desarrollo debido a su relevancia en la asignatura, junto con App Designer y un paquete de soporte específico para el dron Tello. A pesar de las limitaciones de movimiento presentadas por este paquete, el desarrollo del seguidor de línea ha sido satisfactorio, demostrando que el dron Tello es una herramienta versátil con un gran potencial.

El aprendizaje adquirido durante este proceso de programación del dron Tello ha permitido la creación de guiones de prácticas dedicados a enseñar los conceptos básicos de visión por computador, integrando el uso del dron como una herramienta activa en el aprendizaje. Este enfoque busca proporcionar a los estudiantes una experiencia práctica y directa en la aplicación de los conceptos aprendidos utilizando esta novedosa herramienta.

Además, cabe destacar que en este estudio se identifican áreas de mejora que podrían ser abordadas en próximos proyectos. Uno de los posibles aspectos a mejorar podría ser la exploración de nuevos métodos de programación para el control del dron Tello, con el objetivo de aprovechar al máximo sus capacidades sin restricciones significativas. Esta mejora podría contribuir a superar las limitaciones de movimiento presentadas por el paquete de soporte utilizado. Adicionalmente, se plantea la posibilidad de implementar técnicas más avanzadas de visión por computador en futuros desarrollos ampliando las posibilidades de aplicación del dron Tello como herramienta docente.

Este trabajo no solo busca integrar al dron Tello en la enseñanza actual, sino que pretende a ser una iniciativa que abra el camino a nuevos proyectos y

desarrollos. Se abre un nuevo campo de posibilidades y estudios, especialmente en el ámbito de una enseñanza innovadora, que tiene muchas aplicaciones prácticas fuera del ámbito docente.

Anexo I. Práctica 1: Interfaz de control del dron Tello

OBJETIVOS:

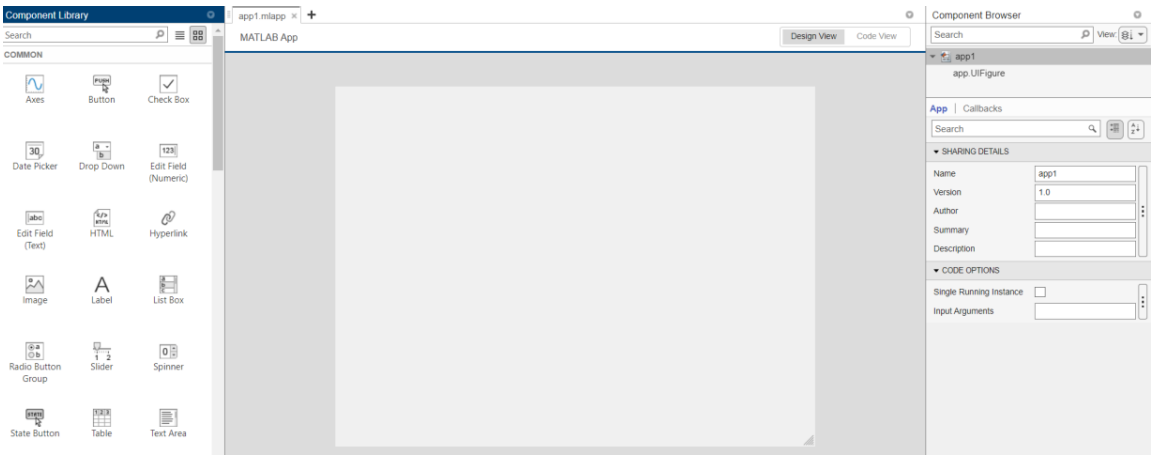
Aprender a utilizar los comandos de control del dron Tello y el entorno de desarrollo integrado, AppDesigner de Matlab.

RESUMEN:

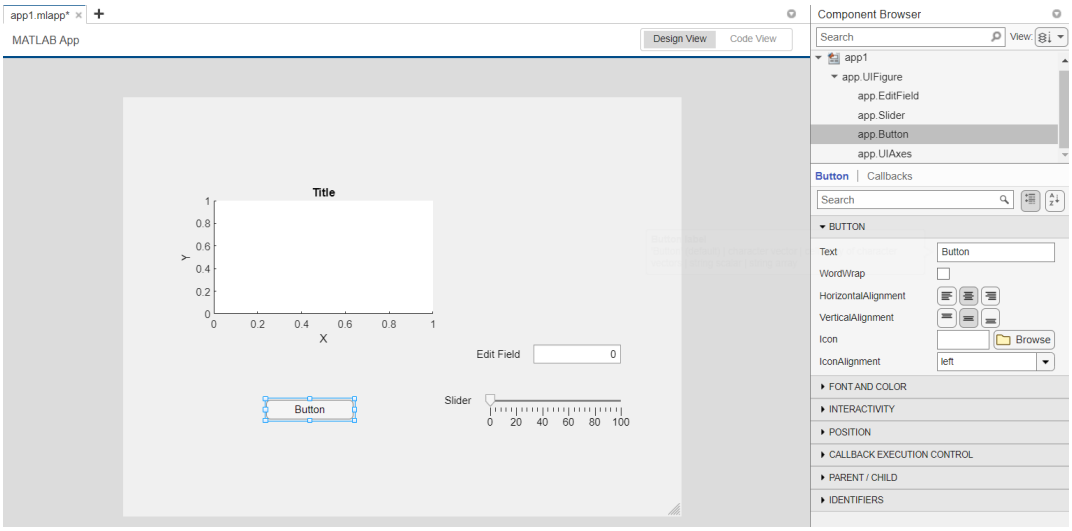
Se detallará las distintas funciones de Matlab necesarias para realizar la conexión y control de movimientos con el dron Tello. Además, se proporcionarán ejemplos básicos de interfaces creadas con AppDesigner.

INTRODUCCIÓN DE APP DESIGNER

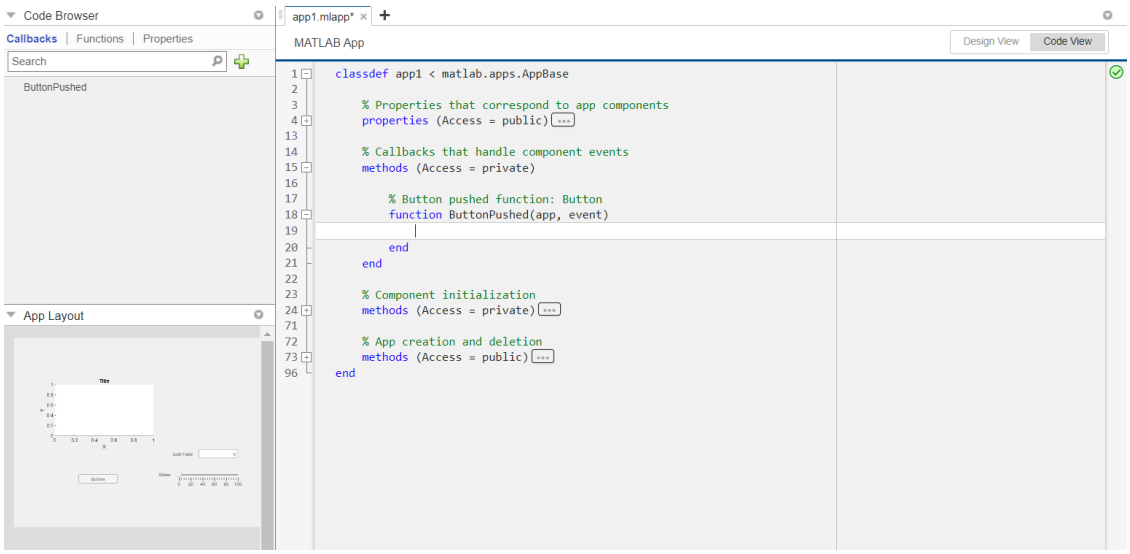
App Designer es un entorno de desarrollo integrado proporcionado por Matlab. Este entorno permite diseñar una gran variedad de aplicaciones arrastrando y soltando los componentes de la biblioteca sobre la interfaz y programando los eventos que ocurren en ella.



En la parte izquierda del entorno se encuentran los componentes visuales. Estos pueden arrastrarse hasta la zona central donde se diseña la aplicación. Y a la derecha, se muestra una lista de los componentes utilizados y opciones avanzadas de diseño.

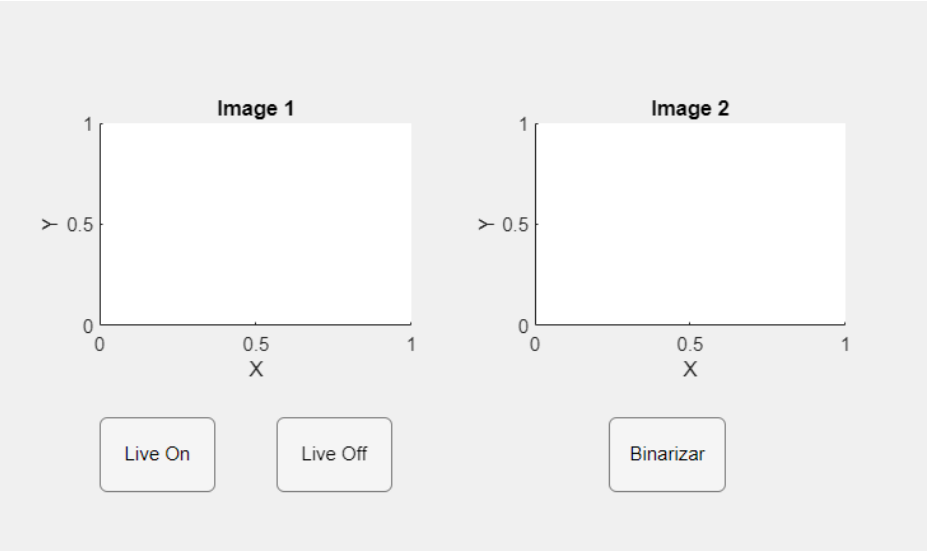


Por otro lado, hay una vista de código en la que el entorno genera automáticamente el código que se utiliza para el diseño de la aplicación. Este código se encuentra bloqueado, sin embargo, se puede crear código para programar el comportamiento de los componentes. A la izquierda de la vista de código hay un navegador donde se pueden crear propiedades, funciones y callbacks (funciones que se ejecutan al ocurrir un evento). Como se observa en la siguiente figura, tras crear el Callback, aparece una línea en color blanco, en la que se puede escribir código.



Las propiedades se usan como variables que pueden ser llamadas desde cualquier función de la aplicación. Para poder utilizarlas se llaman con el prefijo “app.” delante de la variable.

Interfaz con parada en vivo para procesar



Funciones:

- StartupFcn: Función de inicio del programa que obtiene las propiedades de la cámara y las almacena en una estructura.
- LiveOnButtonPushed (Callback): Código para la presentación de imagen en vivo en el axes “Image_1”.
- LiveOffButtonPushed (Callback): Código que para la adquisición en vivo y muestra la última captura.
- BinarizeButtonPushed (Callback): Código que binariza la imagen capturada en el momento y lo muestra en el axes “Image_2”.

Nota: Se utiliza la variable “video” como una propiedad, para poder utilizarse desde las demás funciones de la aplicación, con el nombre de “app.video”.

```
function startupFcn(app)
    vid=videoinput('winvideo');
    vidRes=get(app.vid,'VideoResolution');
    imWidth=vidRes(1);
    imHeight=vidRes(2);
    nBands=get(vid,'NumberOfBands');
    app.video=struct('manejador',vid,'res',vidRes,'anchura',imWidth,'altura',imHeight,'bandas',nBands);
end

% Button pushed function: LiveOn
function LiveOnButtonPushed(app, event)
    hImage = image(zeros(app.video.altura,app.video.anchura,app.video.bandas),'Parent',app.Image_1);
    preview(app.video.manejador,hImage);
end

% Button pushed function: LiveOff
function LiveOffButtonPushed(app, event)
    frame=getsnapshot(app.video.manejador);
    imshow(frame,'Parent',app.Image_1); %Al mostrar el frame se para el preview
end

% Button pushed function: Binarize
function BinarizeButtonPushed(app, event)
    frame=getsnapshot(app.video.manejador);
    imgray=rgb2gray(frame);
    imbw=im2bw(imgray,80/256);%ejemplo de procesado binarizando imagen
    imshow(imbw,'Parent',app.Image_2);
end
```

Interfaz con procesado en vivo

Funciones:

- StartupFcn: Función de inicio del programa que conecta con la cámara de video
- StartButtonPushed (Callback): Código para el procesado de imagen en vivo. Se usa un bucle en el que se hace una captura de imagen, se procesa, y se muestra sin procesar y procesado en los dos ejes.

- StopButtonPushed (Callback): Código que para el procesado en vivo interrumpiendo el bucle.

```
properties (Access = private)
    vid %Estructura con los datos de video
end

% Callbacks that handle component events
methods (Access = private)

    % Code that executes after component creation
    function startupFcn(app)
        app.vid=videoinput('winvideo');%Conexión con la cámara de videoS
    end

    % Button pushed function: Start
    function StartButtonPushed(app, event)
        app.Stop.UserData = false; %flag para interrumpir el bucle y dejar de procesar imagen
        while app.Stop.UserData == false
            frame=getsnapshot(app.vid);
            imgray=rgb2gray(frame);
            imbw=im2bw(imgray,80/256);%ejemplo de procesado binarizando imagen
            imshow(frame,'Parent',app.Image_1);
            imshow(imbw,'Parent',app.Image_2);
        end
    end

    % Button pushed function: Stop
    function StopButtonPushed(app, event)
        app.Stop.UserData = true; %Botón para poner flag true e interrumpir el bucle
    end
end
```

CONEXIÓN TELLO-MATLAB

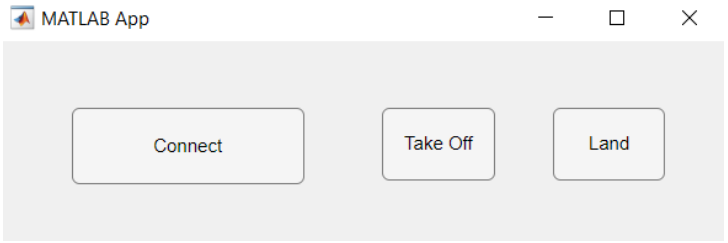
Para el control del dron Tello mediante Matlab, se empleará "Matlab Support Package for Ryze Tello drone", que es una extensión de libre uso ofrecida por Matlab. Esta herramienta permite controlar los movimientos del dron, al igual que la adquisición de imagen y obtención de datos de vuelo, gracias a los comandos que ofrece.

Para la conexión con Tello se deben realizar los siguientes pasos:

1- Encender el dron Tello y esperar al indicador de luz amarillo intermitente.
2- Conectar el ordenador a la red Wi-Fi creada por el dron.
3- Usar la función ryze() para conectar Tello a Matlab como se muestra en la figura. El indicador de luz del dron se fijará en color verde.

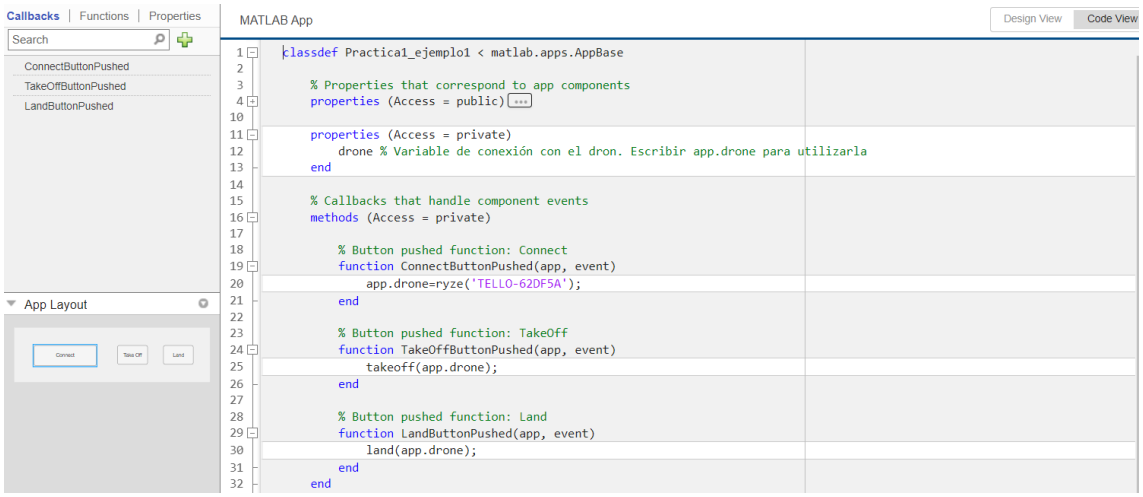
```
Command Window
>> dron=ryze('TELLO-62DF5A')

dron =

  ryze with properties:

                Name: "Tello"
                  ID: "TELLO-62DF5A"
           IPAddress: "192.168.10.1"
               State: "landed"
        BatteryLevel: 48 %
         StationMode: 0
    AvailableCameras: ["FPV"]
```

Interfaz básica de movimiento



Funciones:

- ConnectButtonPushed (Callback): Conectar Matlab con el dron Tello.
- TakeOffButtonPushed (Callback): Despegar el dron.
- LandButtonPushed (Callback): Aterrizar el dron.

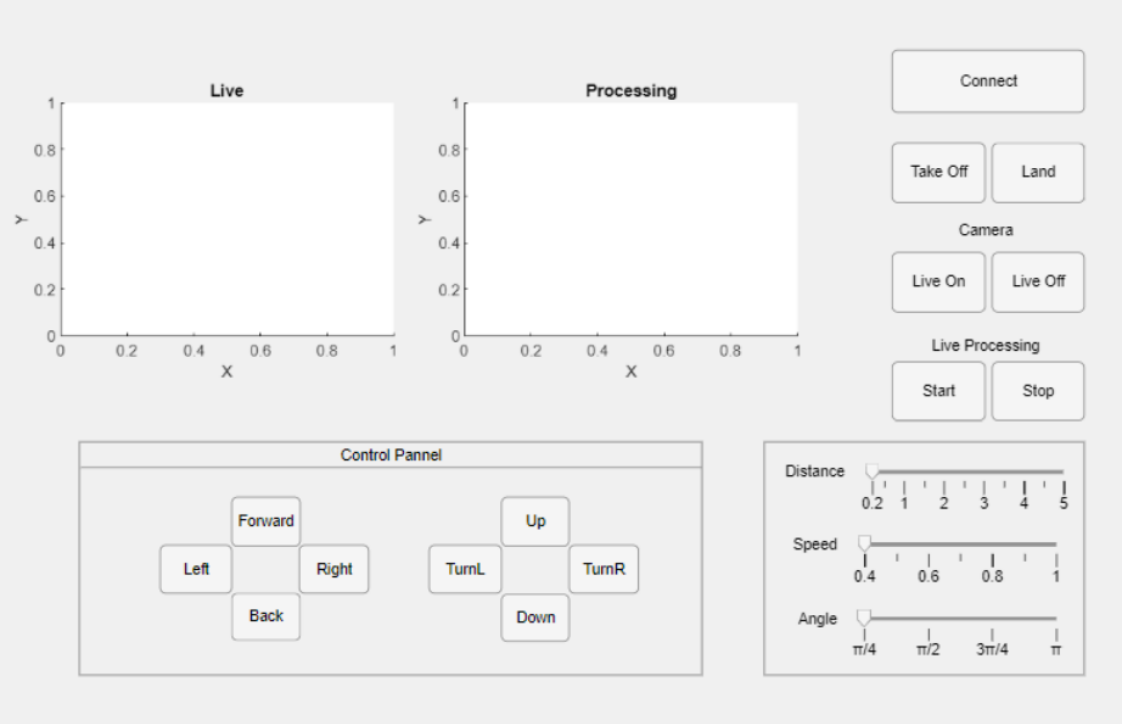


EJERCICIOS

1- Crear una interfaz para conectar con el dron Tello, iniciar la adquisición de imagen en vivo con la cámara del dron y realizar la parada de imagen.
2- Añadir paralelamente, el procesamiento en vivo de la imagen usando los mismos “axes”.
3- Añadir a la interfaz los movimientos básicos y avanzados para el vuelo: despegar, aterrizar, realizar movimientos en todas direcciones y el giro.

Utilizar el componente “slider” para definir: distancia, velocidad y ángulo de los movimientos.

RESOLUCIÓN




```
properties (Access = public)
    drone % Variable de conexión con el dron
    camera % Variable de conexión con la cámara
end

% Callbacks that handle component events
methods (Access = private)

    % Button pushed function: Connect
    function ConnectButtonPushed(app, event)
        app.drone=ryze('TELLO-62DF5A');
        app.camera=camera(app.drone);
    end

    % Button pushed function: TakeOff
    function TakeOffButtonPushed(app, event)
        takeoff(app.drone);
    end

    % Button pushed function: Land
    function LandButtonPushed(app, event)
        land(app.drone);
    end

    % Button pushed function: LiveOn
    function LiveOnButtonPushed(app, event)
        frame=snapshot(app.camera);
        Xlenght=size(frame,2);
        Ylenght=size(frame,1);
        hImage = image( zeros(Ylenght,Xlenght,3),'Parent',app.Image_1);
        preview(app.camera,hImage);
    end
```

```
function LiveOffButtonPushed(app, event)
    frame=snapshot(app.camera);
    imshow(frame,'Parent',app.Image_1);
end

% Button pushed function: Forward
function ForwardButtonPushed(app, event)
    moveforward(app.drone,'Distance',app.DistanceSlider.Value,'Speed',app.SpeedSlider.Value,'WaitUntilDone',false)
end

% Button pushed function: Back
function BackButtonPushed(app, event)
    moveback(app.drone,'Distance',app.DistanceSlider.Value,'Speed',app.SpeedSlider.Value,'WaitUntilDone',false);
end

% Button pushed function: Up
function UpButtonPushed(app, event)
    moveup(app.drone,'Distance',app.DistanceSlider.Value,'Speed',app.SpeedSlider.Value,'WaitUntilDone',false);
end

% Button pushed function: Down
function DownButtonPushed(app, event)
    movedown(app.drone,'Distance',app.DistanceSlider.Value,'Speed',app.SpeedSlider.Value,'WaitUntilDone',false);
end

% Button pushed function: Left
function LeftButtonPushed(app, event)
    moveleft(app.drone,'Distance',app.DistanceSlider.Value,'Speed',app.SpeedSlider.Value,'WaitUntilDone',false);
end
```

```
function RightButtonPushed(app, event)
    moveright(app.drone,'Distance',app.DistanceSlider.Value,'Speed',app.SpeedSlider.Value,'WaitUntilDone',false);
end

% Button pushed function: TurnL
function TurnLButtonPushed(app, event)
    turn(app.drone,-deg2rad(app.AngleSlider.Value));
end

% Button pushed function: TurnR
function TurnRButtonPushed(app, event)
    turn(app.drone,deg2rad(app.AngleSlider.Value));
end

% Button pushed function: StartLiveProcessing
function StartLiveProcessingButtonPushed(app, event)
    app.StopLiveProcessing.UserData = false;
    while app.StopLiveProcessing.UserData == false
        frame=snapshot(app.camera);
        imgray=rgb2gray(frame);
        imbw=im2bw(imgray,80/256);%ejemplo de procesado, se puede usar cualquiera de ejemplo
        imshow(frame,'Parent',app.Image_1);
        imshow(imbw,'Parent',app.Image_2);
    end
end

% Button pushed function: StopLiveProcessing
function StopLiveProcessingButtonPushed(app, event)
    app.StopLiveProcessing.UserData = true;
end
```

Anexo II. Práctica 2: El dron Tello. Seguidor de línea

OBJETIVOS:

El objetivo de la práctica es integrar el control del dron Tello con el uso de un sistema de percepción visual, para obtener un sistema de navegación autónomo.

SEGUIDOR DE LINEA

En esta práctica se pretende crear una interfaz con App Designer para que el dron Tello realice el seguimiento de una línea de forma autónoma. El proceso consta de dos partes: el procesamiento de imagen y el sistema de navegación. Se llevará a cabo con el uso de un espejo que permite orientar el campo de visión hacia el suelo.

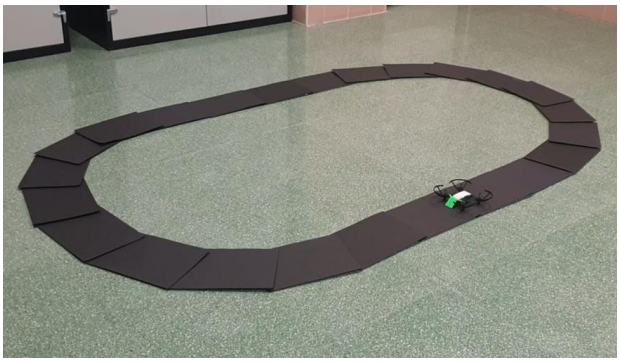


Figura. Línea

Procesado de Imagen

En este proceso se obtendrán las características de la línea que serán interpretadas en el sistema de navegación. La navegación consistirá en el avance del dron con rectificación de orientación y posición cuando sea necesario. Por ello, las características a extraer de la línea serán: el centroide y el ángulo de orientación.

Se proponen los siguientes pasos en el procesamiento de imagen:

- Adquisición de Imagen: Usar la cámara del dron para obtener el fotograma a procesar (**snapshot**).
- Preprocesamiento de Imagen: Obtener la imagen binarizada únicamente con la línea (**rgb2gray**, **im2bw, bwareaopen, imfill**).

Además, hay que tener en cuenta que el espejo invierte la imagen verticalmente (**flip**) y que se trabaja con los píxeles blancos de la imagen (**imcomplement**).

- Extracción de características: Extraer centroide y orientación de la línea (**regionprops**).

Ejercicio 1. Crear una interfaz con procesado de imagen en vivo (usando un bucle) para el dron Tello. Debe incluir botones de inicio y parada, y dos ejes donde se muestre la imagen sin procesar y la imagen procesada.

Sistema de Navegación

Tras la extracción de características de la línea, será necesario interpretarlas, teniendo en cuenta la posición del dron en el centro de la imagen.

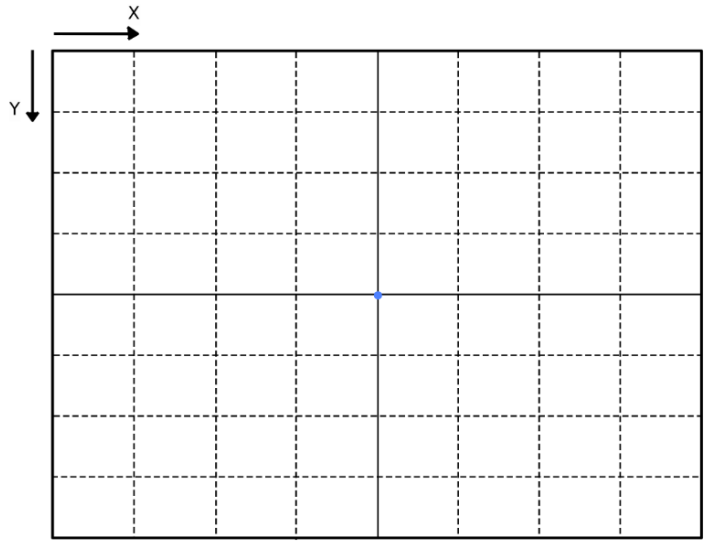


Figura. Ejes de coordenadas de la imagen

Se proponen los siguientes pasos en el sistema de navegación:

- Cálculo de errores: Obtener el error de posición con respecto al centro de la imagen y el error de orientación con respecto al eje Y. El origen de coordenadas se encuentra en la esquina superior izquierda.
- Asignación de variables: A partir de los errores, asignar los argumentos de entrada para las funciones de movimiento.

- Órdenes de movimiento: Rectificar la orientación cuando sea necesario (**turn**). En caso contrario, desplazar el dron de frente y rectificar la posición lateral si es necesario (**move**).

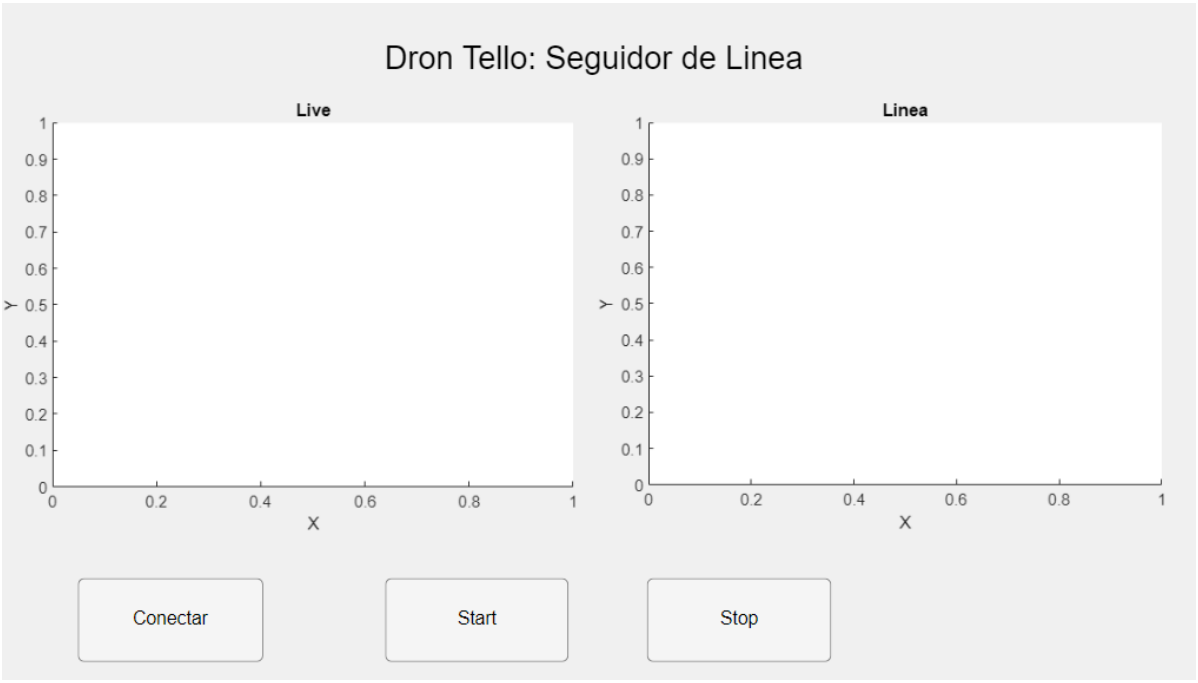
Hay que tener en cuenta que la distancia mínima de movimiento es de 0.2 metros. Para más información detallada sobre las funciones y sus argumentos de entrada, revisar el enlace.

Ejercicio 2. Añadir el sistema de navegación al procesado de imagen de la interfaz creada. El dron debe despegar antes de iniciar el proceso y aterrizar cuando finalice.

Nota: Añadir una orden de aterrizaje en el botón de parada para una posible emergencia.

Ejercicio 3. Modificar el sistema de navegación, suponiendo que la línea tiene inicio y final. Tener en cuenta la posición del centroide para el inicio y fin de línea.

RESOLUCIÓN




```
properties (Access = public)
    drone % Variable de conexión con el dron
    camera % Variable de conexión con la cámara
    Xlenght
    Ylenght
end

methods (Access = private)

    function linedata = ImageProcessing(app)
        im=snapshot(app.camera); %Captura de imagen
        imgray=rgb2gray(im); %Obtener la imagen en escala de grises
        imbw=im2bw(imgray,80/256); %Binarizar la imagen
        iminv=imcomplement(imbw); %Cambiar 0 por 1 y viceversa
        imbw2=bwareaopen(iminv,20000,8); %Eliminar areas con menos pixeles
        imbw3=imfill(imbw2,8,"holes"); %Rellena pixeles dentro de la imagen
        line=flip(imbw3,1); %Invertir imagen en vertical
        linedata = regionprops (line,'Centroid','Orientation'); %Obtener centroide y orientación

        %Muestra de las imagenes y centroide
        imshow(flip(im,1),'Parent',app.LiveAxes); %Muestra la imagen sin procesar
        imshow(line,'Parent',app.LineAxes); %Muestra la imagen procesada
    end
```

```
function Navegation(app,linedata)
    %Calculo de Errores y asignación de angulos y distancias
    Xcentroid=linedata.Centroid(1); %Posición en X del centroide de la linea
    Ycentroid=linedata.Centroid(2); %Posición en Y del centroide de la linea
    errorX=Xcentroid-(app.Xlenght/2); %Error respecto al centro de la imagen
    errorY=Ycentroid-(app.Ylenght/2);
    if abs(errorX)>=(app.Xlenght/8) %Si es menor que la distancia especificada no se mueve
        Xdistance=0.2;
    else
        Xdistance=0;
    end
    Xangle=linedata.Orientation; %Ángulo de la linea con respecto al eje X
    if Xangle>=0
        angle=90-Xangle;
    else
        angle=-90-Xangle;
    end

    %Ordenes de movimiento en función de los errores obtenidos
    if abs(errorY)<2*(app.Ylenght/8)
        if abs(angle)>10 && abs(angle)<=90 && abs(errorX)<(3*(app.Xlenght/8))
            turn(app.drone,deg2rad(angle));
        else
            move(app.drone,[0.2 Xdistance 0],'Speed',0.5,'WaitUntilDone',true);
        end
    elseif abs(errorY)>=2*(app.Ylenght/8)
        if errorY<0
            move(app.drone,[0.2 Xdistance 0],'Speed',0.5,'WaitUntilDone',true);
        elseif errorY>0
            app.StopButton.UserData = true;
        end
    end
end
```

```
% Button pushed function: StartButton
function StartButtonPushed(app, event)
    app.StopButton.UserData = false; %Flag para romper el bucle al pulsar botón Stop
    takeoff(app.drone); %Despegar
    while app.StopButton.UserData == false %Bucle navegación autónoma
        linedata = ImageProcessing(app); %Adquisición y Procesamiento de la Imagen

        if size(cat(2,linedata.Centroid),2)~=2
            fprintf('No se ha detectado centroide\n');
            break %Rompe el bucle si no detecta solo un centroide
        end

        Navegation(app,linedata); %Sistema de navegación
    end
    land(app.drone); %Aterrizar
end

% Button pushed function: StopButton
function StopButtonPushed(app, event)
    app.StopButton.UserData = true;
    land(app.drone);
end

% Button pushed function: Connect
function ConnectButtonPushed(app, event)
    app.drone=ryze('TELLO-62DF5A');
    app.camera=camera(app.drone);
    frame=snapshot(app.camera); %Captura para obtener la resolución de la imagen
    app.Xlenght=size(frame,2);
    app.Ylenght=size(frame,1);
end
```

Bibliografía

[1] M. Reese, «The Steam-Powered Pidgeon of Archytas - The Flying Machine of Antiquity,» 2021. [En línea]. Available: https://www.ancient-origins.net/history-famous-people/steam-powered-pigeon-002179.

[2] V. Delgado, «Historia de los drones,» 2016. [En línea]. Available: http://eldrone.es/historia-de-los-drones/.

[3] R. Austin, Unmanned Aircraft Systems: UAVs design, development and deployment, 2010.

[4] Aeromontes, «Aeromontes,» 2023. [En línea]. Available: https://aeromontes.es/donde-viene-palabra-drone-etimologia/.

[5] F. Giones y A. Brem, «Fron toys to tools: The co-evolution of technological and entrepreneurial developments in the drone industry,» *Business Horizons,* pp. 875-884, 2017.

[6] G. d. E. Ministerio de Fomento, «Plan estratégico para el desarrollo del sector civil de drones en España 2018-2021,» Madrid. España, 2018.

[7] E. Martín, La meteorología en el vuelo de aeróstatos y aerodinos, 2008.

[8] K. Nonami, F. Kendoul, S. Suzuki y D. Nakazawa, Autonomous Flying Robots, 2010.

[9] «Wingtra,» 2023. [En línea]. Available: https://wingtra.com/.

[10] J. Patterson, «An Aerial View of the Future – Drones in Construction,» 2018. [En línea]. Available: https://www.geospatialworld.net/blogs/an-aerial-view-of-the-future-drones-in-construction/.

[11] NASA, «NASA's Dragonfly will fly around Titan looking for origins, signs of life,» 2019. [En línea]. Available: https://www.nasa.gov/news-release/nasas-dragonfly-will-fly-around-titan-looking-for-origins-signs-of-life/.

[12] F. Sattar y M. Nawaz, «Developing Computational Thinking in STEM Education with Drones,» 2023.

[13] «Ryze Robotics,» [En línea]. Available: https://www.ryzerobotics.com/es.

[14] «Tello SDK User Guide,» [En línea]. Available: https://dl-cdn.ryzerobotics.com/downloads/Tello/Tello%20SDK%202.0%20User%20Guide.pdf.

[15] «Tello Manual de Usuario,» [En línea]. Available: https://dl-cdn.ryzerobotics.com/downloads/Tello/201806mul/Tello%20User%20Manual%20V1.0_ES.pdf.

[16] «Umiles Partes de un dron,» 6 Marzo 2023. [En línea]. Available:

https://umilesgroup.com/partes-de-un-dron/.

[17] TomE8, «Thingiverse, DJI Tello mirror clip,» 2019. [En línea]. Available: https://www.thingiverse.com/thing:4070064.

[18] M. Pendergrast, Historia de los espejos, 2001.

[19] R. C. González, Digital Image Processing Using MATLAB, 2020.

[20] «MATLAB App Designer,» Mathworks, [En línea]. Available: https://es.mathworks.com/products/matlab/app-designer.html.

[21] «MATLAB Support Package for Ryze Tello Drones,» [En línea]. Available: https://es.mathworks.com/matlabcentral/fileexchange/74434-matlab-support-package-for-ryze-tello-drones?s_tid=FX_rc3_behav.

[22] R. Szeliski, Computer Vision Algorithms and Applications, 2010.

[23] A. Hornberg, Handbook of Machine and Computer Vision, 2006.