Investigación e implementación de un sistema de aprendizaje sobrevuelo y seguimiento de objetos para mejorar la seguridad en ambientes externos mediante la utilización de IA en combinación con un dron, para la Universidad de las Fuerzas Armadas ESPE sede Latacunga.

Lizano Flores, Kevin Iván y Tite Ruíz, Jonathan Danilo

Departamento de Ciencias de la Energía y Mecánica

Carrera de Ingeniería en Mecatrónica

Trabajo de titulación, previo a la obtención del título de Ingeniero Mecatrónico

Msc. Mendoza Chipantasi, Dario José

11 de enero de 2023

Latacunga

Reporte de verificación de contenido

Tesis - Lizano, Tite.docx

Scanned on: 16:42 December 12, 2022 UTC

6.1%	87	14995
Overall Similarity Score	Results Found	Total Words in Text

Identical Words	470
Words with Minor Changes	99
Paraphrased Words	292
Omitted Words	951

X

Msc. Mendoza Chipantasi, Dario José
C. C.: 0603110834

COPYLEAKS Website | Education | Businesses

Departamento de Ciencias de la Energía y Mecánica

Carrera de Ingeniería en Mecatrónica

Certificación

Certifico que el trabajo de titulación, **"Investigación e implementación de un sistema de aprendizaje sobrevuelo y seguimiento de objetos para mejorar la seguridad en ambientes externos mediante la utilización de IA en combinación con un dron, para la Universidad de las Fuerzas Armadas ESPE sede Latacunga."** fue realizado por los señores **Lizano Flores, Kevin Iván y Tite Ruíz, Jonathan Danilo**; el mismo que cumple con los requisitos legales, teóricos, científicos, técnicos y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, además fue revisado y analizado en su totalidad por la herramienta de prevención y/o verificación de similitud de contenidos; razón por la cual me permito acreditar y autorizar para que se lo sustente públicamente.

Latacunga, 12 de diciembre del 2022

Msc Mendoza Chipantasi, Dario José
C.C.: 0603110834

Departamento de Ciencias de la Energía y Mecánica

Carrera de Ingeniería en Mecatrónica

Responsabilidad de Autoría

Nosotros, **Lizano Flores, Kevin Iván** con cédula de ciudadanía No. 0931850358 y **Tite Ruíz, Jonathan Danilo**, con cédula de ciudadanía No. 1313168179, declaramos que el contenido, ideas y criterios del trabajo de titulación: **"Investigación e implementación de un sistema de aprendizaje sobrevuelo y seguimiento de objetos para mejorar la seguridad en ambientes externos mediante la utilización de IA en combinación con un dron, para la Universidad de las Fuerzas Armadas ESPE sede Latacunga."** es de nuestra autoría y responsabilidad, cumpliendo con los requisitos legales, teóricos, científicos, técnicos, y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Latacunga, 12 de diciembre del 2022

Lizano Flores, Kevin Iván
C.C.: 0931850358

Tite Ruz, Jonathan Danilo
C.C.: 1313168179

Departamento de Ciencias de la Energía y Mecánica

Carrera de Ingeniería en Mecatrónica

Autorización de Publicación

Nosotros, **Lizano Flores, Kevin Iván** con cédula de ciudadanía No. 0931850358 y **Tite Ruíz, Jonathan Danilo**, con cédula de ciudadanía No. 1313168179, autorizamos a la Universidad de las Fuerzas Armada s ESPE publicar el trabajo de titulación: **"Investigación e implementación de un sistema de aprendizaje sobrevuelo y seguimiento de objetos para mejorar la seguridad en ambientes externos mediante la utilización de IA en combinación con un dron, para la Universidad de las Fuerzas Armadas ESPE sede Latacunga."** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de nuestra responsabilidad.

Latacunga, 12 de diciembre del 2022

Lizano Flores, Kevin Iván
C.C.: 0931850358

Tite Ruz, Jonathan Danilo
C.C.: 1313168179

Dedicatoria

«Para mi yo de la escuela (Creo que estarías orgulloso de ti mismo).

Para Monserrat Párraga (Ojalá pudieras verme, estoy dando lo mejor que tengo).»

Tite Ruíz, Jonathan Danilo

Dedicatoria

La realización de esta tesis se la dedico a mis padres quienes me apoyaron constantemente durante el transcurso de mi carrera, supieron darme ánimos para continuar y me enseñaron a salir adelante cuando trataba de desistir, a mis compañeros de trabajo quienes me guiaron cuando más lo necesitaba y me acunaron un espacio para que pudiera progresar, a mis amigos que durante toda la carrera fueron una mano amiga que me enseñaron el gran valor de ser unidos, y especialmente a mi enamorada, Daysi Quimbita quién a pesar de todo lo que pasamos me enseñó a buscar objetivos, metas y guiarme en lo que me apasiona.

Lizano Flores, Kevin Iván

Agradecimientos

«A Monserrat y Ramón, por formar mis cimientos.

A Sixto, por mostrarme lo que verdaderamente significa ser un hombre.

A Zoila, por su amor incondicional y sus cálidos abrazos.

A Grace, por enseñarme que ser altruista no siempre es fácil,

pero que siempre valdrá la pena.

A Julissa, por su valiosa compañía y cariño sincero.

A Mirian, por el amor y las acciones que no merecía,

pero que siempre guardo con cariño en mi corazón.

A Gabriel, Selena, Kevin, Martín, Ronny, Andrea, Edison, Lando y Klever,

por ser la familia que elegí.

A la ESPE, por todo lo aprendido.

A Ecuador, porque aquí conocí todo lo que amo.»

Tite Ruíz, Jonathan Danilo

Agradecimientos

Le doy gracias a mis padres quienes han sido el pilar fundamental de nuestro progreso, sueños, metas, logros a lo largo de nuestra vida personal y académica. A cada uno de mis familiares, amigos que de alguna manera han estado a nuestro lado apoyándonos.

A los docentes que formaron mi carrera, quienes debemos gran parte de nuestro desarrollo, especialmente a Ing. Misael Pazmiño, quien me enseñó a ser honestos y nunca desfallecer ante las grandes adversidades, siempre existe oportunidad y esperanza mientras estemos vivos. De igual manera a Msc. Darío Mendoza, quien nos apoyó y supo guiarnos de la mejor manera en nuestro trabajo de titulación.

Lizano Flores, Kevin Iván

Índice de contenidos

Índice de tablas

Índice de figuras

Resumen

El presente trabajo se enfoca en el problema de reconocimiento y seguimiento autónomo de objetos con un VANT (Vehículo Aéreo no Tripulado) y en tiempo real. Se centra en el uso de RNAs (Redes Neuronales Artificiales) convolucionales y la implementación de YOLO como algoritmo de detección de objetos. Para este sistema se desarrollaron tres etapas principales. La primera etapa consiste en la instalación y el acondicionamiento de un driver que permita enviar comandos de vuelo y recibir fotogramas en tiempo real a través del VANT, haciendo uso de un procesador que permita procesar estos datos y con base en estos, ejecutar un algoritmo de vuelo autónomo. La segunda etapa es la toma de datos de objetivos para el entrenamiento, se elaboró un algoritmo mediante técnicas clásicas de visión artificial para facilitar la captura y el etiquetado de las imágenes, para esto se utilizan técnicas como emparejamiento de plantillas, seguimiento mediante flujo óptico, histogramas, entre otras. La última etapa involucra el seguimiento autónomo del VANT haciendo uso de las coordenadas obtenidas mediante YOLO y el algoritmo de vuelo. Se consiguió un seguimiento óptimo en todos los casos, considerando siempre un entrenamiento de la red neuronal con por lo mínimo 500 imágenes y con una cantidad de épocas mayor o igual a 700.

Palabras clave: vehículo aéreo no tripulado, seguimientos de objetos, YOLO, visión artificial, vuelo autónomo.

Abstract

The present work focuses on the problem of autonomous object recognition and tracking with a UAV (Unmanned Aerial Vehicle) and in real time. It focuses on the use of convolutional ANNs (Artificial Neural Networks) and the implementation of YOLO as object detection algorithm. Three main stages were developed for this system. The first stage consists of the installation and conditioning of a driver that allows sending flight commands and receiving frames in real time through the UAV, making use of a processor that allows processing this data and, based on this data, executing an autonomous flight algorithm. The second stage is the acquisition of target data for training, an algorithm was developed using classical computer vision techniques to facilitate the capture and labeling of images, using techniques such as template matching, optical flow tracking, histograms, among others. The last stage involves the autonomous tracking of the UAV using the coordinates obtained by YOLO and the flight algorithm. Optimal tracking was achieved in all cases, always considering a training of the neural network with at least 500 images and with a number of epochs greater than or equal to 700.

Keywords: unmanned aerial vehicle, object tracking, YOLO, artificial vision, autonomous flight.

Capítulo I.

Aspectos Generales

Planteamiento del Problema

La seguridad es un término bastante discutido en todo ámbito, tanto público como privado, la necesidad de seguridad ha llevado a varios desarrollos tecnológicos, como lo son los sistemas de vigilancia y simuladores con realidad virtual, los mismos que se aplican a las prácticas del ámbito industrial, así como también simuladores de momentos de crisis, entre otros.

La incorporación de cuadricópteros a este tema ha aumentado las posibilidades, debido a que estos elementos, pueden movilizarse a distintas zonas, sin tener un tripulante en el mismo. Con la incorporación de cámaras y sistemas de conexión a servidores, se puede tener un sistema de seguridad móvil, aumentando así, las oportunidades de objetivos con acciones indebidas, intrusiones, entre otros.

Aunque las soluciones existen, el procedimiento de los mismos incluye un operario, el cual, por medio de operaciones dirige el dron a distancia. En esto se ve una dependencia del dispositivo por el operario, dejando marcada una necesidad para que este sea completamente autónomo. Existen varios trabajos relacionados los cuales permiten, solamente la detección de objetos con algoritmos de visión artificial, sin incorporar algoritmos de seguimiento. Agregando a la problemática, estos algoritmos son dependientes de terceros, es decir, es necesario contratar con los propietarios para poder tener acceso a los mismos. La tesis busca precisamente crear un algoritmo propio el cual puede ser implementado en cualquier dron que cumpla características dadas por los autores. Existen varios ejemplos de adquisiciones de drones por parte de empresas actualmente, y por lo general, estos se buscan por temas de seguridad precisamente.

Tal es el caso del ECU911, el cual cuenta con 2 aeronaves no tripuladas, “Drones”, utilizadas para la seguridad pública. Los cuales funcionan como sistemas de seguridad móviles,

llegando a zonas donde el ser humano no puede acceder. El cual en 2016 brindó apoyo visual en la implosión realizada en un centro comercial en Portoviejo.

Al igual que EMASESA, la cual incorpora drones para gestionar la seguridad de su infraestructura, la empresa menciona que los drones se encargan de la recepción y monitorización de las alarmas, ampliando la visión y el control de los perímetros y los tiempos de respuesta ante posibles intrusiones.

Los problemas planteados vieron una solución tecnológica en la aplicación de drones, debido a que las condiciones de ubicación, control o vigilancia se vuelven inaccesible para el ser humano. Estos drones se suelen adquirir con la capacidad de detección, pero no de seguimiento, por lo que con anterioridad se mencionó y se pueden comentar en los ejemplos, necesitan de un operador con experiencia y habilidad en estos dispositivos. Con la utilización de redes neuronales y visión artificial se pueden mejorar los sistemas ya sean de seguimiento o detección, mejorando y ampliando de esta manera las posibilidades de su utilización y desarrollo, ya que actualmente no existen desarrollos nacionales que se acoplen a las necesidades locales.

Antecedentes

Las primeras técnicas robustas de reconocimiento y ubicación de objetos sobre una imagen mediante visión por computadora, surgen a partir de las investigaciones de David G. Lowe, quien en 2004 presenta su paper “Distinctive Image Features from Scale-Invariant Keypoints”, en el cual expone:

> Este artículo presenta un método para extraer características invariantes distintivas de las imágenes que pueden utilizarse para realizar una correspondencia fiable entre diferentes vistas de un objeto o escena. Las características son invariantes a la escala y la rotación de la imagen, y se demuestra que proporcionan una coincidencia robusta a través de una gama sustancial de distorsión afín, el cambio en el punto de vista 3D, la adición de ruido, y el cambio en la iluminación. (G. Lowe, 2004, pág. 1)

Luego, con el desarrollo de la visión artificial impulsada por las RNAs (Redes Neuronales Artificiales), se han conseguido muchos avances en el reconocimiento de objetos mediante una cámara. En la actualidad existen varios dispositivos comerciales capaces de realizar este tipo de reconocimientos, el más común y accesible es un teléfono celular moderno. Con la inmersión de aplicaciones móviles potenciadas por inteligencia artificial IA, los celulares actuales pueden conectarse a una RNA en la nube y reconocer objetos particulares en su entorno. Sin embargo, existen también dispositivos actuales más relacionados al tema del presente trabajo.

Los Vehículos Aéreos no Tripulados (VANTs), suelen incorporar algoritmos de inteligencia artificial para realizar tareas de búsqueda, reconocimiento del entorno y ejecutar maniobras autónomas de acuerdo a este entorno. Esta necesidad surge tanto en aplicaciones comerciales tales como hacer tomas de videos con manos libres, análisis e inspección de terrenos y estructuras, etc., como en aplicaciones militares, analizando áreas con objetivos altamente peligrosos. Aunque este tipo de equipos son algo escasos de encontrar en el mercado por su precio. Dentro de la academia también se han desarrollado proyectos similares, tal como se plantea en el trabajo de fin de grado del señor (Fernández, 2017), en el cual se describe “el proceso llevado a cabo para diseñar y construir un Dron cuatrirrotor ligero robotizado y la electrónica asociada para que siga objetos mediante visión artificial”. El autor en este proyecto se apoya de las bibliotecas de OpenCV en un entorno Linux, todo Open Source.

De manera análoga, en la ESPE Latacunga se desarrolló un proyecto de titulación similar, en el cual los autores automatizaron el seguimiento de personas dentro de un área específica mediante el uso de un drone y visión artificial implementada, (Morales y Paucar, 2017) explican que “Esto se realiza por medio del uso software libre y un sistema operativo dedicado explícitamente al control y comunicación de robots. Por medio de visión artificial se consigue la detección de personas que es el objetivo principal de este proyecto.”

Las aplicaciones en seguridad para los VANT no pasan desapercibidas en la realidad local, siendo así que, en noviembre del 2020, cuatro drones fueron adquiridos por el Municipio de Latacunga, según lo reporta el portal La Gaceta (Gaceta, 2020). Esta inversión en seguridad se enfatiza en el control de personas en espacios públicos y se ha seguido utilizando hasta la fecha actual. Sin embargo; existen algunas limitaciones para obtener un mayor rendimiento utilizando este tipo de sistemas. La utilización de un producto comercial siempre está sujeta a las especificaciones del fabricante, ya que el diseño del producto posee un enfoque determinado y muchas veces no tiene la flexibilidad suficiente para adaptarse a nuevas o diferentes tecnologías. Los drones adquiridos por el municipio, son pilotados de manera manual por las calles de Latacunga y su éxito en el seguimiento de personas sospechosas o de cualquier objetivo, dependen directamente de la habilidad del piloto.

El punto fuerte del presente trabajo, es diseñar un sistema que permita flexibilidad en la comunicación y sea susceptible a mejoras de software. Desarrollando en primera instancia un algoritmo que sea capaz de seguir una referencia dada y que esta referencia pueda cambiar sin necesidad de un retorno del vehículo.

Justificación e Importancia

El presente proyecto pretende encontrar una alternativa de solución a la vigilancia de un área protegida frente al acceso de elementos móviles no autorizados. Soluciones actuales a este problema requiere personal dentro de la zona y/o un sistema de monitorización fijo extendido por todo el terreno. La apuesta por la presente alternativa se centra en utilizar menos recursos para realizar el mismo trabajo. Se busca solución a esta problemática ya que la seguridad de un área es especialmente importante para la propiedad privada y mucho más si se trata en ambientes militares y/o educativos.

En pocas ocasiones la ciudadanía reporta casos relacionados a la inseguridad social, ya que la mayoría trata de pasarlos por desapercibidos debido a intimidaciones o miedo a represalias. Por estas razones nace el proyecto de implementar un VANT con visión artificial

integrada, con el fin de vigilar de manera autónoma y eficiente. Al ser un dispositivo aéreo, este puede permanecer vigilando las zonas en pleno vuelo, con la implementación de reconocimiento y seguimiento se puede detectar intrusos, así como, seguir la pista de los mismos. El desarrollo del presente, será de autoría, evitando dependencia de terceros, pensado para ser implementado en drones con especificaciones dadas por los autores.

Objetivos

Objetivo General

Investigar e implementar un sistema de aprendizaje sobrevuelo y seguimiento de objetos para mejorar la seguridad en ambientes externos mediante la utilización de IA en combinación con un dron, para la Universidad de las Fuerzas Armadas ESPE sede Latacunga.

Objetivos Específicos

- Investigar y recopilar información acerca de proyectos relacionados ejecutados con anterioridad para obtener información de procedimientos previos.
- Obtener conocimientos acerca del funcionamiento, mantenimiento, programación y configuración del procesador del Dron Bebop 2.
- Investigar y desarrollar los métodos y algoritmos que se pretenden utilizar en la captura, análisis, aprendizaje, detección y seguimiento de objetivos.
- Optimizar algoritmo para su próxima implementación
- Diseñar e implementar el Interfaz humano máquina (HMI) y la comunicación del sistema necesaria para el funcionamiento.
- Realizar pruebas y simulaciones del sistema previo a la implementación.
- Implementar algoritmo de reconocimiento y aprendizaje por fases.
- Realizar pruebas en conjunto para verificar y corregir errores.

Hipótesis

¿Un sistema de aprendizaje sobrevuelo en combinación con un dron ayudará a realizar un seguimiento de objetos de forma óptima a manera de mejorar la seguridad en ambientes externos para la Universidad de las Fuerzas Armadas ESPE sede Latacunga?

Variables de la investigación

Variable independiente

Sistema de aprendizaje sobrevuelo utilizando un VANT.

Variable dependiente

Seguimiento de objetivos de forma óptima a manera de mejorar la seguridad en ambientes externos para la Universidad de las Fuerzas Armadas ESPE sede Latacunga.

Capítulo II.

Marco Teórico

Visión Artificial

Se trata de una disciplina que integra varias técnicas de otros campos de estudio como las matemáticas, informática, álgebra lineal, IA, etc., con el objetivo de procesar imágenes digitales para obtener información de interés, analizarla y tomar decisiones en base a este análisis.

En general, una de las herramientas más útiles de la visión artificial, es el uso de técnicas de IA como las RNA. Las RNA juegan un importante rol al momento de realizar el análisis de la imagen digital, ayudándose siempre de una etapa anterior de procesamiento en que se utilizan técnicas clásicas de visión por computadora como segmentación de color, detección de bordes, detección de líneas verticales y horizontales, etc.

El avance de las técnicas de visión artificial y de las unidades de procesamiento digitales, permite actualmente realizar el análisis de fotografías digitales prácticamente en tiempo real, con un nivel de precisión aceptable. Existen diferentes algoritmos y bibliotecas de código abierto que ayudan a los usuarios a utilizar estas herramientas sin necesidad de ser un especialista y de esta manera, se sustenta una comunidad en la que se mejoran estos procesos continuamente.

Redes Neuronales Artificiales

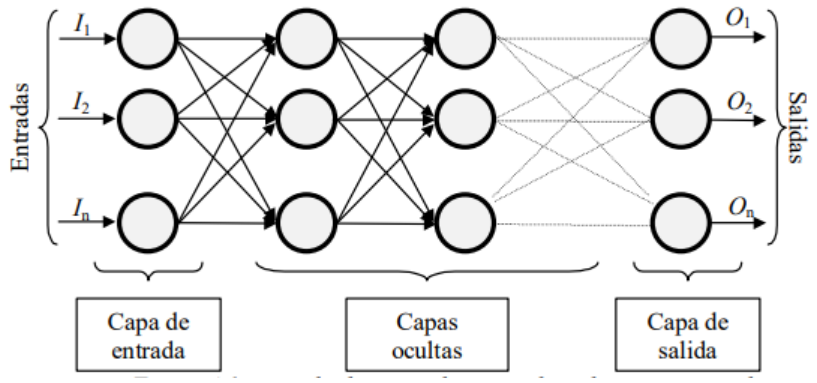
Las RNA, están inspiradas en las redes neuronales biológicas del cerebro humano. Están constituidas por elementos que se comportan de forma similar a la neurona biológica en sus funciones más comunes. Estos elementos están organizados de una forma parecida a la que presenta el cerebro humano. Las RNA al margen de “parecerse” al cerebro presentan una serie de características propias del cerebro. Por ejemplo, las RNA aprenden de la experiencia, generalizan de ejemplos previos a ejemplos nuevos y abstraen las características principales de una serie de datos.

La estructura general de una RNA, está organizada por nodos y flujos de información que se interconectan entre sí para llegar a un resultado. En sus distintas capas (Figura 1) se inicia un proceso diferente y cada capa tiene un objetivo:

- **Capa de entrada:** Adquiere las señales de entrada, donde la información se encripta para la red neuronal para próximamente procesarla.
- **Capas ocultas:** Se obtienen las características abstractas de la respuesta de la capa de entrada y genera descripciones para poder generar predicciones.
- **Capa de salida:** En esta capa se adquieren los resultados, donde existe una clasificación, interpretación y detección de lo obtenido en la señal de entrada.

Figura 1

Elementos básicos de una red neuronal artificial



Nota. Imagen obtenida de *(Matich, 2001, pág. 12)*

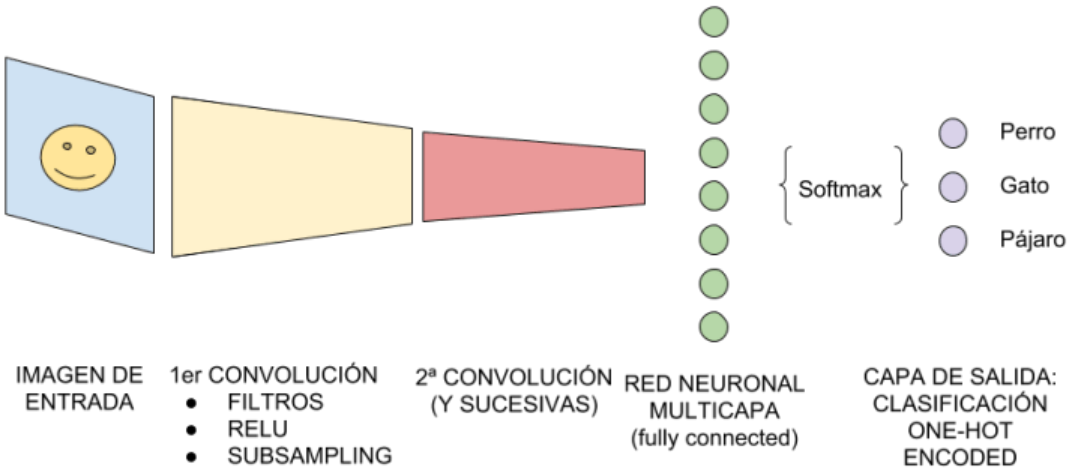
Redes Neuronales Convolucionales

Una red neuronal artificial convolucional (RNAC) es un tipo de RNA especializada para trabajar con imágenes en el campo de la visión artificial. Para poder reconocer objetos, la RNA convolucional aplica una serie de filtros mediante la convolución de un kernel sobre la imagen, de manera consecutiva e independiente. Los valores de cada kernel utilizado se obtienen a través del entrenamiento de la RNA. Según (Barrios Arce, 2020) Las RNACs, consisten en múltiples capas de filtros convolucionales de una o más dimensiones. Después de cada capa,

por lo general se añade una función para realizar un mapeo causal no-lineal. Como cualquier red empleada para clasificación, al principio estas redes tienen una fase de extracción de características, compuesta de neuronas convolucionales, luego hay una reducción por muestreo y al final tendremos neuronas de perceptrón más sencillas para realizar la clasificación final sobre las características extraídas, tal como se ilustra en la Figura 2.

Figura 2

Arquitectura de una RNA convolucional



Nota. Imagen extraída de *(Barrios Arce, 2020)*

Aprendizaje y Entrenamiento de una RNAC

El entrenamiento de una RNA corresponde al proceso mediante el cual se definen automáticamente los valores al utilizar en los kernels, con ayuda de ejemplos predefinidos. Es decir, se necesita un banco de ejemplos de cada clase para ingresar a la RNAC y, posteriormente, poder usar la red para que clasifique imágenes que no se le han mostrado antes. La regla general es que entre más amplio sea el banco de imágenes para su entrenamiento, tendrá una mayor probabilidad de acierto la predicción de la RNA. El entrenamiento se inicia con valores aleatorios, el cual, a través de diferentes técnicas, busca resultados cercanos o similares a los esperados. El proceso se puede explicar como una

función en 3 dimensiones, con un punto posicionado en el plano de manera aleatoria, donde al acercarse al punto de mayor depresión, se establecería en un entrenamiento óptimo.

Aprendizaje en Marcha

Para que una RNA pueda aprender mientras el sistema se encuentra en marcha, es necesario trabajar con programación concurrente, que permita ejecutar procesos en paralelo para que no existan conflictos o demoras en el funcionamiento del sistema.

Una técnica popular para el aprendizaje en marcha, es utilizar los denominados *Checkpoints*, que son una técnica de tolerancia a fallos para procesos de larga duración. El sistema retorna a un punto donde su aprendizaje fue el mejor, por lo que en el entrenamiento pueden añadirse otros ejemplos para reentrenar la red debido a que sus pesos son guardados en su mejor estado, los mismos que pueden ser utilizados para verificar otros ejemplos o ser reentrenados.

You Only Look Once (YOLO)

YOLO es un algoritmo de detección de objetos en tiempo real que utiliza un método que divide las imágenes en un sistema de cuadrículas y cada celda es responsable de detectar objetos dentro de sí misma (Ultralytics, 2020). Se basa principalmente en la idea de correspondencia entre las regiones de la imagen de entrada en una RNAC y las regiones en la salida de la misma. YOLO utiliza una RNA multicapa basada en “*Darknet*” y en la actualidad se presenta en su versión v5.

El algoritmo de YOLO fue presentado por primera vez en 2015 y significó un gran cambio en la detección de objetos en tiempo real, ya que hasta entonces los algoritmos existentes dedicados a esta tarea no resultaban óptimos en la práctica. Cabe mencionar que YOLO, al ser un algoritmo de detección, podría acoplarse incluso a cualquier otra RNAC fuera de *Darknet*. Así como también es posible entrenar la RNAC con cualquier objeto definido por el usuario.

Darknet

Se trata de un framework para RNAs, escrito en C y CUDA (Compute Unified Device Architecture), admite cálculo de GPU (Unidad de procesamiento gráfico) y de CPU (Unidad de procesamiento central). Darknet está inspirada en GoogleNet y se estructura de 24 capas convolucionales con funciones de activación LeakyReLu.

Es importante aclarar que Darknet ofrece un sistema de clasificación de objetos por imagen, pero no brinda posición ni tamaño de los objetos detectados. Con esta RNA se obtuvo buenos resultados en la detección y clasificación de objetivos en la escena; pero un punto importante y por el que se lo descartó, es que su tiempo de entrenamiento también es muy alto para la aplicación, tardando de entre 1 a 3 horas para realizar entrenamiento para una detección aceptable.

Programación Concurrente

La programación concurrente atiende a la necesidad de ejecutar un proceso o conjunto de procesos en tiempo real. Un sistema de tiempo real se define como “cualquier sistema en el que el tiempo en el que se produce la salida es significativo. Esto generalmente es porque la entrada corresponde a algún movimiento en el mundo físico, y la salida está relacionada con dicho movimiento. El intervalo entre el tiempo de entrada y el de salida debe ser lo suficientemente pequeño para una temporalidad aceptable”. (Daintith & Wright, 2008)

Existen diferentes técnicas para ejecutar una programación concurrente y sus definiciones varían ligeramente según el lenguaje de programación; sin embargo, el concepto es el mismo. Las técnicas más populares son la programación multihilo y el multiprocesamiento. El primero suele dividir el tiempo de procesamiento de un núcleo del procesador para que diferentes procesos se ejecuten de manera intermitente. Debido a que los intervalos de tiempos son muy pequeños, se tiene la percepción de que se están ejecutando los procesos en paralelo. La programación multiprocesos, hace uso de los núcleos del

procesador para poder ejecutar procesos en paralelo verdaderamente, esto está limitado estrictamente por el hardware del sistema.

Lenguaje de programación Python

Python es un lenguaje de programación relativamente joven, de alto nivel y de mucha demanda en la actualidad debido a su flexibilidad y fácil aprendizaje. Este lenguaje fue creado por Guido van Rossum a principios de los 90's y se desarrolla bajo una licencia Open Source avalada por OSI (modelo de interconexión de sistemas abiertos), por lo que existe una gran comunidad apoyando en su evolución, compartiendo conocimiento, experiencias y ayudando a todos los que desean iniciar en el aprendizaje de Python.

Python es utilizado activamente para diversos fines, entre los que se encuentran:

- Desarrollo web
- Desarrollo de escritorio
- Interfaz gráfica de usuario (GUI)
- Comunicación con bases de datos
- Análisis de datos
- Inteligencia artificial
- Científico y numérico
- Educacional

Sin duda alguna, el punto más fuerte de Python es la comunidad que lo respalda, ya que debido a esto se cuenta con gran información en la red, proyectos, bibliotecas, foros, recomendaciones, etc. Todo esto convierte a Python en un lenguaje óptimo y poderoso para trabajar en cualquiera de sus aplicaciones ya mencionadas.

Bibliotecas en Python

Una de las mayores ventajas de trabajar con Python, es la gran cantidad de bibliotecas disponibles, tanto propias de Python como de terceros. Para el caso concerniente, se están

utilizando bibliotecas correspondientes al campo de la inteligencia artificial, a la programación concurrente, GUI y al driver para controlar al VANT. Estas bibliotecas de programación servirán como pilares para el sistema que se pretende desarrollar. A continuación, se presentan brevemente algunas de las más importantes.

OpenCV

Contiene diversas funciones para el procesamiento digital de imagen y la visión artificial, permitiendo realizar operaciones a los mapas de bits ingresados y obtener información de interés (OpenCV, 2022).

PyQt5

Es un set de enlaces de Python para Qt v5. Permite que Python se use como un lenguaje de desarrollo de aplicaciones alternativo a C++ en todas las plataformas compatibles. Se utiliza para el desarrollo de la GUI y la programación multihilo. (Riverbank Computing, 2020)

Scikit-Image

Se trata de una biblioteca de código abierto para el libre uso de algoritmos de procesamiento de imágenes digitales, desarrollada por una comunidad de voluntarios.

Google Colaboratory

Google Colaboratory es un producto de *Google Research* que permite cualquier usuario con una cuenta de Google, ejecutar código Python a manera de un cuaderno *Jupyter.* Esto es posible gracias a que Google asigna una máquina virtual al usuario de forma dinámica para que el código ejecutado cuente con suficiente poder de procesamiento. Google asigna recursos de GPU o TPU (unidad de procesamiento tensorial) de manera automática y totalmente gratis, siendo una excelente herramienta para aprendizaje automático, análisis de datos y educación.

Vehículo Aéreo No Tripulado (VANT)

Son aquellos vehículos aéreos que no requieren de una persona a bordo para ser pilotados. También suelen ser llamados UAV o drones en el ámbito militar. Se utiliza el acrónimo RPA (Remotely Piloted Aircraft) en el ámbito civil. Todos estos nombres tienen sus

marcadas o sutiles diferencias, pero comparten su idea fundamental. En el presente trabajo se utilizará de manera general el acrónimo VANT para referirse a este tipo de vehículos y/o sistemas.

Los VANTs se pueden dividir inicialmente en dos grandes grupos: VANT de ala fija y VANT de ala rotatoria.

VANT de ala fija

La característica principal de estos VANTs, es que poseen un perfil alar que hace posible la sustentación del vehículo en el aire, tienen una arquitectura similar al de los aviones (Figura 3). Su mayor ventaja es la autonomía de vuelo, ya que su diseño aerodinámico les permite sustentarse en el aire por horas. Para su despegue y aterrizaje se necesita de una pista, análoga a las pistas de aviones. No es posible realizar vuelo estacionario con ellos, por lo que sus aplicaciones entran más en el reconocimiento de grandes áreas, fumigaciones, vuelos de larga distancia, etc.

Figura 3

VANT de ala fija de la empresa Embention, modelo F350G

Nota. Imagen tomada de *(Embention, 2022)*

VANT de ala rotatoria

Estos vehículos mantienen su vuelo gracias a la velocidad y fuerza en sus rotores, que hacen mover unas hélices que le permiten sustentarse en el aire. Su arquitectura se parece más a la de los helicópteros, aunque su principal diferencia con ellos, es que los VANT de ala rotatoria suelen tener entre 4, 6 u 8 rotores.

Son los VANT más extendidos comercialmente, dedicados a las tareas de vigilancia, reconocimiento, seguimiento y cualquier aplicación que requiera un control de vuelo estacionario y versátil. Su principal inconveniente es su baja autonomía, ya que debe alimentar a múltiples rotores para realizar su vuelo. Suele acompañarse de una batería integrada, y debe mantenerse un equilibrio entre el peso de la batería y su capacidad de almacenamiento para la alimentación de los rotores.

Figura 4

VANT de ala rotativa de la empresa Embention, modelo M600WP

Nota. Imagen tomada de *(Embention, 2022)*

Parrot Bebop 2

Se trata de un modelo de VANT con 4 alas rotatorias, también llamado “cuadricóptero”. Ofrece una capacidad de vuelo en promedio de 25 minutos y cuenta con una cámara integrada con lente tipo ojo de pez. Este VANT es, en general, robusto ante las corrientes bruscas de viento durante el vuelo y ofrece unas características generales que permiten la integración de este cuadricóptero en el presente proyecto. Más adelante se evaluarán sus características técnicas para su selección.

Figura 5

Cuadricóptero Parrot Bebop 2

Nota. Obtenido de (Apple, s.f.)

Movimientos de un VANT de cuatro hélices

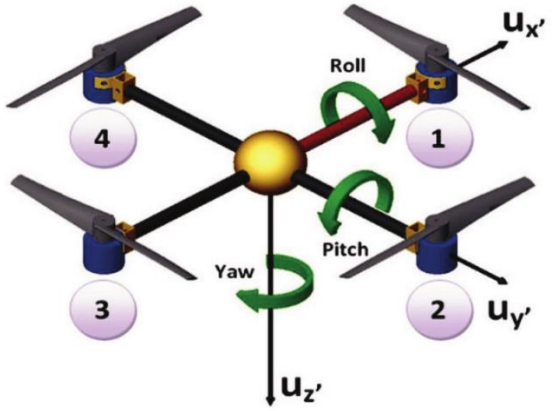
Este tipo de VANT posee cuatro rotores (Figura 6):

- Rotor 1: frontal
- Rotor 2: lateral derecho
- Rotor 3: posterior
- Rotor 4: lateral izquierdo

Los movimientos combinados de estos rotores dan lugar a 4 movimientos principales que puede ejecutar el VANT, explicados en la siguiente sección.

Figura 6

Tipos de rotación de un VANT de cuatro hélices



Nota. Imagen obtenida de *(Kumar, 2015)*

Movimiento de alabeo (Roll ϕ)

Se consigue alterando las velocidades de los motores 2 y 4, manteniendo las velocidades de los motores 1 y 3 invariantes, produciendo una rotación alrededor de u_x (Figura 6) y, por consiguiente, moviendo el VANT de izquierda a derecha o viceversa.

Movimiento de cabeceo (Pitch θ)

Se consigue alterando las velocidades de los motores 1 y 3, manteniendo las velocidades de los motores 2 y 4, produciendo una rotación alrededor de u_y (Figura 6) y, por consiguiente, moviendo el VANT de adelante atrás o viceversa.

Movimiento de guiñada (Yaw ψ)

Se incrementa la velocidad de los rotores 1 y 3 mientras se disminuye la de los rotores 2 y 4. Esto produce un desbalance aerodinámico que permite que el VANT gire sobre el eje u_z.

Movimiento vertical (Thrust)

Este movimiento se observa al incrementar o reducir la velocidad de los cuatro motores en igual magnitud generando un desplazamiento vertical en el eje u_z equivalente a la velocidad de los motores.

Capítulo III.

Métodos

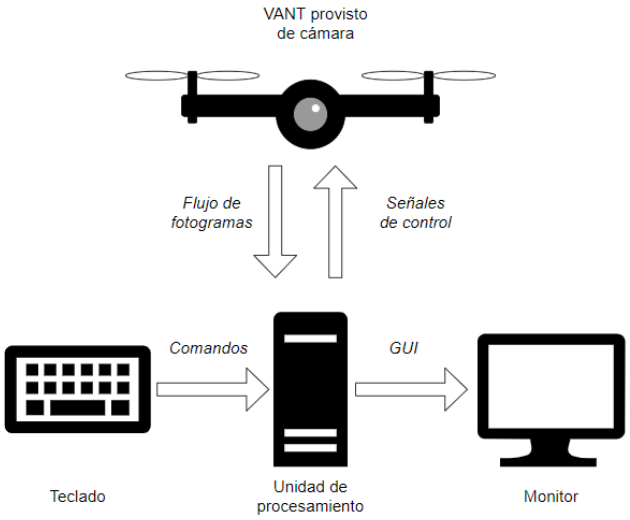
Arquitectura del Sistema

Arquitectura Física

El sistema debe estar conformado esencialmente de un VANT provisto de una cámara, este será el agente que se desplace por el espacio con el objetivo de tomar fotogramas a tiempo real de su entorno. Luego, estas imágenes deben ser procesadas, por lo que también se deberá disponer de una unidad de procesamiento (UP) con el poder computacional suficiente para realizar el análisis de las imágenes y que, posteriormente, envíe señales de control de movimiento al VANT. También será necesario un teclado para la entrada de comandos al sistema y una pantalla que permita al supervisor monitorizar el sistema. Todo tal como se esquematiza en la Figura 7.

Figura 7

Arquitectura física del sistema



Arquitectura del software

En función a la necesidad de mantener conexiones abiertas, envíos de señales y recepción – muestreo de fotogramas, se requiere de una descomposición modular del software

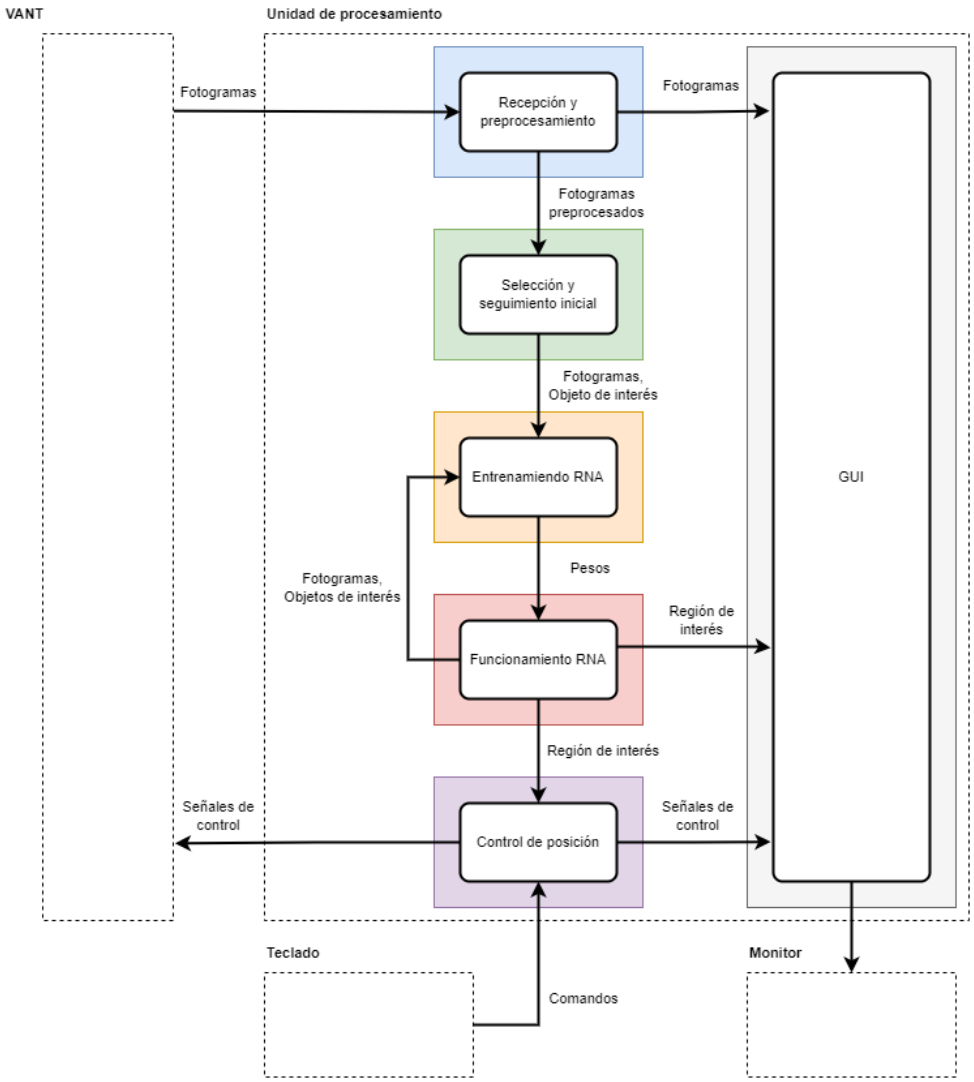
que se ejecutará en la UP. Separando cada una de las etapas del procesamiento de las imágenes de la siguiente manera:

- Recepción de fotogramas
- Selección inicial de objetivo
- Algoritmo de seguimiento inicial
- Guardado y etiquetado de fotogramas para el entrenamiento
- Entrenamiento de RNA en paralelo
- Algoritmo de seguimiento usando la RNA

El diseño estructurado del software requerido se detalla en la Figura 8.

Figura 8

Arquitectura de software del sistema



Consideraciones Importantes y Reestructuración del Sistema.

Hasta este punto, existe una discrepancia importante entre la hipótesis planteada y la propuesta de funcionamiento del sistema. Según las investigaciones realizadas con respecto al tiempo de entrenamiento y características del tiempo de vuelo del VANT, la propuesta se reestructura bajo la opción de un entrenamiento local o un entrenamiento remoto.

En este contexto, se propone el uso de una máquina en la nube con un buen poder de procesamiento para realizar la tarea de entrenamiento, o en su defecto, realizar un

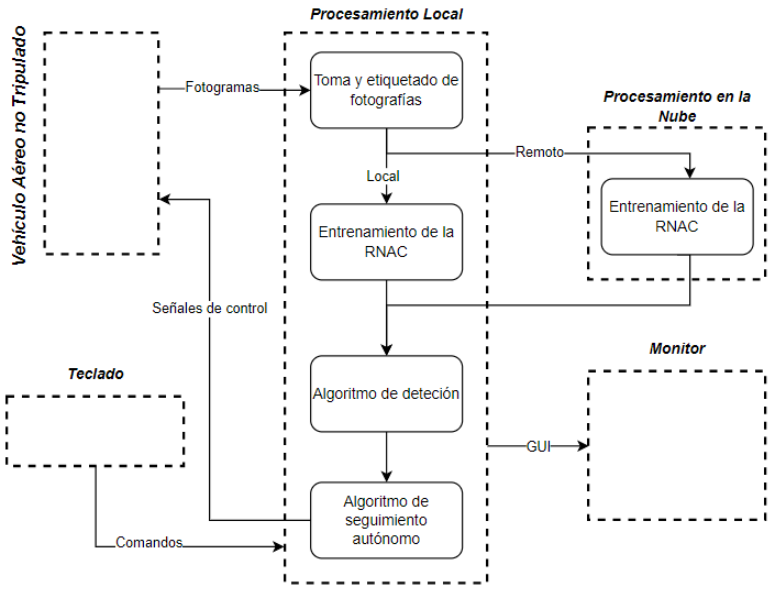
entrenamiento en una máquina local; pero en ambos casos, se requiere que el entrenamiento se realice antes de poner en marcha el vuelo automático para seguimiento. Por estos motivos se necesita una nueva división en etapas para el sistema, tal como sigue:

- Toma de fotos y etiquetado para entrenamiento
- Entrenamiento (local o en la nube)
- Carga del modelo al sistema
- Vuelo de seguimiento automático

Estas etapas se esquematizan en la Figura 9

Figura 9

Arquitectura de software reestructurada



Selección de Componentes y Herramientas

Para realizar la selección de componentes y herramientas del sistema, se utiliza el método de selección cualitativo por puntos (Tabla 1). Este método permite asignar valores cuantitativos a características subjetivas de la alternativa de importancia para su uso (Sapag Chain, Sapag Chain, & Sapag Puelma, 2014, pág. 156). El procedimiento es el siguiente:

1. Se definen los factores, que son características decisivas para cada conjunto de alternativas.
2. A cada factor se le asigna un peso, que es una ponderación relativa entre ellos, con la base de que la suma de los pesos debe ser igual a 1.
3. Se evalúa cada alternativa para cada factor dentro de una escala predefinida (comúnmente de 0-10).
4. Se pondera la evaluación de cada alternativa, multiplicándola por el peso del factor asociado.
5. La alternativa seleccionada será aquella cuya suma de ponderaciones sea mayor.

A continuación, se procede a mostrar un ejemplo del procedimiento, teniendo en cuenta 3 factores de análisis y 3 alternativas a ser evaluadas.

Tabla 1

Ejemplo de matriz de selección del método cualitativo por puntos

Factor	**Peso**	**Alternativa A**		**Alternativa B**		**Alternativa C**	
		Eval.	Pond.	Eval.	Pond.	Eval.	Pond.
Factor 1	0.3	5	1.5	7	2.1	3	1.5
Factor 2	0.25	6	1.5	4	1	9	2.25
Factor 3	0.45	3	1.35	8	3.6	6	2.7
TOTAL	1		4.35		6.7		6.45

Existen elementos que se seleccionan principalmente por su disponibilidad inmediata o de su bajo costo de adquisición para ser utilizados dentro del proyecto. Para estos componentes se justificará su uso debidamente.

VANT

El VANT a utilizarse en el proyecto será un cuadricóptero marca Parrot, modelo Bebop 2 (Figura 10). Se ha elegido este VANT ya que existe disponibilidad del mismo dentro de la Universidad de las Fuerzas Armadas ESPE Sede Latacunga para desarrollos de trabajos de titulación como este, y más importante, cuenta con las características necesarias para el desarrollo del sistema, detalladas en la Tabla 2.

Figura 10

VANT Parrot Bebop 2 color rojo

Nota. El modelo y color del VANT de la imagen es el mismo con el que se trabajará. Obtenido de *(Apple, s.f.)*

Tabla 2

Características técnicas del VANT Parrot Bebop 2

Características técnicas	
Alto	8.9 cm
Ancho	38.2 cm
Longitud	32.8 cm
Peso	500 g
Batería	2700 mAh
Resolución de imagen	14 Mpx

Características técnicas	
Autonomía	25 min.
Estabilización de imagen	3 ejes
Almacenamiento	8 GB.
Alcance	2 km

Nota. Características recopiladas de (Apple, s.f.)

Unidad de Procesamiento Local

Para la selección de la UP local se cuenta principalmente con 3 alternativas: Una computadora de escritorio, computadora portátil, Módulo Raspberry. La tabla de selección se muestra a continuación:

Tabla 3

Matriz de selección para la UP

Factor	Peso	PC		Laptop		Raspberry	
		Eval.	Pond.	Eval.	Pond.	Eval.	Pond.
Poder de procesamiento	0.60	9	5.4	9	5.4	6.5	3.9
Portabilidad	0.15	3	0.45	9	1.35	8	1.2
Precio	0.25	7	1,75	4	1	10	2.5
TOTAL	1		7.6		7.75		7.6

Como indican los resultados, la mejor alternativa según los factores evaluados, es la computadora portátil, también es importante destacar que esta integra tanto una pantalla como un teclado en el mismo dispositivo, por lo que también es un ahorro de recursos. La laptop disponible es una marca Acer, modelo Nitro 5 AN515-55, cuyas características técnicas se detallan en la Tabla 4.

Las siguientes selecciones se realizarán del mismo modo y se indicará en la nota de la tabla la opción escogida.

Tabla 4

Características técnicas de la unidad de procesamiento

Características técnicas	
Procesador	Intel Core i7-9750H de 6 núcleos (hasta 4.5 GHz)
RAM	16 GB DDR4 2666 MHz
Coprocesador de gráficos	NVIDIA GeForce GTX 960
Disco duro	Flash Memory Solid State

Procesamiento en la nube

El procesamiento en la nube se pretende utilizar solo para la etapa de entrenamiento de la RNAC, ya que para todo lo demás, es necesaria la conexión cercana entre el procesador y el VANT. Así, se ha optado por utilizar el servicio de Google Colaboratory para el cómputo del entrenamiento de la red, ya que con este servicio se puede acceder de manera gratuita a una máquina de Google en la nube, utilizando procesamiento por GPU y una cantidad de memoria RAM dinámica que depende del código ejecutado.

Sistema Operativo

Para el sistema operativo se tienen como alternativas a Windows y a Ubuntu. A continuación, la tabla se selección:

Tabla 5

Matriz de selección para el sistema operativo

Factor	Peso	Windows Lite		Ubuntu	
		Eval.	Pond.	Eval.	Pond.
Velocidad en multitarea	0.3	6	1.8	9	2.7
Seguridad	0.1	10	1	5	0.5
Estabilidad	0.1	9	0.9	6	0.6
Soporte a la GPU	0.5	10	5	6	3
TOTAL	1		8.7		6.8

Nota. Se selecciona a Windows Lite como sistema operativo para trabajar

Como se indican en los resultados, la mejor alternativa para el proyecto es Windows lite, cabe recalcar que la elección se basa especialmente en el soporte para la GPU y la velocidad ante procesos multitarea. Por lo cual se utilizará Windows LTSC 10.

Bibliotecas para desarrollo de software

Dentro de las bibliotecas que se requieren utilizar, se destaca OpenCV, la cual es una biblioteca de código abierto orientada a la visión artificial y al aprendizaje automático (OpenCV, 2022). A demás de esta, se utilizará TensorFlow para el aprendizaje automático de una de las redes neuronales del sistema. Se eligen estas bibliotecas de antemano porque se cuenta con experiencia previa en las mismas y se conoce que son de mucha utilidad y muy populares en el ámbito de la visión artificial y el aprendizaje automático.

Lenguaje de Programación

Actualmente existen decenas de lenguajes de programación según la aplicación que se requiera; sin embargo, en este caso, la utilización de OpenCV condiciona las alternativas para la selección del lenguaje, las cuales son: Java, C++, Python.

Tabla 6

Matriz de selección para el lenguaje de programación

Factor	Peso	Java		C++		Python	
		Eval.	Pond.	Eval.	Pond.	Eval.	Pond.
Simplicidad de sintaxis	0.2	5	1	5	1	10	2
Velocidad de ejecución	0.4	7	2.8	9	3.6	8	3.2
Documentación y bibliotecas	0.4	9	3.6	10	4	9	3.6
TOTAL	1		7.4		8.6		8.8

Nota. Se selecciona a Python como lenguaje de programación para trabajar

Respecto a los resultados, se selecciona a Python como lenguaje de programación a utilizar, debido a la facilidad y la comunidad que apoya en los temas de machine learning, redes neuronales, y análisis de datos.

Driver para el VANT

Para poder controlar al Bebop 2 mediante un software, se necesita de un driver que comunique la parte lógica con la parte física. Este driver deberá ser compatible para ser llamado mediante comandos o bibliotecas en Python. Según la literatura recabada, se tienen dos principales opciones: la primera es la utilización de la plataforma para desarrollo de proyectos robóticos ROS, el cual cuenta con una biblioteca llamada bebop_autonomy, la cual permite controlar los VANT bebop 1 y 2, y se basa en el ARDroneSDK3 oficial de Parrot (Monajjemi & Bamford, 2015).

La segunda opción es utilizar pyparrot, una biblioteca de Python desarrollada por la Dra. Amy McGovern. Esta biblioteca permite programar VANTs Parrot Minidrone, Mambo, Swing y Bebop 1 y 2 (McGovern, 2018).

Tabla 7

Matriz de selección para el driver del VANT

Factor	Peso	ROS		pyparrot	
		Eval.	Pond.	Eval.	Pond.
Facilidad de uso	0.1	6	0.6	10	1
Focalización de propósito	0.3	3	0.9	10	3
Personalización	0.3	10	3	10	3
Estabilidad	0.3	8	2.4	7	2.1
TOTAL	1		6.9		9.1

Nota. Se selecciona a pyparrot como el driver para trabajar con el VANT Bebop 2

Con base *en* los resultados, se selecciona a pyparrot como driver para el manejo del VANT, debido a la facilidad en la modificación en su código fuente y la simplicidad de las acciones que conlleva a invocaciones simplificadas y entendibles.

Algoritmo de detección de objetos

Para la detección de objetos en tiempo real, actualmente se tienen algunas alternativas tanto gratuitas como de pago. Sin embargo, dada la familiaridad que se tiene con YOLO, uno de los mejores algoritmos para detección de objetos en tiempo real en la actualidad, se ha elegido trabajar con el mismo. YOLO se presenta al día de hoy en su última versión v5 y es totalmente libre para ser utilizada por cualquier desarrollador.

La ventaja principal de YOLO es su velocidad de trabajo, pudiendo detectar objetos hasta en 60 fps, Así mismo, en su versión v5 presenta mayor precisión en sus detecciones, pudiendo detectar varios objetos dentro de una escena. Este algoritmo trabaja con una RNA convolucional preentrenada que se puede personalizar para entrenarla según los objetos que el usuario desee detectar.

Toma y Etiquetado de Fotografías

El proceso de toma y etiquetado de fotos, puedes volverse largo y tedioso por la gran cantidad de datos necesaria para el entrenamiento. Según la experiencia previa, se necesita mínimo de entre 500 a 1000 imágenes para un reconocimiento aceptable. Hacer esto de forma manual, es decir, imagen por imagen, llevaría horas frente al computador. Con el objetivo de ahorrar tiempo para este paso, se ha implementado un algoritmo de seguimiento inicial, el cual sigue a un objetivo dentro de un video siempre que se encuentre en un entorno cercano y no se desplace con rapidez. Para el desarrollo del algoritmo, se cumplen los siguientes partes.

Preprocesamiento del fotograma

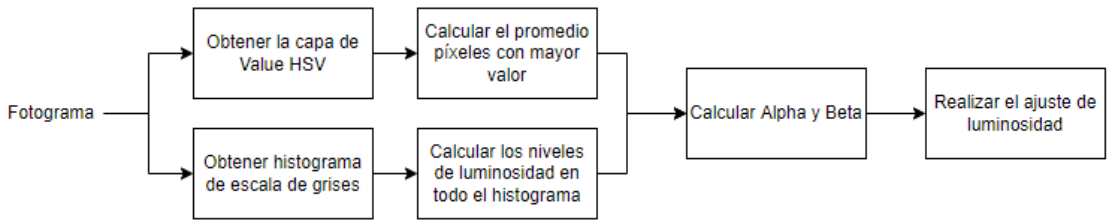
Debido a que el sistema trabajará en un ambiente no controlado, los fotogramas obtenidos del Bebop pueden estar sujetos a variaciones de luminosidad, por lo que es necesaria una etapa de preprocesamiento del fotograma, donde se controlen los niveles de brillo y contraste de manera automática. Para esto, se utiliza una transformación lineal a nivel de pixeles, como se indica en la ecuación (1), en donde los parámetros $\alpha > 0$ y β, llamados ganancia y polarización, controlan el brillo y el contraste de cada pixel, respectivamente. (Szeliski, 2021, págs. 111-112).

$$g(x) = \alpha f(x) + \beta \quad \textbf{(1)}$$

El algoritmo implementado para esta etapa se presenta en el siguiente diagrama de bloques:

Figura 11

Diagrama de bloques para la compensación de brillo



Este algoritmo permite ponderar la cantidad de luz, para próximamente realizar un ajuste en la imagen a partir de los parámetros α y β. Lo que, en ciertos casos de contraluz, ayuda a calibrar el brillo y contraste para mejorar la detección de objetos.

Detección Inicial Mediante Características Ponderadas

Para conseguir que el VANT capture imágenes de un objetivo dado inicialmente, se debe ubicar al objetivo dentro del entorno visual de la cámara. Debido a que desde un inicio no se conoce nada acerca del objeto a identificar, es necesario que, a partir de la primera selección, el VANT sea capaz de obtener características determinantes que le permitan filtrar el objetivo de los demás elementos del entorno y centrarse en ello.

Recabando información dentro de la literatura, se tomó como primera opción la utilización de algoritmos clásicos de detección de características en imágenes, como lo son SIFT, SURF u ORB. Cada uno de estos algoritmos tiene su respectiva fundamentación matemática; pero todos hacen esencialmente lo mismo: Encontrar características fuertes dentro de una imagen dada. Estas características deben tener la particularidad de que, si el mismo objeto es rotado, escalado y/o movido dentro de una escena, aún puedan ser reconocibles y detectadas por el algoritmo.

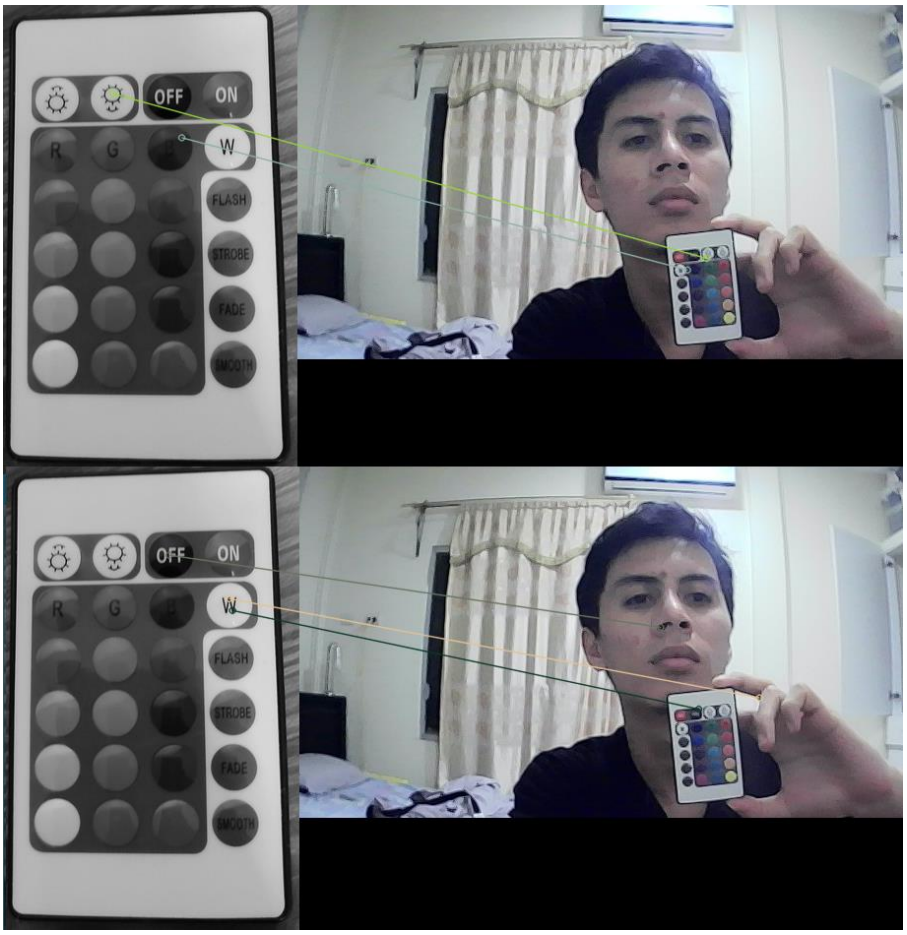
Scale Invariant Feature Transform (SIFT) es un detector de características desarrollado por Lowe en 2004. Aunque SIFT es un algoritmo fuerte al momento de reconocer objetos, tiene un coste computacional elevado, es por eso que se han desarrollado variantes a partir de este algoritmo para mejorar su rendimiento. Tal es el caso de ORB, publicado en 2011 por Ethan Rublee; Vincent Rabaud; Kurt Konolige y Gary Bradski. Se trata de una alternativa eficiente ante SIFT o SURF.

Utilizando ORB para ubicar características de un objetivo dado, dentro de una escena, se presentaron inconvenientes, el principal fue la detección de falsos positivos dispersos por toda la escena (véase Figura 12). ORB trabaja muy bien cuando se busca un objetivo que

posea muchas características distintivas en su superficie; sin embargo, cuando se trabaja con objetivos en los que no se encuentren muchas de estas características, la detección suele fallar al confundir las características del objeto con otras encontradas en el entorno. Por tanto, se desistió de trabajar con un detector de características de este tipo, ya que no resultó confiable para la aplicación.

Figura 12

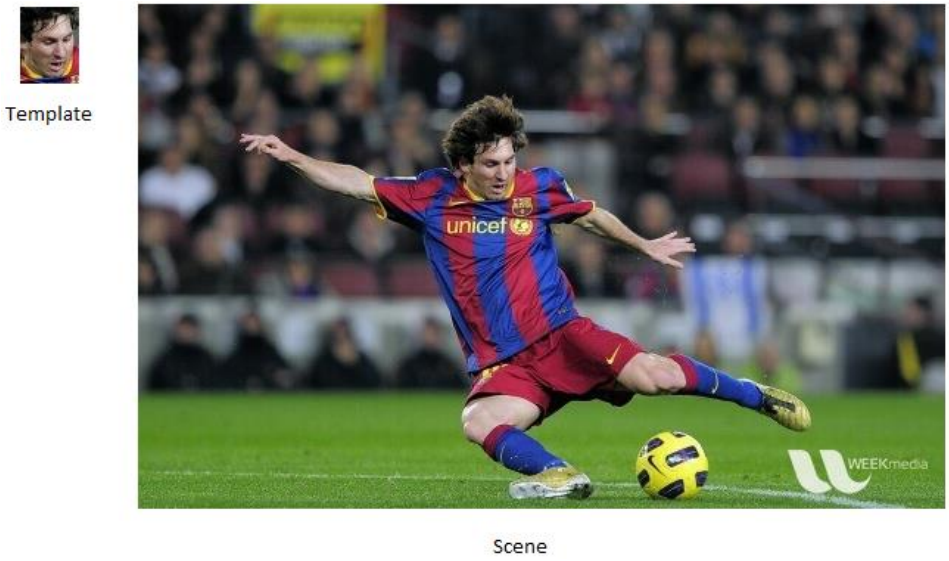
Pruebas de detección de objetos realizadas con el algoritmo ORB



Posteriormente, se probó trabajar con una técnica de emparejamiento de plantilla. Esta técnica realiza una convolución entre dos matrices: la plantilla y la escena, como por ejemplo las mostradas en la Figura 13.

Figura 13

Ejemplo de plantilla utilizada en técnicas de emparejamiento



Nota. Obtenido de (OpenCV, s.f.)

La biblioteca de OpenCV ofrece un método denominado *matchTemplate()*, el cual devuelve una tercera matriz producto de la convolución de las dos matrices de entrada. Esta tercera matriz presenta los puntos en donde la imagen de escena se asemeja más a la imagen dada de la plantilla, como se puede apreciar en la Figura 14

Figura 14

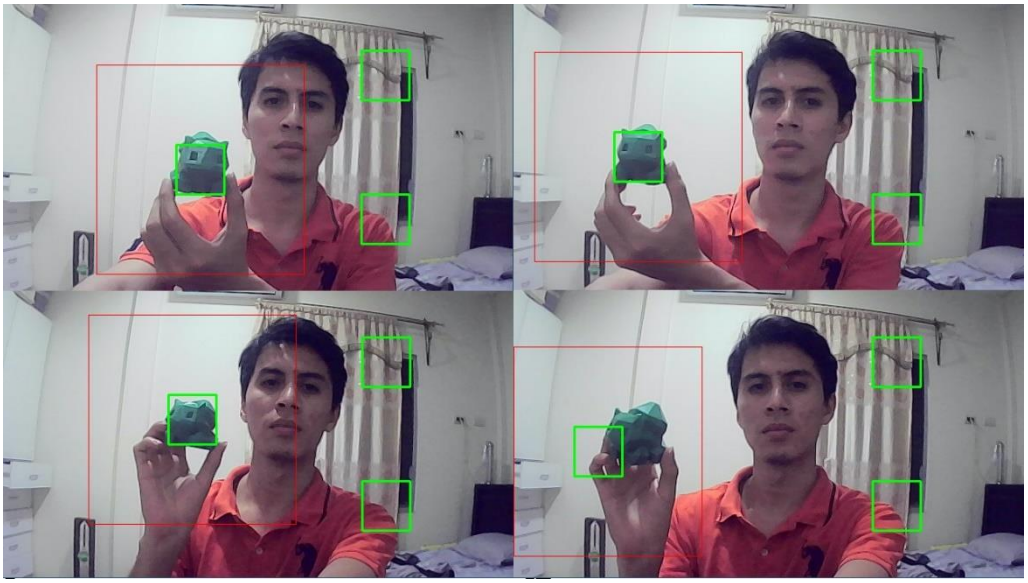
Ejemplo resultado de emparejamiento de plantilla con OpenCV

Nota. Obtenido de (OpenCV, s.f.)

Esta técnica detecta muy bien un objeto en video cuando la plantilla se refresca cada fotograma. Se obtuvo buenos resultados al realizar pruebas con el VANT; sin embargo, se presentaba problemas en los casos que había mucho movimiento en la escena o del objeto: el *target* marcado se corría de a poco a medida que el objetivo se movía lo mínimo.

Figura 15

Pruebas de detección de objetos con emparejamiento de plantilla



Dados los inconvenientes presentados, se tuvo que desarrollar un algoritmo personalizado para la detección inicial del objeto en la escena. Para el seguimiento de un target dado inicialmente, se utilizó el método de Lucas-Kanade para resolver el flujo óptico y encontrar el objeto en cada fotograma. Luego, para estar seguros de que el target en el nuevo fotograma sea el mismo que el del fotograma anterior, es decir, para reducir posibles falsos positivos en el seguimiento, se realiza una comparativa entre los targets tomados. El parecido entre los targets se lo evalúa mediante un promedio, el cual recibe como entradas el resultado de 3 métodos de comparación de imágenes:

- Comparación de histogramas

- Comparación por convolución
- Comparación por hash (función criptográfica) perceptual.

Explicando un poco más acerca de la función criptográfica hash, esta se utiliza particularmente cuando se transforma un objeto, sea una variable o un documento, denominado bloque de datos en una codificación de longitud fija. En este punto podríamos comparar dos imágenes por la semilla generada por esta conversión, realizando una comparación de cada uno de sus caracteres, obteniendo un porcentaje de similitud.

El promedio obtenido devuelve un porcentaje de similitud entre el último target capturado y el nuevo, si el porcentaje pasa el umbral designado; entonces se toma el nuevo target como verdadero, caso contrario, se realiza una búsqueda usando el método de emparejamiento de plantillas alrededor del último target ubicado. Para esto, se toman los últimos 5 targets aceptados anteriormente como plantillas y se empieza una búsqueda iterada dentro de la región cercada. A continuación, se muestra un esquema donde se explica de una forma más detallada este proceso.

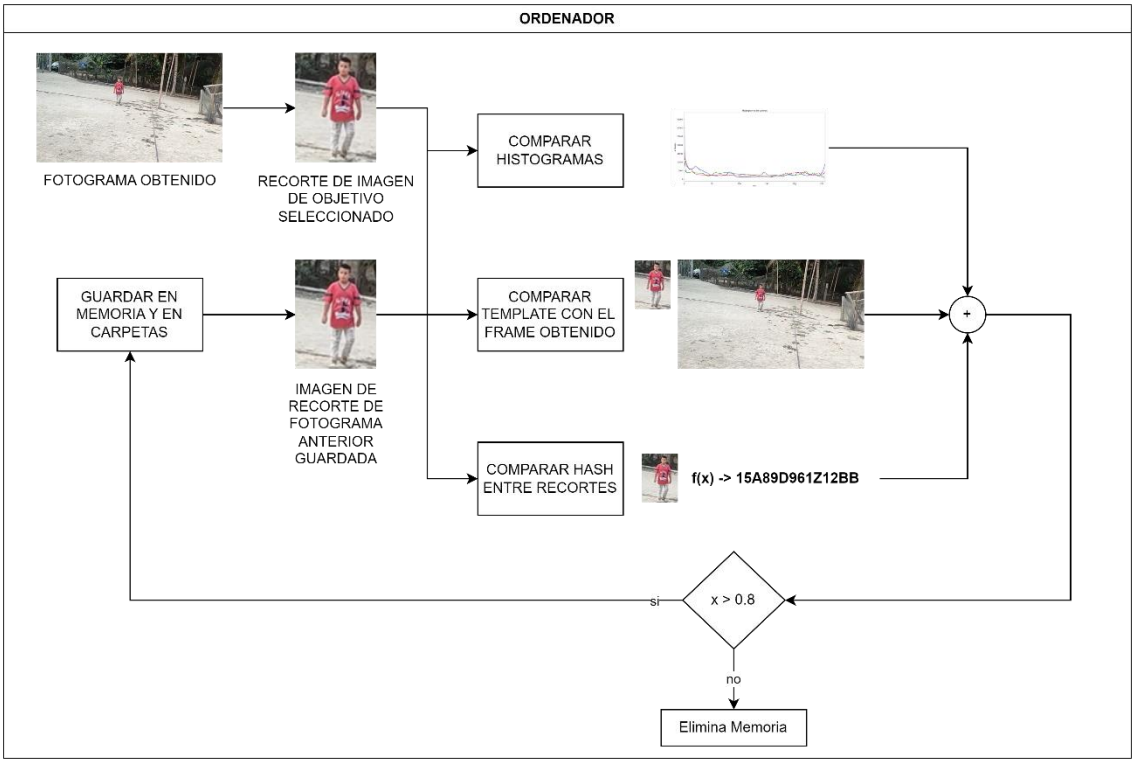
Figura 16

Flujo de información entre el VANT y el Procesador

Como se menciona anteriormente, el dron envía los fotogramas y recibe órdenes de movimiento, en el caso de que estos se envíen.

Figura 17

Algoritmo inicial de obtención de recorte de imagen de un objetivo.



Como se observa, el método obtiene la imagen y a través de Lukas Kanade, se obtiene la posición del objetivo según el flujo óptico, esto dando como resultado un recuadro del objeto seleccionado en la imagen capturada posterior. En los siguientes pasos se obtienen los histogramas de ambas imágenes y se comparan, así como la plantilla y el hash. Próximamente se obtiene un promedio, se decide si cumple con el umbral y se guarda la imagen capturada. Todo este proceso es necesario, debido a que no podemos ingresar datos que no correspondan al objeto que se va a entrenar, en el caso que este proceso no se considere tomamos el riesgo de ingresar ruido a la red neuronal aumentando las posibilidades de fallo en el momento de la detección.

Para entendimiento más técnico, se detallan los pasos del algoritmo a continuación:

- Se obtienen los histogramas de cada imagen, con la librería de OpenCV, método calcHist, y se comparan obteniendo un valor de 0 a 1.
- Se realiza la comparación por plantillas, esto se realiza con la librería de OpenCV el método matchTemplate, obteniendo un valor de 0 a 1.
- Se realiza la comparación con el método de hash, obteniendo un valor de 0 a 1.
- Realizado este procedimiento se obtiene un promedio de estos valores, el cual, si cumple con las especificaciones de umbral, se procede a guardar la imagen para ser comparada próximamente en memoria y en carpeta local.
- De esta manera cada sujeto obtenido en la imagen, pasa por este proceso en el cual se obtiene la imagen con mayor similitud.
- Cabe recalcar que el algoritmo solo toma en cuenta, forma, color y tamaño, por lo que es susceptible a errores. Como los siguientes:
    - Al ser un código que compara las secciones obtenidas, en el caso que el sujeto ejemplo desaparece de la escena, cualquier otro sujeto detectado puede ser tomado por el algoritmo como el más cercano a similitud.
    - Los cambios bruscos de brillo pueden afectar al algoritmo, aunque por esta misma razón la ponderación de histograma, equivalente al color de imagen, tiene una asignación menor.
- Algunas de las ventajas del algoritmo son las siguientes:
    - Puede ser susceptible a cambios o modificaciones.
    - La velocidad de procesamiento no se ve muy afectada.
    - Tiene un buen porcentaje de asertividad.
    - Pueden compararse varias imágenes, en el caso que exista una inconsistencia de valores, no se da un resultado preciso y el algoritmo sigue trabajando hasta encontrar la mejor opción.

Recopilación de datos para el entrenamiento

La identificación y seguimiento del objeto con ayuda de Lucas-Kanade, desemboca en la captura de datos para el entrenamiento posterior de la RNA. Cada imagen debe tener su correspondiente archivo de texto que contenga las coordenadas del objeto dentro de la escena. El formato para el etiquetado de cada imagen se presenta en la página oficial de YOLO. Las especificaciones del archivo .txt son las siguientes:

- Ya que solo se permite etiquetar un objeto por fotograma, cada archivo de texto debe contener una sola fila sin saltos de línea.
- La fila debe tener el formato: *0 X_centro Y_centro Ancho Alto.* El “0” al inicio indica que se trata de la primera y única clase.
- Los valores de *X_centro, Y_centro, Ancho* y *Alto*, deben estar normalizados entre 0 y 1, en correspondencia al ancho o alto total de la imagen, respectivamente.
- El sistema de coordenadas de referencia para obtener los valores de *X_centro* y *Y_centro* tiene su origen en el pixel superior izquierdo de la imagen, con valores X+ a la derecha y Y+ hacia abajo (véase
-
- Figura 18).

Figura 18

Sistema de coordenadas para el etiquetado usado en YOLO

Nota. Obtenido de (Ultralytics, 2020)

Entrenamiento de YOLO

Esta es una parte importante del proyecto ya que en este punto se tienen dos alternativas mencionadas con anterioridad: Realizar un entrenamiento local o un entrenamiento remoto. A priori se esperaría que realizar el entrenamiento con los servidores de Google resulte en un menor tiempo de ejecución que hacerlo en un entorno local. No obstante, el sistema se diseñará para que ambas alternativas queden a libre elección del usuario.

El archivo de ejecución para el entrenamiento es el mismo que el de YOLO en esencia, pero se debe tener en cuenta que los datos de entrenamiento deben en la carpeta predefinida como “FULL” y copiar un cuarto de los datos en la carpeta predefinida como “FULL_VAL”. Esta última requiere estos datos al azar para realizar el proceso de validación en el entrenamiento.

Revisión y Ajuste de Pyparrot

Pyparrot es una biblioteca con los comandos principales para controlar al Bebop de manera autónoma, los métodos definidos y su manual de uso se puede ver en su documentación oficial (McGovern, 2018). No obstante, se ha encontrado necesario modificar un par de cosas dentro de los archivos de pyparrot para poder trabajar de manera óptima al momento de desarrollar el software.

Recepción de Fotogramas

El primer comando probado dentro de la documentación de pyparrot, fue la conexión y la recepción de fotogramas en tiempo real. Para esto, una vez que se establece la conexión con el VANT, se llama al método bebop.start_video_stream(), que envía una señal al Bebop para que empiece a trasmitir el video. Es importante también llamar al método bebop.set_video_framerate('30_FPS'), esto con el objetivo de que la tasa de envío de frames sea la más alta posible, ya que posteriormente se analizarán estos fotogramas y se necesita un flujo constante.

El principal problema encontrado, fue que, al momento de receptar y presentar los fotogramas, estos se distorsionaban en gran medida por lapsos de tiempo de entre 5 a 15 segundos, lo cual no es tolerable para la aplicación. Se buscó solucionar esto reduciendo la tasa de fotogramas, aumentando el buffer de recepción, modificando el formato de entrada, entre otras muchas opciones; sin embargo, se encontró solución estableciendo una serie de parámetros que permite al usuario modificar como OpenCV recibe los datos multimedia de entrada.

OpenCV trabaja conjuntamente con el software de decodificación multimedia ffmpeg, por tanto, ciertos parámetros de ajuste también se pueden aplicar a la librería de visión por computadora. Todas estas opciones de configuración se pueden encontrar en la documentación oficial de ffmpeg (ffmpeg, 2022). Así, el código correspondiente a la configuración para una recepción estable de fotogramas, queda de la siguiente manera:

```
import cv2
import os
os.environ["OPENCV_FFMPEG_CAPTURE_OPTIONS"] = "protocol_whitelist;file,rtp,udp |
flag;low_delay | err_detect;explode | probesize;64 | fflags;discardcorrupt | max_delay;50 |
analyzeduration:3000000 | chunk_duration;500000 | vcodec;libx264 | flags2;showall |
avioflags;direct | flush_packets;1 | output_ts_offset;0 | rtsp_flags;listen | rtsp_transport;udp"
from pyparrot.Bebop import Bebop
import tracking

if __name__ == "__main__" :
        bebop = Bebop('Bebop2', '192.168.42.1')
        success = bebop.connect(5)
        if (success):
                bebop.set_video_stream_mode('low_latency')
                bebop.set_video_framerate('30_FPS')
                bebop.start_video_stream()
```

Comandos de Vuelo

Dentro de los métodos destinados al control de vuelo en pyparrot, se encuentran dos principales para el desplazamiento del Bebop:

- fly_direct(roll, pitch, yaw, vertical_movement, duration) Pilota al Bebop usando los movimientos de balanceo, cabeceo, guiñada y movimiento vertical especificados. Los comandos se repiten durante los segundos dados en el parámetro de duración.
- move_relative(dx, dy, dz, dradians) Mueve el bebop un número relativo de metros en x (adelante / atrás, adelante es positivo), y (derecha / izquierda,

derecha es positiva), z (arriba / abajo, positivo es abajo) y “*dradians*” en radianes.

Sin embargo, ninguno de estos métodos sirve para realizar un control manual por medio del teclado o mediante un control automático, ya que para el primero se necesita del parámetro tiempo que no se sabe de antemano y el segundo depende de la distancia que tampoco se conoce. Así, se optó por crear un método propio, modificando el script de los métodos de pyparrot, el cual se encuentra en pyparrot/Bebop.py. El método implementado se define a continuación:

```
def fly(self, roll, pitch, yaw, vertical_movement):
        my_roll = self._ensure_fly_command_in_range(roll)
        my_pitch = self._ensure_fly_command_in_range(pitch)
        my_yaw = self._ensure_fly_command_in_range(yaw)
        my_vertical = self.ensure_fly_command_in_range(vertical_movement)
        command_tuple = self.command_parser.get_command_tuple("ardrone3", "Piloting", "PCMD")
      self.drone_connection.send_single_pcmd_command(command_tuple, my_roll, my_pitch, my_yaw,
my_vertical)
```

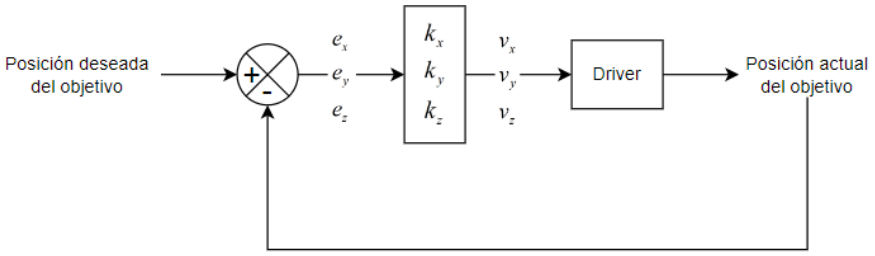
Este método permite que el VANT se desplace en cualquiera de sus variantes mientras la señal se envía, y se detiene cuando esta deja de enviarse.

Control en el seguimiento

El script para el control de la posición del Bebop se debe ejecutar en un hilo independiente y debe contemplar tanto el control manual por teclado como el control automático utilizando los fotogramas capturados (Figura 19). Se precisa que este proceso se ejecute en paralelo para evitar errores por la entrada en bucles que este script implica. También es importante señalar que así se procura una acción rápida ante emergencias con el fin de evitar colisiones.

Figura 19

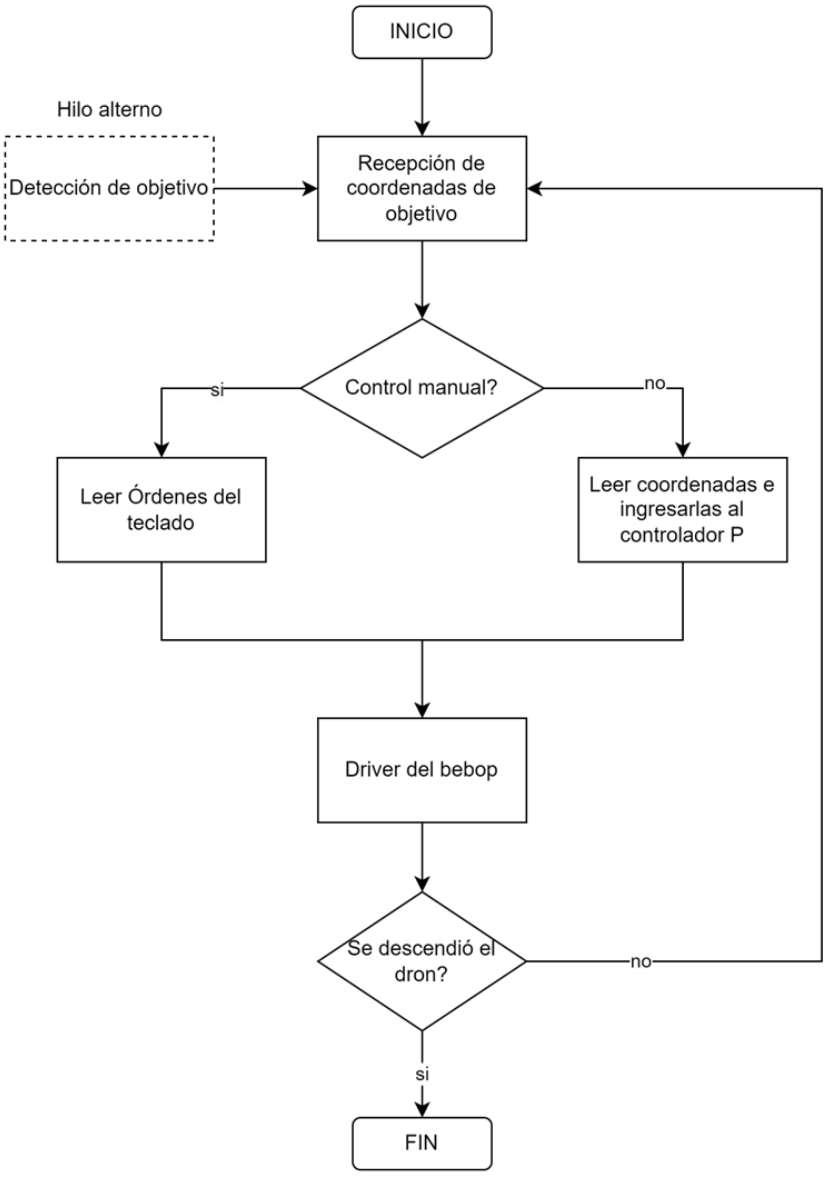
Diagrama de bloques del control de posición del VANT



El algoritmo para el control de posición del VANT se presenta en el siguiente diagrama de flujo:

Figura 20

Diagrama de flujo del algoritmo de control de posición del VANT



En la implementación del código en Python, la clase recibe los siguientes parámetros: El objeto Bebop creado con pyparrot, el setpoint que debe seguir el objeto detectado (se ubica por defecto en el centro de la cámara) y la velocidad máxima que se desea que se alcance cuando se ejecuta el control.

Predicción y seguimiento

Una vez obtenido el modelo entrenado, está próximo para realizar las predicciones de la escena capturada. Estos valores se obtienen utilizando la librería pytorch de Python, la cual ingresa el frame obtenido en el momento, y da como resultado un listado de todas las posibles predicciones en la escena. Bajo esta premisa, puede existir la casualidad que el dron obtenga resultados no esperados o falsas detecciones. Por esta razón solamente se toma el resultado con mayor porcentaje de predicción.

Figura 21

Array de salida durante la detección

DETECCIONES					
OBJETO	X	Y	WEIGHT	HEIGHT	ACCUARY
0	300	300	50	100	90

Interfaz Gráfica de Usuario

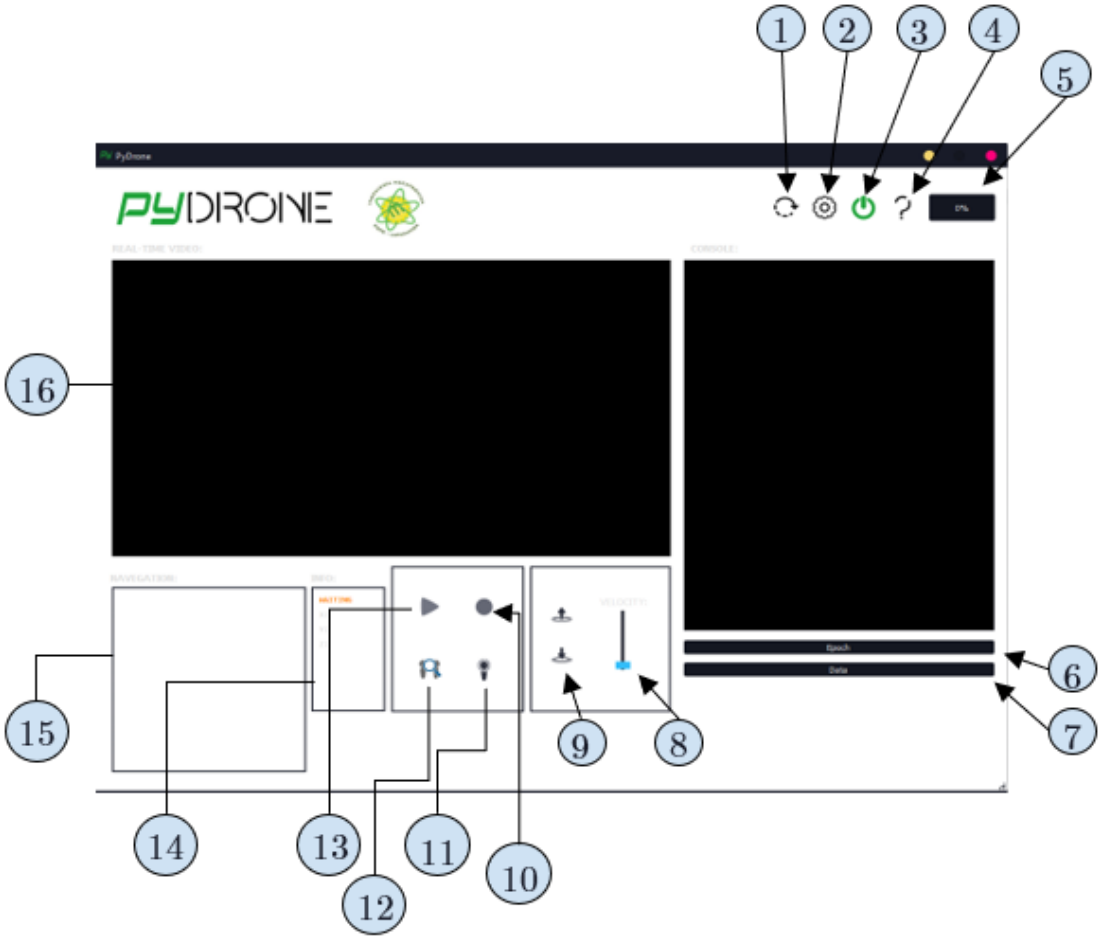
La GUI (Graphical User Interface) se desarrolló en Python, con ayuda de la biblioteca PyQt5 y cuenta con los siguientes elementos:

1. **Botón Cargar Modelo:** Carga un modelo previamente entrenado para ser usado en la detección mediante YOLO.

2. **Botón Configuraciones:** Define las configuraciones para la toma de datos para el entrenamiento y permite definir la ruta de donde se cargará el modelo entrenado.
3. **Botón On/Off:** Establece o termina la conexión del procesador con el VANT.
4. **Botón ayuda:** Presenta en la consola una ayuda para el manejo de comandos.
5. **Barra de estado de batería:** Muestra el porcentaje de carga de la batería
6. **Barra de progreso de aprendizaje:** Muestra el progreso de aprendizaje de la RNA.
7. **Barra de progreso de datos:** Muestra la cantidad de datos que se van almacenando.
8. **Deslizador de velocidad:** Controla la velocidad máxima del VANT entre un rango de 5 a 50.
9. **Botones Take Off y Land:** Despega y Aterriza el VANT, respectivamente.
10. **Botón Stop:** Finaliza la transmisión de fotogramas desde el VANT.
11. **Botón Tracking:** Inicia o detiene el seguimiento del objetivo.
12. **Botón New Profile:** Elimina los datos anteriores de entrenamiento e inicia uno nuevo.
13. **Botón Play:** Inicia la transmisión de fotogramas desde el VANT.
14. **Panel de información:** Muestra las coordenadas y el niel de precisión del objeto a seguir.
15. **Ventana de navegación:** Muestra una referencia de posicionamiento del objeto con respecto al centro de la escena.
16. **Videocámara:** Muestra el video en tiempo real desde la cámara del VANT.
17. **Consola:** Muestra la información de la consola y comandos que se envían y reciben del VANT.

Figura 22

Partes de la GUI del proyecto.



Capítulo IV.

Resultados y Análisis

Durante el desarrollo de todo el proyecto, se realizaron varias pruebas que permitieron corregir ciertos métodos que se pensaban como soluciones a priori. En primera instancia, se había considerado el funcionamiento de un entrenamiento sobrevuelo del sistema. Para descartar esta idea, se intentaron pruebas de detección de objetivos mediante una RNAC entrenada de manera local y con apenas 5 capas convolucionales. Esta red se entrenaba con un número fijo de 100 imágenes. Los datos de las pruebas realizadas con esta red se presentan en las tablas: Tabla 8, Tabla 9, Tabla 10 y Tabla 11, con sus respectivas gráficas.

Tabla 8

Datos de prueba con Persona 1: Jonathan

Épocas	Tiempo de entrenamiento	Tiempo de prueba	Tiempo de seguimiento
30	4 min.	10 min.	3,5 min.
40	6 min.	10 min.	4 min.
50	7,5 min.	10 min.	5 min.
60	9 min.	10 min.	3 min.

.

Figura 23

Datos de prueba con Persona 1: Jonathan

Tabla 9

Datos de prueba con Persona 2: Kevin

Épocas	Tiempo de entrenamiento	Tiempo de prueba	Tiempo de seguimiento
30	4 min.	10 min.	3 min.
40	7,3 min.	10 min.	4 min.
50	7,9 min.	10 min.	2,5 min.
60	8 min.	10 min.	3,5 min.

Figura 24

Datos de prueba con Persona 2: Kevin

Tabla 10

Datos de prueba con Objeto 1: Muñeco

Épocas	Tiempo de entrenamiento	Tiempo de prueba	Tiempo de seguimiento
30	5 min.	10 min.	3 min.
40	5,5 min.	10 min.	4 min.
50	6 min.	10 min.	1 min.
60	9 min.	10 min.	6 min.

Figura 25

Datos de prueba con Objeto 1: Muñeco

Tabla 11

Datos de prueba con Objeto 2: Planta

Épocas	Tiempo de entrenamiento	Tiempo de prueba	Tiempo de seguimiento
30	4,5 min.	10 min.	4 min.
40	5 min.	10 min.	4 min.
50	5,5 min.	10 min.	2 min.
60	7,5 min.	10 min.	5 min.

Figura 26

Datos de prueba con Objeto 2: Planta

La Tabla 12 muestra un resumen de todas las pruebas a 60 épocas

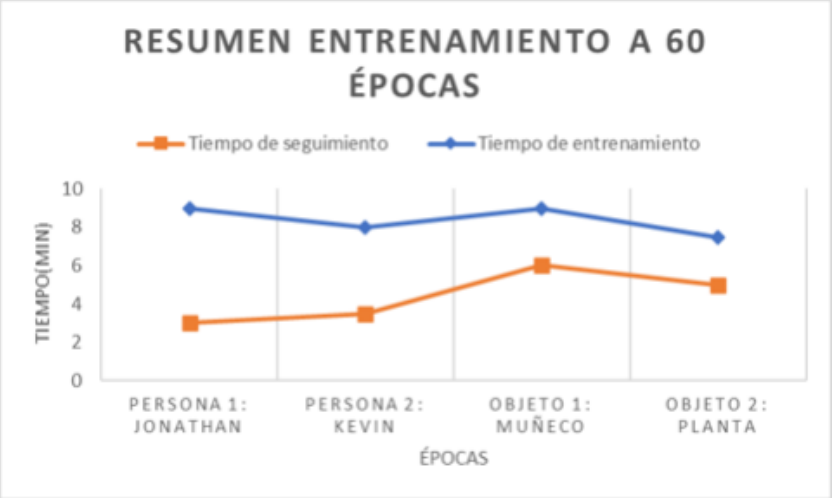
Tabla 12

Mejor seguimiento en cada caso probado

	Épocas	Tiempo de entrenamiento	Tiempo de prueba	Tiempo de seguimiento
Persona 1: Jonathan	60	9 min.	10 min.	3 min.
Persona 2: Kevin	60	8 min.	10 min.	3,5 min.
Objeto 1: Muñeco	60	9 min.	10 min.	6 min.
Objeto 2: Planta	60	7.5 min	10 min.	5 min.

Figura 27

Mejor seguimiento en cada caso probado



En estas pruebas se obtuvo que los tiempos de seguimiento eran en promedio, menos de la mitad del tiempo en que se ponía a prueba el sistema, lo que quiere decir que no supone un seguimiento óptimo de objetivos. Para este caso se priorizó un tiempo de entrenamiento corto, de alrededor de 3 minutos, con el objetivo de que el entrenamiento se realice mientras el VANT seguía en vuelo, no obstante, se sacrificó por mucho la precisión de la RNAC, llevando a un deficiente funcionamiento del sistema.

Uno de los puntos principales de este proyecto fue el desarrollo de un algoritmo que siga a una referencia en la imagen sin entrenamiento previo. Este algoritmo permite el seguimiento de objetivos de manera inicial para recopilar información acerca del sujeto. Se realizaron varias pruebas para corroborar su funcionamiento (

Tabla 13).

Tabla 13

Pruebas del algoritmo de reconocimiento por características ponderadas

	Tiempo de prueba	Tiempo de seguimiento
Persona 1: Jonathan	5 min.	3 min.
Persona 2: Kevin	5 min.	4.5 min.
Objeto 1: Muñeco	5 min.	4 min.
Objeto 2: Planta	5 min.	3.5 min.

Según las pruebas realizadas, este algoritmo funciona correctamente para breves intervalos de tiempo, alrededor de 2 minutos. Y siempre que el objetivo no se mueva con mucha velocidad relativa. Por estas características es que esta herramienta se la utiliza para realizar capturas al objetivo señalado en pantalla por cortos periodos de tiempo.

Luego, una vez unidas todas las etapas, se realizó una prueba de entrenamiento de todo el sistema, para contrastar los tiempos de ejecución entre el entrenamiento remoto y el local. Los datos se tomaron entrenando 4 casos diferentes, cada caso con una variación de épocas entrenadas específicas. Se presentan comparativas entre el tiempo de entrenamiento local y remoto, sin embargo, se toma un tiempo de seguimiento único para ambos casos, obtenido del promedio del tiempo de seguimiento local y en remoto.

Tabla 14

Datos entrenamiento objeto 1: Escudo de mecatrónica

Épocas	Tiempo de entrenamiento local	Tiempo de entrenamiento remoto	Tiempo de prueba	Tiempo de seguimiento
300	350 min.	15 min.	10 min.	7 min
500	560 min.	30 min.	10 min.	9.5 min.
700	720 min.	45 min.	10 min.	10 min.

Figura 28

Datos entrenamiento objeto 1: Escudo de mecatrónica

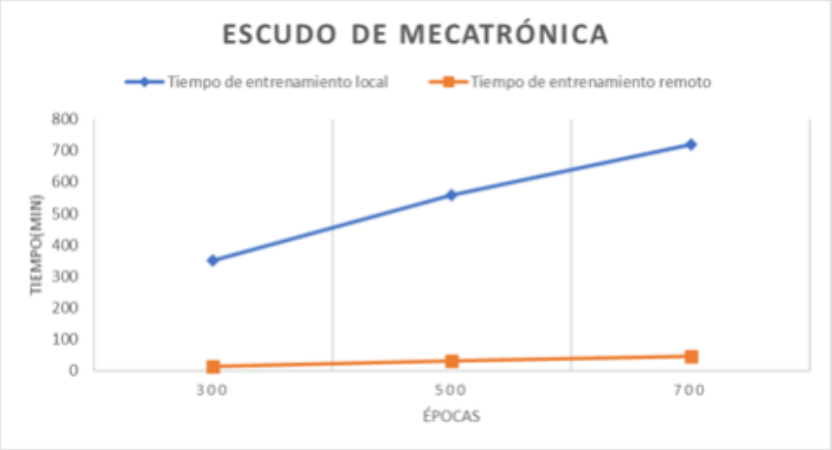


Tabla 15

Datos entrenamiento persona 1: Josué

Épocas	Tiempo de entrenamiento local	Tiempo de entrenamiento remoto	Tiempo de prueba	Tiempo de seguimiento
300	400 min.	17 min.	10 min.	5 min
500	550 min.	25 min.	10 min.	7 min
700	800 min.	40 min.	10 min.	10 min.

Figura 29

Datos entrenamiento persona 1: Josué

Tabla 16

Datos entrenamiento persona 2: Jonathan

Épocas	Tiempo de entrenamiento local	Tiempo de entrenamiento remoto	Tiempo de prueba	Tiempo de seguimiento
300	370 min.	10 min.	10 min.	5.5 min.
500	540 min.	20 min.	10 min.	8 min.
700	750 min.	40 min.	10 min.	10 min.

Figura 30

Datos entrenamiento persona 2: Jonathan

Tabla 17

Datos entrenamiento persona 3: Kevin

Épocas	Tiempo de entrenamiento local	Tiempo de entrenamiento remoto	Tiempo de prueba	Tiempo de seguimiento
300	320 min.	18 min.	10 min.	4 min.
500	580 min.	36 min.	10 min.	8.5 min.
700	730 min.	43 min.	10 min.	10 min.

Figura 31

Datos entrenamiento persona 3: Kevin

Como se observa en los datos obtenidos, se tiene un seguimiento aceptable a partir de las 500 épocas. Se recomendaría un entrenamiento de 1000 para evitar casi por completo los falsos positivos.

También se puede observar una amplia diferencia entre los tiempos de entrenamiento local y remoto. Mostrándose al entrenamiento local como no práctico para el proyecto, ya que se necesitaría horas de entrenamiento para realizar un entrenamiento confiable.

Cabe señalar que todas estas pruebas fueron realizadas bajo la condición de que el objeto no se realizaba movimientos rápidos, los sujetos de entrenamiento se movían a una velocidad promedio de caminata y se contaba con ausencia de vientos fuertes en el ambiente.

Capítulo V.

Validación

Evaluación de Hipótesis

Debido a que se posee datos de una sola variable categórica, es recomendable utilizar la herramienta de comprobación de ***ji cuadrado de bondad de ajuste.*** Según (Díaz Narváez, 2006) detalla que esta prueba, se utiliza para comparar uno o varios valores teóricos con respecto a valores experimentales, por otro lado, según las especificaciones del autor se define a H0, como la hipótesis en un caso positivo y nuestra H1, como la hipótesis a rechazar. A continuación, se indicará las posturas que se tomaron para concluir el proyecto.

Hipótesis nula (H_0)

Un sistema de aprendizaje sobrevuelo en combinación con un dron ayudará a realizar un seguimiento de objetos de forma óptima a manera de mejorar la seguridad en ambientes externos.

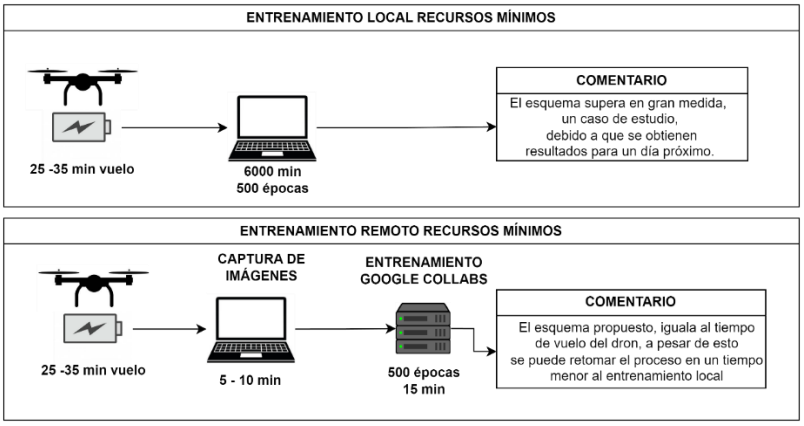
Hipótesis Alternativa (H_1)

Un sistema de aprendizaje sobrevuelo en combinación con un dron no ayudará a realizar un seguimiento de objetos de forma óptima a manera de mejorar la seguridad en ambientes externos.

Comprobación

Figura 32

Diferencias de funcionamiento entre entrenamiento local y remoto



A lo largo del desarrollo del presente proyecto y con sustento en investigaciones, trabajos anteriores y experimentos, se han replanteado ciertos procedimientos para que las soluciones fueran viables y cumpla la función de un seguimiento sobrevuelo mediante inteligencia artificial. Una de las conclusiones más importantes a las que se llegó, es que un aprendizaje sobrevuelo para los equipos considerados en este trabajo, no es viable. Esto se debe a que el tiempo de entrenamiento para una detección aceptable, supera por más de 9 horas al tiempo de vuelo del Parrot Bebop 2.

Con base en este impedimento, se concluye que no se puede comprobar ninguna de las hipótesis planteadas. No obstante, es congruente y de interés el análisis de la siguiente hipótesis:

¿Un sistema de reconocimiento de objetos usando IA en combinación con un VANT, ayudará a realizar un seguimiento de objetos de forma óptima con el objetivo de mejorar la seguridad en ambientes externos para la Universidad de las Fuerzas Armadas ESPE sede Latacunga?

Para esta hipótesis reestructurada se desglosan las variables respectivas:

- **Variable independiente:** Sistema de reconocimiento de objetos usando IA.
- **Variable dependiente:** Seguimiento de objetos de manera óptima con el objetivo de mejorar la seguridad en ambientes externos para la Universidad de las Fuerzas Armadas ESPE sede Latacunga

Así mismo, se plantean las hipótesis a validar:

- **H_o:** Un sistema de reconocimiento de objetos usando IA en combinación con un VANT, ayudará a realizar un seguimiento de objetos de forma óptima.
- **H_1:** Un sistema de reconocimiento de objetos usando IA en combinación con un VANT, no ayudará a realizar un seguimiento de objetos de forma óptima.

Para evaluar si el sistema de reconocimiento ayudará a realizar un seguimiento óptimo, se tomarán en cuenta los tiempos de seguimiento obtenidos a 500 épocas en las tablas:

Tabla 14,

Tabla 15, Tabla 16 y

Tabla 17. Para un seguimiento óptimo de objetivos, se esperaría a que el tiempo de seguimiento sea igual al tiempo de prueba, es decir, que el VANT haya seguido exitosamente al objetivo durante toda la prueba. Dadas estas consideraciones, se puede plantear la Tabla 18, en la cual se tienen los tiempos de seguimiento y tiempos de prueba, como frecuencias observadas y frecuencias esperadas, respectivamente.

Tabla 18

Datos de tiempos de prueba vs tiempo de seguimiento

	Tiempo de prueba	Tiempo de seguimiento

Objeto 1: Escudo de mecatrónica	10 min.	9.5 min.
Persona 1: Josué	10 min.	7 min.
Persona 2: Jonathan	10 min.	8 min.
Persona 3: Kevin	10 min.	8.5 min.
Objeto 2: Planta	10 min.	6.5 min.

Luego, se procede a calcular el estadístico de ji cuadrado con la ecuación (2)

$$\chi^2 = \sum \frac{\left(o_i - e_i\right)^2}{e_i} \quad \textbf{(2)}$$

Tabla 19

Cálculos para determinar ji cuadrado

Objetivo	o	e	$\frac{\left(o_i - e_i\right)^2}{e_i}$
1	9.5	10	0.025
2	7	10	0.9
3	8	10	0.4
4	8.5	10	0.225
5	6.5	10	1.225

Por tanto, se tiene que χ^2 (ji cuadrado) es:

$$\chi^2 = 2.775 \quad \textbf{(3)}$$

Los grados de libertad (gl), para este caso corresponden al número de categorías menos 1, por tanto $gl = 5 - 1 = 4$. Luego, estableciendo un nivel de significancia de $\alpha = 0.05$,

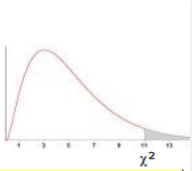
se ubica el valor crítico de ji cuadrado crítico (χ_c^2) en su tabla de distribución (Figura 33), resultando ji cuadrado como se muestra en la ecuación (4)

Figura 33

Valores críticos de la distribución JI cuadrado

Cátedra: Probabilidad y Estadística
Facultad Regional Mendoza
UTN

Tabla D.7: VALORES CRÍTICOS DE LA DISTRIBUCIÓN JI CUADRADA



g.d.l	0,001	0,005	0,01	0,02	0,025	0,03	0,04	0,05	0,10	0,15	0,20	0,25	0,30	0,35	0,40
1	10,828	7,879	6,635	5,412	5,024	4,709	4,218	3,841	2,706	2,072	1,642	1,323	1,074	0,873	0,708
2	13,816	10,597	9,210	7,824	7,378	7,013	6,438	5,991	4,605	3,794	3,219	2,773	2,408	2,100	1,833
3	16,266	12,838	11,345	9,837	9,348	8,947	8,311	7,815	6,251	5,317	4,642	4,108	3,665	3,283	2,946
4	18,467	14,860	13,277	11,668	11,143	10,712	10,026	9,488	7,779	6,745	5,989	5,385	4,878	4,438	4,045
5	20,515	16,750	15,086	13,388	12,833	12,375	11,644	11,070	9,236	8,115	7,289	6,626	6,064	5,573	5,132

Nota: Obtenido de (Universidad Atacama de Chile)

$$\chi_c^2 = 9.488 \quad \textbf{(4)}$$

Debido a que $\chi^2 < \chi_c^2$, se acepta la hipótesis nula y se puede afirmar que un sistema de reconocimiento de objetos usando IA en combinación con un VANT, ayuda a realizar un seguimiento de objetos de forma óptima.

Conclusiones

- El seguimiento se concluye como satisfactorio, bajo la consideración de un entrenamiento mayor a 500 épocas.
- La conexión del ordenador con el dron se realiza de manera satisfactoria.
- La conexión y envío de datos en tiempo real se realizó de forma satisfactoria.
- Se logró modificar el código de entrenamiento para que se realicen detecciones de un objetivo a la vez.
- Bajo las especificaciones dadas con respecto a las épocas mínimas, para realizar el entrenamiento, se determina que, bajo las pruebas realizadas, la predicción y el seguimiento son satisfactorios en un 80% - 95% del tiempo de prueba que se vaya a realizar.
- Una de las conclusiones más importantes a las que se llegó, es que un aprendizaje sobrevuelo para los equipos considerados en este trabajo, no es viable.
- Entre mayor cantidad de escenarios se contemple, se llamará a escenarios, a las capturas del objetivo en varios lugares y posiciones, se obtendrán mejores resultados en el seguimiento, cabe indicar que de igual manera se debe aumentar la cantidad de épocas a entrenar.
- Es necesario tener un set de información del objetivo elegido, esto se lo puede obtener utilizando el aplicativo, o utilizando programas especializados como labelImg, para obtener el set de datos.
- Un entrenamiento con un set de datos menor a 500 imágenes disminuye drásticamente el porcentaje de aciertos en la predicción.
- El entrenamiento de imágenes en formato GRAY no es contemplado por la herramienta YOLO, a pesar de que es posible realizar el entrenamiento, no se recomienda utilizarlo en tiempo real, debido a que las predicciones se vuelven imprecisas y poco confiables.

- En función a los resultados, el objeto que más tiempo detectó y realizó seguimiento fue el escudo de mecatrónica, esto debido a que es un objetivo definido, el cual no cambia con el paso del tiempo.
- Con base en la investigación y pruebas realizadas, la factibilidad de realizar un entrenamiento y seguimiento en tiempo real al mismo tiempo, puede conllevar a un alto consumo de cómputo y tiempo de vuelo, lo cual no es soportado por la carga de la batería del dron.
- El seguimiento bajo un entrenamiento previo es viable en ambientes que no superen las capacidades del dron.
- Al realizar entrenamiento con épocas menores a las 500 se obtiene mayor ruido en las predicciones, obteniendo mayor cantidad de falsos positivos en la escena.
- Según los resultados obtenidos, a pesar de que el modelo ha sido entrenado, pueden existir falsas detecciones, esto debido a que no se contemplan todos los escenarios. Para explicar de mejor manera esta conclusión, se expone como ejemplo, un sujeto A el cual se encuentra a una distancia considerable y ha sido entrenado previamente, y por alguna situación el sujeto se cambia de ropa, esta situación al no ser contemplada en la etapa de entrenamiento, no será prevista y se perderá el enfoque de seguimiento.
- Existen algunos posibles problemas con respecto al seguimiento, uno de los casos detectados, al cual no se puede dar solución, es el escenario en el cual un rayo de luz o brillo afecta directamente a la cámara del VANT, esto en el proceso de seguimiento, puede evitar que el dron siga al objetivo incluso perderlo de su zona de visualización, manteniéndose estable del aire, en el caso de que no exista intervención del usuario, el dron descenderá cuando la batería se encuentre cerca del 5% de la carga.
- En base a las características del dron, la distancia de vuelo máxima es de 2 km., en el cual el dron se desconecta del ordenador y estableciéndose como activo en el aire, lo cual recae en una limitación para el presente proyecto. En el caso de que no exista

intervención del usuario, el dron descenderá cuando se encuentre cerca del 5% de la carga.

- Pese a las limitaciones del hardware para realizar pruebas de entrenamiento sobrevuelo, se comprobó exitosamente la funcionalidad del seguimiento automático y su potencial en aplicaciones de seguridad, búsqueda, rastreo y cualquier otra en la que intervenga un proceso de reconocimiento y seguimiento autónomo.
- Debido a que el sistema trabaja en un ambiente no controlado, los fotogramas obtenidos del Bebop pueden estar sujetos a variaciones de luminosidad, por lo que es necesaria una etapa de preprocesamiento del fotograma, donde se controlen los niveles de brillo y contraste de manera automática.
- Métodos OpenSources como ORB para ubicar un objetivo mediante características fuertes presenta inconvenientes, debido a que demuestra una alta probabilidad de presentar falsos positivos, lo que dificulta el seguimiento de un objeto específico.
- Al utilizar ORB, como un método de seguimiento de objetivos, se obtuvieron mayor cantidad de errores en la detección inicial, por lo cual no se puede considerar como un método a utilizar en tiempo real.
- La respuesta dada utilizando el método template, ofrecido por la dependencia opencv, es acertada con respecto a la búsqueda dentro de una imagen; sin embargo, se presentaban problemas en los casos que había mucho movimiento en la escena o del objeto, mostrando un claro desplazamiento del recuadro objetivo inicialmente definido con el paso del tiempo.
- A pesar de que el sistema actualmente trabaje bajo la opción de entrenamiento remoto, se pueden realizar las configuraciones para trabajar todo en modo local, bajo criterio del usuario.
- El sistema se encuentra preparado para poder realizar reentrenamiento de modelo, esto ayuda en gran medida a contemplar todos los escenarios posibles, claramente, se debe

tomar en cuenta, la necesidad de recopilación de información para la formación de una red especializada.

- Con el driver pyparrot, existen inconvenientes con respecto a la recepción de paquetes dañados en la obtención de imagen, esto se resuelve disminuyendo los fps, el tamaño de la imagen y estableciendo una calidad baja, lo cual genera un envío de paquetes más pequeños y ligeros.
- En el transcurso de la investigación, se realizó un entrenamiento sobre vuelo con la ayuda de tensorflow, bajo la arquitectura de checkpoints para un reentrenamiento consecutivo, a pesar de que la detección era poco aceptable en sus primeras etapas de entrenamiento, el método utilizado para obtener secciones del frame era bastante pobre, dando como resultado falsos positivos, de igual manera, se esperaba que el entrenamiento se concluyera después de varias etapas de entrenamientos, pero la capacidad de la batería del dron es un factor decisivo, evitando la visualización de buenos resultados a la largo del tiempo, por lo cual se desistió de este método.

Recomendaciones

- Debido a que el proyecto tiene opción a entrenamiento remoto, se recomienda que el sistema se utilice con dos redes wifi o una red wifi y una ethernet, para mantener la conexión con el dron y mantenerse conectado a internet para realizar el entrenamiento.
- Se recomienda realizar pruebas con una máquina que supere en gran medida a los requerimientos mínimos dados en la tesis. Bajo este esquema será posible trabajar en paralelo todo el sistema, mientras se está manejando el dron se espera a que el entramiento finalice para próximamente realizar el seguimiento debido.
- Bajo entorno de desarrollo, se recomienda trabajar con entornos virtuales para evitar errores en la instalación de versiones.
- Realizar pruebas de entrenamiento y seguimiento con un dron de mayor capacidad en la batería.
- Evitar trabajar con una velocidad mayor a la establecida por defecto, lo cual genera una disminución drástica en el tiempo de vuelo.
- No utilizar el seguimiento sin un entrenamiento previo con épocas menores a 500, debido a que las predicciones se vuelven inestables lo que provoca mayores falsos positivos en la detección.
- Realizar un reentrenamiento de un modelo a estudiar contemplando todos los escenarios posibles.
- El sistema se puede aprovechar en gran medida, realizando un servicio que pueda conectar al dron a la red, donde un maestro envíe señales y el dron envíe datos por un medio de un protocolo de red, todo esto será posible implementando una interfaz de red al dron.
- Conectar el sistema a otros VANTs, conservando la misma lógica de recepción de fotogramas y envío de señales de movimiento.

- No se recomienda trabajar en áreas cerradas debido a que el dron no está diseñado para trabajar en espacios cerrados.
- No se recomienda aumentarle la velocidad al controlador de modo automático, al realizarlo de esta manera, se puede observar que el sistema se vuelve inestable.
- No realizar pruebas o seguimientos con el dron bajo condiciones de alta ventisca, esto puede provocar que se descargue la batería a una mayor velocidad y posiblemente el dron tienda a chocar con algún objeto.
- Evitar realizar pruebas en áreas con poca luminosidad para asegurar que la predicción sea satisfactoria.
- En la captura inicial, se recomienda que el objetivo no realice movimientos bruscos, para evitar perder el enfoque y reiniciar la detección del objetivo.

Bibliografía

Apple. (s.f.). *Apple*. Obtenido de Parrot Bebop 2 Drone: https://www.apple.com/sg/shop/product/HJW62PA/A/parrot-bebop-2-drone

Barrios Arce, J. I. (2020). *Redes Neuronales Convolucionales*. Obtenido de juanbarrios: https://www.juanbarrios.com/redes-neurales-convolucionales/

Daintith, J., & Wright, E. (2008). *Oxford Dictionary of Computing.* Oxford University Press. Obtenido de https://books.google.com.ec/books?vid=9780199234004&redir_esc=y

Díaz Narváez, V. P. (2006). Definición de Ji-Cuadrado. En V. P. Díaz Narváez, *Metodología de la investigación científica y bioestadística: para médicos, odontólogos y estudiantes de ciencias de la salud* (págs. 242-245). Santiago de Chile: RIL editores.

Domínguez Mínguez, T. (2021). *Visión artificial: aplicaciones prácticas con OpenCV-Python* (Primera ed.). España: Marcombo. Obtenido de https://books.google.com.ec/books?id=FE1OEAAAQBAJ&printsec=frontcover&hl=es#v=onepage&q&f=false

Embention. (2022). *Artículos del archivo Drones NMAND archivos - Embention*. Obtenido de Embention: https://www.embention.com/es/productos/drones-nmand/

ffmpeg. (2022). *Documentation*. Obtenido de ffmpeg: https://ffmpeg.org/documentation.html

G. Lowe, D. (5 de Enero de 2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 1. Obtenido de https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf

Keras. (Agosto de 2022). *About Keras*. Obtenido de Keras: https://keras.io/about/

Kumar, P. &. (Abril de 2015). Quad Band Signal Strength Monitoring System Using Quadcopter and Quad Phone. doi:10.13052/jge1904-4720.511

Lemon Urriago, J. P., & Vandenberg Torres, S. J. (2019). Trabajo de Grado. *Algoritmo Rápido de Seguimiento de Objetos*. Bogotá, Colombia: Pontificia Universidad Javeriana. Obtenido de

https://repository.javeriana.edu.co/bitstream/handle/10554/57493/TG_1827_Algoritmo%20rapido%20de%20seguimiento%20de_LemmonVandenberg.pdf?sequence=1

Matich, D. (Marzo de 2001). Informática Aplicada a la Ingeniería de Procesos – Orientación I. *Redes Neuronales: Conceptos Básicos y Aplicaciones*, 12. Obtenido de https://www.frro.utn.edu.ar/repositorio/catedras/quimica/5_anio/orientadora1/monograias/matich-redesneuronales.pdf

McGovern, A. (5 de Enero de 2018). *Welcome to pyparrot's documentation!* Obtenido de Read the Docs: https://pyparrot.readthedocs.io/en/latest/

Monajjemi, M., & Bamford, T. (2015). *bebop_autonomy - ROS Driver for Parrot Bebop Drone (quadrocopter) 1.0 & 2.0*. Obtenido de Read the docs: https://bebop-autonomy.readthedocs.io/en/latest/

Neto, P. (2008). *Ensayos sobre seguridad ciudadana.* Quito: FLACSO-Sede Ecuador : Municipio del Distrito Metropolitano de Quito.

Nieto Guerrero, E. &. (05 de Marzo de 2020). Desarrollo de un modelo matemático, cinemático y dinámico con la aplicación de software, para modificar el funcionamiento de un dron, para que este realice monitoreo automático. *RECIAMUC*, 332-343. doi:10.26820/recimundo/4.(1).esp.marzo.2020.332-343

OpenCV. (2022). *About - OpenCV*. Obtenido de OpenCV: https://opencv.org/about/

OpenCV. (s.f.). *OpenCV: Template Matching*. Obtenido de OpenCV Open Source Computer Vision: https://docs.opencv.org/4.x/d4/dc6/tutorial_py_template_matching.html

Puerto, K. (20 de Abril de 2016). *Parrot Bebop 2, análisis: autonomía y facilidad de vuelo ideales para entrar en el mundo de los drones*. Obtenido de Xataka: https://www.xataka.com/drones/parrot-bebop-2-analisis-autonomia-y-facilidad-de-uso-para-la-renovacion-de-este-divertido-companero-de-vuelo

QThread. (6 de junio de 2022). *QThread - Qt for python*. Obtenido de https://doc.qt.io/qtforpython/PySide6/QtCore/QThread.html

Riverbank Computing. (2020). *What is PyQt?* Recuperado el Mayo de 2022, de Riverbank Computing | Introduction: https://www.riverbankcomputing.com/software/pyqt/

Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. *IEEE*, 1-8. doi:10.1109/ICCV.2011.6126544

Sapag Chain, N., Sapag Chain, R., & Sapag Puelma, J. M. (2014). *Preparación y evaluación de proyectos* (Sexta ed.). México D.F., México: McGRAW-HILL. Obtenido de http://104.207.147.154:8080/handle/54000/1243

Szeliski, R. (2021). *Computer Vision: Algorithms and Applications* (Segunda ed.). Washington DC, USA: Springer. Obtenido de http://szeliski.org/Book/

Ultralytics. (2020). *YOLOv5 Documentation*. Obtenido de Ultralytics: https://docs.ultralytics.com/

Universidad Atacama de Chile. (s.f.). *Departamento de matemáticas de la Universidad Atacama de Chile.* Obtenido de Universidad Atacama de Chile: https://mat.uda.cl/hsalinas/cursos/2010/eyp2/Tabla%20Chi-Cuadrado.pdf