

# Reconocer y Monitorear Cultivos Ilícitos Mediante Drones Autónomos

*Valderrama J. Diaz J.*

## Resumen

Los problemas de implementar sistemas de navegación completamente autónomos que pueden ser utilizados en aplicaciones militares, de rescate o investigación, se considera muy difícil, específicamente en áreas boscosas. Uno de los objetivos del sistema es encontrar caminos a través de bosques y selvas que el dron pueda seguir y en donde pueda identificar plantas específicas. La principal herramienta para dar solución es emplear redes neuronales artificiales con entrenamiento continuo y profundo y, con el uso de sensores que faciliten la navegación en los recorridos. Otro objetivo, es minimizar el uso de capas de las redes neuronales que pueden ocasionar inconvenientes en la navegación e identificación de patrones.

**Palabras clave:** Webots, Redes Neuronales, Perceptrón Multicapa, GPS, Autónomo, Sensores, Identificación, Patrones, Imágenes, Control, Navegación, Entrenamiento.

## Abstract

The problems of implementing fully autonomous navigation systems that can be used in military, rescue or research applications, is considered very difficult, specifically in forested areas. One of the goals of the system is to find paths through forests and jungles that the drone can follow and where it can identify specific plants. The main tool to solve it is to use artificial neural networks with continuous and deep training and, with the use of sensors that facilitate navigation on the routes. Another objective is to minimize the use of neural network layers that can cause inconveniences in the navigation and identification of patterns.

**Keywords:** Webots, Neural Networks, Multilayer Perceptron, GPS, Autonomous, Sensors, Identification, Patterns, Images, Control, Navigation, Training.

## Introducción

Actualmente el uso de Drones para reconocimiento de imágenes, especialmente en la Industria Militar, ha ido en aumento y no solo se con aplicaciones de uso recreativo, de rescate en desastres, para labores de Delivery, en

eventos deportivos y demás. La mayor problemática de la utilización de drones es el alcance con respecto al posicionamiento, GPS (Sistema de Posicionamiento Global) y adicional a esto, el entrenamiento que debe tener un piloto para poder controlarlo

completamente sin generar un siniestro, teniendo en cuenta los altos costos de dichos dispositivos [3] (Hasegawa, A., Goto, T., & Hirano, S., 2018).

Existen varios reportes de navegación autónoma de Drones especialmente en pistas de Drones con obstáculos reconfigurables llevados a cabo en el programa STEM [5] (Lu, J., & Smith, B., 2017), donde estudiantes ponen a prueba sus conocimientos para superar los obstáculos en el menor tiempo posible. Los obstáculos pueden ser sorteados gracias a la capacidad de los sensores instalados en los dispositivos de vuelo, los que deben tener una alta eficiencia y alcance para evadir los obstáculos y notificar constantemente al controlador el estado actual.

En Colombia, específicamente en la Universidad Autónoma de Bucaramanga, se llevó a cabo un proyecto de Vehículo autónomo con el fin de ayudar a la oficina de correspondencia a realizar las entregas evitando así el desplazamiento de las personas dentro del Campus [1]

(Angarita Noriega, J. C., Roa Prada, S., & Moncada Guayazan, C. E., 2020). La desventaja o principal limitación se da por el alcance logrado que no supera los 1000m, sin embargo, esto puede ser corregido por antenas con mayor alcance o por sensores GPS, pero esto tiene el inconveniente del aumento en los costos.

El entrenamiento de estos drones en ambientes reales garantiza que éstos no se vean envueltos en situaciones de peligro mediante el uso de cámaras y algoritmos de adaptación al medio, dicho algoritmo ayudado de Redes Neuronales [7]

(Rivas Asanza, W., & Mazón Olivo, B., 2018) puede brindar la posibilidad de encontrar senderos en zonas boscosas para brindar a las personas un camino de acceso para realizar rescate o como es nuestro caso

acceder a zonas de cultivos ilícitos. La programación de estas redes neuronales exige un conocimiento previo de lenguajes de programación, pero son muy eficientes y capaces de aprender los recorridos realizados para detectar nuevos obstáculos si se llegan a presentar. Lo anterior también es posible complementarlo con el reconocimiento de patrones en las imágenes de los terrenos explorados y que, en nuestro país, son muy cambiantes, hasta similares, pero con características diferentes según el clima, altura y otros factores relevantes que deben ser tenidos en cuenta cuando se realice la programación de las redes neuronales.

Para garantizar una buena identificación de los senderos y caminos, al igual que los cultivos ilícitos es necesario utilizar reconocimiento de patrones mediante el uso de tablas de indexación difusa [8] lo cual nos brindaría una inferencia rápida y robusta, especialmente para el reconocimiento de los cultivos donde la forma de la hoja es el objeto principal de procesamiento al igual que se ha llevado a cabo para el seguimiento e identificación de humanos por medio de Drones de reconocimiento facial. Aquí se debe ser muy preciso con los patrones que deben ser reconocidos ya que la extensa y gran variedad de flora puede llegar a generar reconocimientos erróneos, esto puede ser corregido con una adecuada programación, reconocimiento no solo de patrones sino de diferencias específicas de colores y contar con cámaras de alta definición, que pueden ser de un costo razonable, gracias al desarrollo continuo en dispositivos encargados de imagen y video.

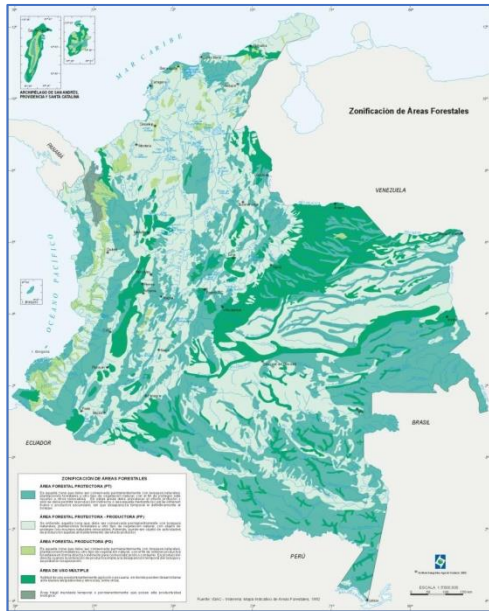


Figura 1. Mapa de zonas forestales Instituto Geográfico Agustín Codazzi.

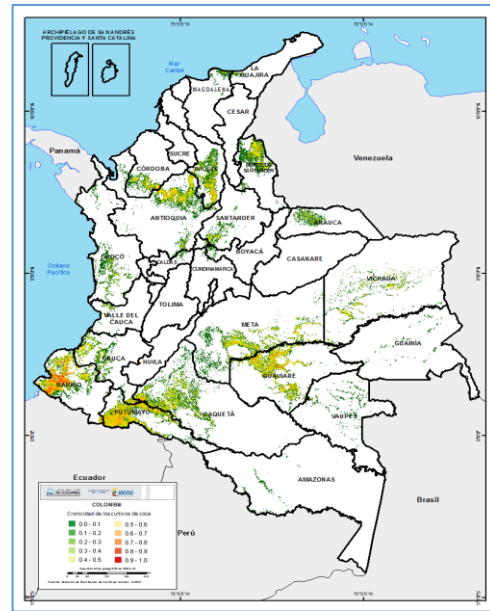


Figura 2. Mapa Cronicidad cultivos de coca Observatorio de drogas de Colombia.

El objetivo principal de nuestro artículo es utilizar todas las técnicas utilizadas que se describieron anteriormente con el fin de garantizar el monitoreo y reconocimiento de cultivos ilícitos en zonas de difícil acceso valiéndonos de Drones de conducción autónoma con un entrenamiento para enfrentar situaciones reales en zonas boscosas de la geografía colombiana, brindando así una ayuda tecnológica para garantizar la erradicación de dichos cultivos y el reemplazo por cultivos agrícolas.

## Planteamiento del problema

El difícil acceso a zonas en donde se encuentran concentrados los cultivos ilícitos ha sido uno de los principales obstáculos para un monitoreo y erradicación eficaz, así como el abandono estatal y diversos problemas sociales derivados de la proliferación de estos cultivos. La disposición geográfica en estas zonas facilita la ilegalidad.

El principal inconveniente que dificulta la implementación de proyectos de detección, monitoreo, control y revisión constante de cultivos ilícitos en zonas de difícil acceso, radica en la diversidad geográfica del país y la deficiencia en vías de acceso a zonas apartadas, sin contar con la problemática social que lleva a que el incremento de cultivos ilícitos sea cada vez mayor. Otro de los óbices, son las políticas no implementadas por el estado para brindar soluciones alternativas a la aspersión [12] (Gómez Jiménez, Andrés Felipe, 2020) y apoyo a la investigación que brinden soluciones tecnológicas mucho más eficientes.

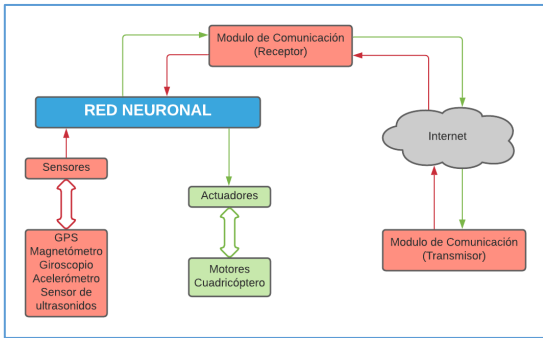


Figura 3. Flujo de procesos Dron, autoría propia.

Se debe considerar la gran dificultad de acceso a muchas zonas ricas en recursos naturales, pero en las cuales no hay presencia del estado y hay una gran fragmentación social, política, y económica desde el inicio de la historia de Colombia. Los estados ilegales [11] (Pardo, Daniel, 2020) son parte del problema principal en el que, basados en la desconfianza de las zonas apartadas hacia el gobierno central, los grupos al margen de la ley aprovechan todas las diferencias sociales para la siembra de cultivos ilícitos involucrando a las comunidades en actividades ilegales, pero a las que no hay más alternativas.

Comparando las figuras 1 y 2 se observa que las zonas más afectadas por cultivos ilícitos son zonas geográficamente accidentadas con grandes bosques y ríos que dificultan el acceso.

## Marco Teórico

### Modelado en Webots

El lenguaje utilizado en Webots es el de modelado de realidad virtual (VRML de sus siglas en inglés de Virtual Reality Modeling Language), es el lenguaje de programación utilizado para modelar el mundo virtual donde se simulará el comportamiento de los robots. Este lenguaje posibilita la descripción de

objetos 3D a partir de prototipos basados en formas geométricas básicas o estructuras descritas a partir de vértices y aristas como se explica en Cyberbotics. [23] (Cortes, Edmundo, Palomino, Amparo & Martínez, Cesar, 2019)

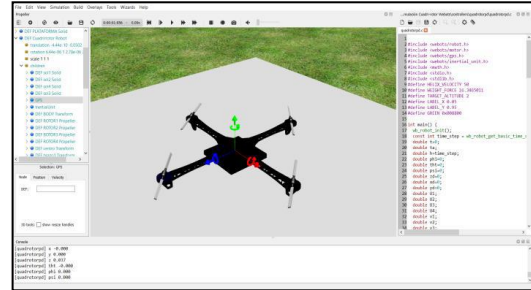


Figura 4. Interfaz Gráfica de Webots, autoría propia generada por software.

La figura 4 muestra la interfaz gráfica del software de simulación Webots que está conformada por:

1. El Árbol de escenas (izquierda): Muestra la información de describe un mundo simulado en forma jerárquica. El árbol de escenas está estructurado como un archivo VRML97, compuesto por una lista de nodos y cada nodo contiene campos.
2. Ventana 3D (centro): Permite interactuar con la simulación 3D. En esta ventana se puede, navegar en la escena, mover un objeto, aplicar fuerza a un objeto y aplicar torque a un objeto.
3. Editor de texto (derecha): Es un editor de texto de múltiples pestañas que permite desarrollar controladores. El editor presenta resaltado de sintaxis para el lenguaje compatible con Webots.

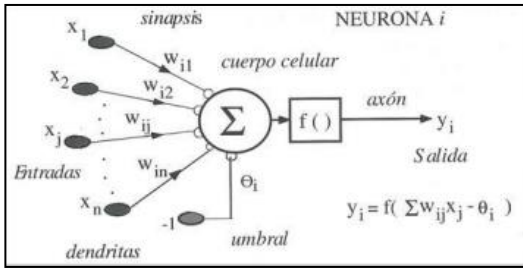


Figura 5. Modelo para neurona artificial o Perceptrón simple, tomado de Link9.

4. Consola (abajo): Muestra mensajes de compilación y salidas de ejecución.

### Redes Neuronales Artificiales

Una red neuronal artificial (ANN) es un esquema de computación distribuida inspirada en la estructura del sistema nervioso de los seres humanos. La arquitectura de una red neuronal es formada conectando múltiples procesadores elementales, siendo éste un sistema adaptivo que posee un algoritmo para ajustar sus pesos (parámetros libres) para alcanzar los requerimientos de desempeño del problema basado en muestras representativas.

El elemento fundamental de una red neuronal es la "neurona", también denominada nodo", cuyo modelo" puede verse en la Figura 4. En ella, la suma de las  $n$  entradas  $x_j$  de la neurona  $i$ , ponderadas con los pesos sinápticos  $w_{ij}$ , genera la entrada ponderada total o "potencial postsináptico" de la neurona  $i$ . Los pesos sinápticos  $w_{ij}$  miden la intensidad de la interacción entre las dos neuronas que están conectadas por el enlace. Posteriormente, se aplica una función de activación o transferencia ( $f$ ) a la diferencia entre el "potencial postsináptico" y el umbral  $\theta_i$ , obteniéndose la salida de la neurona ( $y_i$ ).

Donde  $y_i(t) = f(\sum_{j=1}^n w_{ij}x_j - \theta_i)$  es el modelo matemático más elemental. La función de activación  $f()$  se suele considerar determinista, y en la mayor parte de los modelos es monótona creciente y continua.

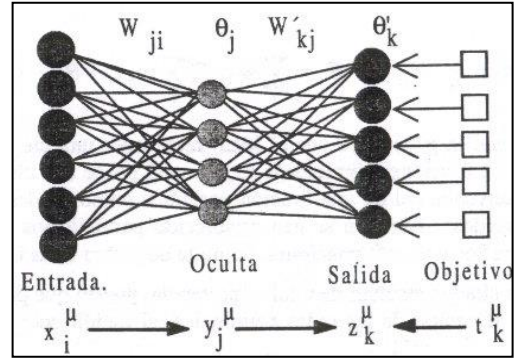


Figura 6. Representación de un Perceptrón Multicapa (MLP), tomado de Link11.

El Perceptrón Multicapa es uno de los tipos de redes más comunes. Se basa en otra red más simple llamada perceptrón simple solo que el número de capas ocultas puede ser mayor o igual que una. Es una red unidireccional (feedforward). La arquitectura típica de esta red es la siguiente:

Las neuronas de la capa oculta usan como regla de propagación la suma ponderada de las entradas con los pesos sinápticos  $w_{ij}$  y sobre esa suma ponderada se aplica una función de transferencia de tipo sigmoide, que es acotada en respuesta. El aprendizaje que se suele usar en este tipo de redes recibe el nombre de retro propagación del error (backpropagation). Como función de coste global, se usa el error cuadrático medio. Es decir, que dado un par  $(x_k, d_k)$  correspondiente a la entrada  $k$  de los datos de entrenamiento y salida deseada asociada se calcula la cantidad:

$$E(w_{ij}, \theta_j, w'_{kj}, \theta'_k) =$$

$$\frac{1}{2} \sum_p \sum_k \left[ d_k^p - f \left( \sum_j w'_{kj} y_j^p - \theta'_k \right) \right]^2$$

Vemos que es la suma de los errores parciales debido a cada patrón (índice  $p$ ), resultantes de la diferencia entre la salida deseada  $d_p$  y la salida que da la red  $f(\cdot)$  ante el vector de entrada  $x_k$ . Si estas salidas son muy diferentes de las salidas deseadas, el error cuadrático medio será grande.  $f$  es la función de activación de las neuronas de la capa de salida e y la salida que proporcionan las neuronas de la última capa oculta. Sobre esta función de coste global se aplica algún procedimiento de minimización. En el caso del MLP se hace mediante un descenso por gradiente. Las expresiones que resultan aplicando la regla de la cadena son las siguientes:

$$\delta w'_{kj} = -\epsilon \frac{\partial E}{\partial w'_{kj}}$$

$$\delta w'_{ji} = -\epsilon \frac{\partial E}{\partial w'_{ji}}$$

$$\delta w'_{kj} = \epsilon \sum_p \Delta_k^p y_j^p$$

$$\text{con } \Delta_k^p = [d_k^p - f(v_k^p)] \frac{\partial f(v_k^p)}{\partial v_k^p}$$

$$\delta w_{kj} = \epsilon \sum_p \Delta_k^p y_j^p$$

$$\text{con } \Delta_k^p = \left( \sum_k \Delta_k^p w'_{kj} \right) \frac{\partial f(v_k^p)}{\partial v_k^p}$$

Siendo  $y_k$  las salidas de la capa oculta. El aprendizaje por backpropagation queda como sigue:

1. Inicializar los pesos y los umbrales iniciales de cada neurona. Hay varias posibilidades de inicialización siendo las más comunes las que introducen valores aleatorios pequeños.
2. Para cada patrón del conjunto de los datos de entrenamiento
  - a. Obtener la respuesta de la red ante ese patrón. Esta parte se consigue propagando la entrada hacia adelante, ya que este tipo de red es feedforward. Las salidas de una capa sirven como entrada a las neuronas de la capa siguiente, procesándolas de acuerdo con la regla de propagación y la función de activación correspondientes
  - b. Calcular los errores asociados según la ecuación
  - c. Calcular los incrementos parciales (sumandos de los sumatorios). Estos incrementos dependen de los errores calculados.
3. Calcular el incremento total para todos los patrones, de los pesos y los umbrales según las expresiones en la ecuación
4. Actualizar pesos y umbrales
5. Calcular el error actual y volver al paso 2 si no es satisfactorio. [24] (Palacios, Francisco, 2003)

### Los sistemas inteligentes de transporte

El Departamento de Transporte de EE.UU. (DOT por sus siglas en inglés), presenta los ITS como todos aquellos sistemas orientados a la *mejora de la seguridad y movilidad en el transporte a través de la integración de tecnologías avanzadas de comunicación en los vehículos y la infraestructura de transporte*. La sociedad de ITS del IEEE (en inglés, Instituto de Ingenieros Eléctricos y Electrónicos) da una definición más amplia,

presentándolos como *aquellos que emplean tecnologías sinérgicas y conceptos de ingeniería de sistemas para desarrollar y mejorar el transporte de todo tipo*. Por su parte, la organización europea de ITS ERTICO los introduce como la *integración de tecnologías de comunicación e información con la infraestructura de transporte, los vehículos y los usuarios; con el objetivo de sacar un mayor provecho a las redes de transporte, mejorando la seguridad y reduciendo el impacto en el medio ambiente*.

En concreto, los ITS se perfilan como la combinación de tecnología punta para el desarrollo de sistemas avanzados que, al ser integrados en los vehículos e infraestructura, mejoran la sostenibilidad del transporte –i.e. aéreo, marítimo y terrestre–.

Aprovechando las ventajas de los ITS en la automatización de vehículos, sería posible por ejemplo emplear controladores automáticos que sean capaces de reducir la distancia de separación entre vehículos en las carreteras, aumentando la capacidad de estas últimas sin necesidad de modificar su configuración actual. Así mismo, se reduciría/eliminaría el factor humano en la conducción, responsable de cerca del 90% de los accidentes ocurridos en España, de acuerdo con cifras de la Dirección General de Tráfico (DGT) (Soria, 2001). En cuanto a la eficiencia energética, se podría regular la velocidad de los vehículos automáticamente, buscando siempre la mayor eficiencia con bajas emisiones (Shladover, 2009).

No obstante, no hace falta esperar la entera automatización del vehículo y la infraestructura para observar los beneficios que trae consigo la aplicación

de los ITS al transporte por carretera. Hoy en día ya prácticamente todos los fabricantes de vehículos ponen a disposición del usuario distintos sistemas inteligentes (ADAS) que mejoran y facilitan significativamente la labor de conducción. [15] (Jorge, Godoy, 2013)

## **Sensores**

En el desarrollo del proyecto la base principal radica en señales de video y datos de telemetría.

El diseño se ha basado en los datos de sensores disponibles para la navegación. Centrándonos en las características técnicas de los sensores, dispone de dos cámaras integradas:

- ✓ La principal, una cámara frontal en alta definición de 720p a 30 fps.
- ✓ La segunda cámara, menos potente, es QVGA de 60 fps y con una colocada de forma vertical con orientación descendente.

La segunda cámara, no solo brinda datos de imagen, sino que tiene el principal objetivo de medir la velocidad.

En cuanto a los sensores empleados se cuenta con herramientas para la estabilización automática.

- ✓ Para el control de altitud integra un sensor de ultrasonidos, este genera datos de telemetría en función de los cambios de altitud.
- ✓ Acelerómetro digital de 3 ejes con precisión de  $\pm 50\text{mg}$  para monitorizar los movimientos de posición.
- ✓ Giroscopio de 2 ejes y giróscopo piezoeléctrico de precisión de  $2000^\circ/\text{s}$  para los movimientos de cabeceo, giro y viraje.

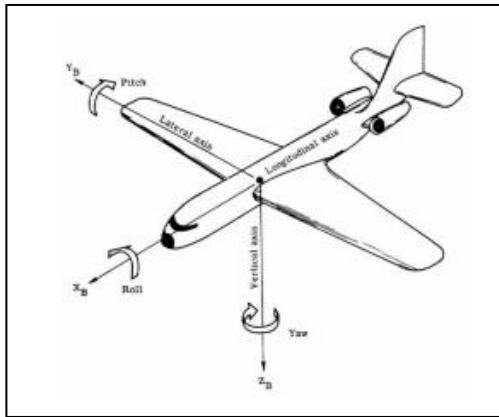


Figura 7. Cabeceo, giro y viraje tomado de Link12.

- ✓ Magnetómetro de 3 ejes con 6° de precisión con fines de orientación
- ✓ Modulo GPS NEO-6 con conexión para tarjetas Arduino, frecuencia de 5Hz.

El conjunto de estos sensores proporciona toda la información al software para la navegación. La fusión de los datos de los sensores proporciona los ángulos de Euler, utilizados para la estabilización. [17] (Fernández, Juan, 2013)

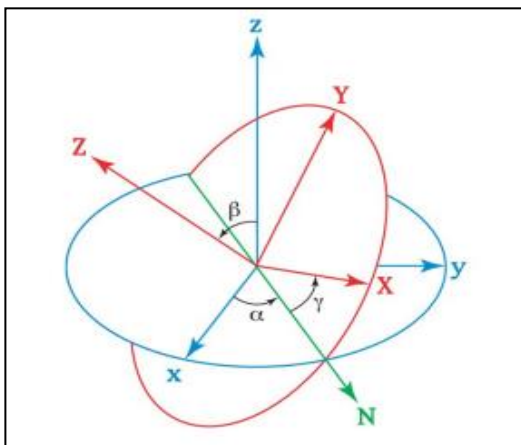


Figura 8. Ángulos de Euler, tomado de Link13

### Sensor de ultrasonidos

Hay diferentes modos de generar y recibir ondas ultrasónicas para aplicaciones de medición de distancias. Comúnmente se

pueden emplear ondas continuas o también ondas en forma de pulso. Los métodos de onda continua esencialmente son dos: los basados en la medición de la diferencia de fase entre la señal transmitida y la señal recibida de una onda modulada en amplitud, y los basados en la medición de la diferencia de frecuencia entre la señal transmitida y la señal recibida de una onda modulada en frecuencia [18] (Navarro, Danilo, Ríos, Luis Hernando, Parra, Hernando 2004).

En los sistemas basados en tiempo de vuelo, el sensor genera un pulso ultrasónico el cual se transmite a través del medio (típicamente aire) hasta que es reflejado por alguna superficie reflectora. Midiendo el tiempo entre la transmisión y la recepción del eco, la distancia al reflector puede ser estimada indirectamente mediante la formula  $d = \frac{1}{2} V \cdot t_f$ , donde  $V$  representa la velocidad del sonido en el medio de transmisión y  $t_f$  el tiempo de vuelo.

### Magnetómetro

Para este proyecto se plantea la utilización del módulo HMC5883L, un magnetómetro de 3-ejes que puedes usar como brújula o compás digital. Su interface de comunicación es de tipo I2C por lo que puede conectarse a una amplia gama de sistemas digitales. Posee un regulador de voltaje en placa, por lo que puede alimentarse directamente con 5V. Del mismo modo el puerto I2C es compatible tanto con 3.3V como con 5V, sin necesidad de usar un convertor adicional.

Su pequeño tamaño lo hace ideal para proyectos como Drones y sistemas de



navegación autónomos [19] (Naylamp Mechatronics).

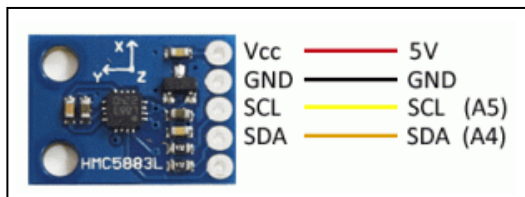


Figura 9. Conexiones magnetómetro, tomado de Link14.

## Giroscopio

Un giroscopio se basa en la variación de capacitancia entre el silicio y los elementos mecánicos, el sensor genera cambios capacitivos con cambios de velocidad angular.

Un giroscopio 3D tiene tres sensores giroscópicos montados ortogonalmente. Una medición de la fuerza-g se expresa en metros/segundo/segundo ( $m/s/s$ ), donde  $1g$  es igual a la fuerza gravitacional de la tierra. El mecanismo de detección de los tres giroscopios es también capacitivo.

Un giroscopio 3D mide la aceleración angular alrededor de los ejes X, Y y Z. Si la aceleración lineal se impone en un giroscopio, las distancias entre la placa fija interna y la masa permanecen invariables. Posteriormente, el giroscopio no responderá a una velocidad lineal.

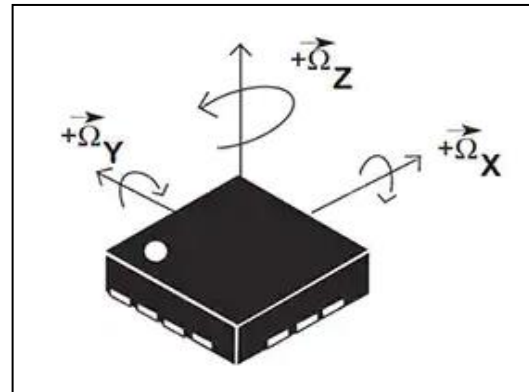


Figura 10. Giroscopio tridimensional, tomado de Link17.

## GPS (Sistema de Posición Global)

El sistema de posición global más conocido como GPS utiliza un Sistema Global de Navegación por Satélite (GNSS) que consiste en una constelación de satélites que transmite señales de radio utilizadas para el posicionamiento y localización en cualquier parte del globo terrestre, ya sea en tierra, mar o aire [22] (Pascual, Jorge, 2017). Estos dispositivos permiten determinar coordenadas geográficas desde un punto según la recepción de señales provenientes de la red de satélites artificiales.

El módulo NEO-6 pertenece a una familia de receptores fabricados por U-Blox, que pueden ser conectados con facilidad a un autómatas o procesador como Arduino.

La familia de receptores GPS NEO-6 están diseñados para tener un pequeño tamaño, pequeño coste, y pequeño consumo. La intensidad de corriente necesaria es de unos  $37mA$  en modo de medición continuo.

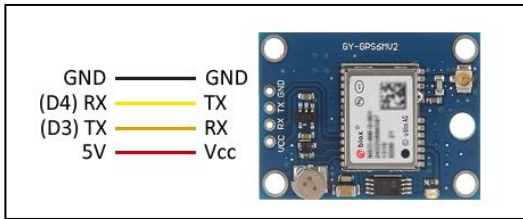


Figura 11. Conexiones GPS, tomado de Link15.

## Solución propuesta

El sistema propuesto tiene las siguientes novedades que son soportadas de acuerdo al marco teórico propuesto:

- ✓ Completa autonomía en el reconocimiento de los trayectos realizados teniendo la mínima intervención de operadores humanos y/o de sistemas informáticos que permitan el análisis de las imágenes recolectadas [8] (System of Autonomous Navigation of the Drone in Difficult Conditions of the Forest Trails, 2018)
- ✓ Completo diseño y simulación mediante el software Webots, para la posterior construcción e implementación del prototipo
- ✓ Autonomía (se busca que sea total) de operadores humanos en los trayectos complejos debido a zonas boscosas tomando decisiones de navegación, esto con la capacidad de aprendizaje constante en los recorridos realizados [4] (Kersandt, K., Muñoz, G., & Barrado, C., 2018)
- ✓ Comunicación y posicionamiento constante para la ubicación en tiempo real del vehículo, permitiendo monitorear el éxito del recorrido y de las tareas programadas. Las tareas programadas son de reconocimiento

del entorno y completar el trayecto planeado;

- ✓ Implementar una plataforma móvil, basada en inteligencia artificial, aprovechando el hardware del prototipo para fortalecer el software de acuerdo a los obstáculos identificados en cada caso. [7] (Rivas Asanza, W., & Mazón Olivo, B., 2018)



Figura 12. Muestras complejidad de los trayectos a) Tomado de Link16, b) tomado de Link7, c) Tomado de Link8.

Como se expuso, se busca que el sistema tenga total autonomía luego de haber sido entrenado en diversos recorridos siendo capaz de tomar decisiones propias, pero con reporte de estas en caso de tener que corregir.



Figura 13. Decisiones posibles según el recorrido, tomado de Link10.

## Metodología y Resultados

Este artículo fue elaborado como revisión bibliográfica basada en los artículos consultados y citados en la sección de referencias. Se realizaron comparativas entre modelos de drones, tecnologías utilizadas, métodos de rastreo y evasión de obstáculos, alcance de objetos fijados en los drones y estudio de los posibles terrenos en donde se puede desarrollar la actividad. Esta comparativa se realiza entre los mismos artículos consultados.

### Diseño ambiente simulado en Webots.

Se toma como base el ambiente de ejemplo basado en Dron Mavic 2 Pro de la compañía DJI, el cual posee como características principales:



Figura 14. Drone DJI Mavic 2 Pro, tomado de Link18.

- ✓ Cámara Hasselblad L1D-20c de 20 Megapíxeles, soporte de video en 4K. Función ActiveTrack 2.0 que proveen reconocimiento de precisión, predicción de trayectoria, seguimiento de alta velocidad y detección de obstáculos.
- ✓ Cuadrimotor rotacional para las cuatro propelas, además de tres motores rotacionales (cabeceo, giro y viraje) para el movimiento de la cámara.
- ✓ LED 's frontales de señalización.
- ✓ Sensor de Posición.
- ✓ Brújula.
- ✓ GPS.
- ✓ Giroscopio.
- ✓ Unidad Inercial

Se llevan a cabo las modificaciones sobre el entorno de tal forma que se observe un bosque y objetos que simulen un campamento de procesamiento de Coca rodeado de cultivos ilícitos (Figura 15).

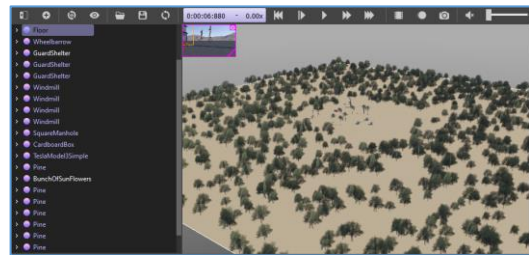


Figura 15. Entorno (world) diseñado para la simulación, autoría propia generada por software.

### Controlador

Se lleva a cabo el diseño de un Controlador en Java (Figura 13) que se integra al Software Webots y mediante el cual se conducirá el Dron en primera instancia con el fin de obtener ejemplos de Entrenamiento para ser ingresados

posteriormente al Perceptrón Multicapa desarrollado en Netbeans.

```

DronController.java
1 // File: DronController.java
2 // Date: 03/05/2021
3 // Description: Controlador diseñado para Dron - CUM
4 // Author: Oscar Rodríguez
5
6 import com.cyberbotics.webots.controller.Robot;
7 import com.cyberbotics.webots.controller.Camera;
8 import com.cyberbotics.webots.controller.Compass;
9 import com.cyberbotics.webots.controller.GPS;
10 import com.cyberbotics.webots.controller.Gyro;
11 import com.cyberbotics.webots.controller.InertialUnit;
12 import com.cyberbotics.webots.controller.Keyboard;
13 import com.cyberbotics.webots.controller.LED;
14 import com.cyberbotics.webots.controller.RotationalMotor;
15 import com.cyberbotics.webots.controller.Motor;
16 import com.cyberbotics.webots.controller.Accelerometer;
17
18
19 public class DronController extends Robot
20 {
21     private final int timeStep=32;
22
23     private Keyboard key; // Teclado
24     private InertialUnit InU[]; // Unidad Inercial
25     private Camera camara; // Camara

```

Figura 16. Controlador Dron diseñado en Webots, autoría propia generada por software.

Las salidas del Perceptrón Multicapa generan un archivo de texto que contiene los valores de perturbaciones (Cabeceo, Giro y Viraje) aplicadas al Dron para llevar a cabo la conducción y evasión de obstáculos. De esta forma se calcula la entrada de actuación de los motores como se observa en la Figura 17.

```

// Calcule las entradas del roll, pitch y yaw al igual que la altura vertical
final double roll_input = k_roll_p * Math.max(-1.0, Math.min(1.0, roll)) + roll_acceleration + roll_disturbance;
final double pitch_input = k_pitch_p * Math.max(-1.0, Math.min(1.0, pitch)) - pitch_acceleration + pitch_disturbance;
final double yaw_input = yaw_disturbance;
final double clamped_difference_altitude = Math.max(-1.0, Math.min(1.0, target_altitude - altitude + k_vertical_offset));
final double vertical_input = k_vertical_p * pow(clamped_difference_altitude, 3.0);

// Actuación de los motores teniendo en consideración todas las entradas calculadas.
final double front_left_motor_input = k_vertical_thrust + vertical_input - roll_input - pitch_input + yaw_input;
final double front_right_motor_input = k_vertical_thrust + vertical_input + roll_input - pitch_input - yaw_input;
final double rear_left_motor_input = k_vertical_thrust + vertical_input - roll_input + pitch_input - yaw_input;
final double rear_right_motor_input = k_vertical_thrust + vertical_input + roll_input + pitch_input + yaw_input;
final double pitch_camera_disturbance_1 = pitch_camera_disturbance;
final double yaw_camera_disturbance_1 = yaw_camera_disturbance;
motor[0].setVelocity(front_left_motor_input);
motor[1].setVelocity(front_right_motor_input);
motor[2].setVelocity(rear_left_motor_input);
motor[3].setVelocity(rear_right_motor_input);
camera_motor[0].setPosition(pitch_camera_disturbance_1);
camera_motor[1].setPosition(yaw_camera_disturbance_1);

```

Figura 17. Cálculos para la actuación de los Motores autoría propia generada por software.

Para la detección de los cultivos ilícitos se activa la función de Reconocimiento de la cámara del Dron mediante la cual se muestra en pantalla la información el objeto observado tales como nombre, posición y orientación relativa, tamaño de

la imagen a guardar, posición del objeto en la imagen y las componentes RGB del objeto como se evidencia en la Figura 18.

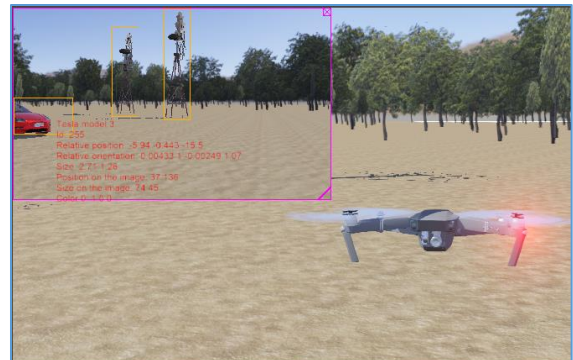


Figura 18. Reconocimiento de Imágenes Cámara integrada en el Dron, autoría propia generada por software.

Para la detección de los cultivos tomamos dos imágenes de prueba adquiridas por la cámara del Dron desde dos puntos de vista distintos de tal forma que se observen los cultivos al igual que árboles, terrenos de tierra y el horizonte.

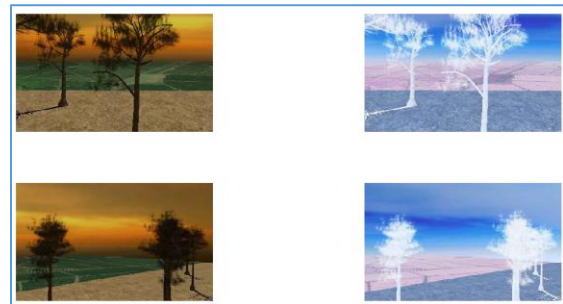


Figura 19. Procesamiento de Imagen con resultado, autoría propia generada por software.

En la Figura 19 encontramos el resultado de un procesamiento previo realizado mediante Octave donde empleamos la función:

$$imadjust(I, [low\_in; high\_in])(1)$$

Esta función permite realizar un ajuste de los valores de intensidad del mapa de colores sobre una imagen para resaltar los

tonos que deseemos, en nuestro caso los cultivos. En la Figura observamos a la izquierda dos imágenes de entrada y a la derecha las imágenes de resultado con la misma dimensión de la original después de aplicar el ajuste.

```

Hoja_Coca.m
1 pkg load image
2
3 I1=imread('Camara1.jpg');
4 I2=imread('Camara2.jpg');
5
6 R1=I1(:,:,1);
7 G1=I1(:,:,2);
8 B1=I1(:,:,3);
9
10 R2=I2(:,:,1);
11 G2=I2(:,:,2);
12 B2=I2(:,:,3);
13
14 Ajusta1=imadjust(I1,[1 0],[0 1]);
15 Ajusta2=imadjust(I2,[1 0],[0 1]);
16
17 RF1=Ajusta1(:,:,1);
18 GF1=Ajusta1(:,:,2);
19 BF1=Ajusta1(:,:,3);
20
21 RF2=Ajusta2(:,:,1);
22 GF2=Ajusta2(:,:,2);
23 BF2=Ajusta2(:,:,3);
24
25 subplot(2,2,1)
26 imshow(I1);
27 subplot(2,2,2)
28 imshow(Ajusta1);
29 subplot(2,2,3)
30 imshow(I2);
31 subplot(2,2,4)
32 imshow(Ajusta2);

```

Figura 20. Script de Ajuste de imagen y obtención de Ejemplos de entrenamiento, autoría propia generada por software.

Luego de tener la imagen Ajustada vamos a obtener los ejemplos de entrenamiento (Matrices RGB de imágenes de entrada y salida) para una Adalina la cual realizará el procesamiento de la imagen y resaltará los cultivos ilícitos para llevar a cabo la correcta detección de los cultivos.

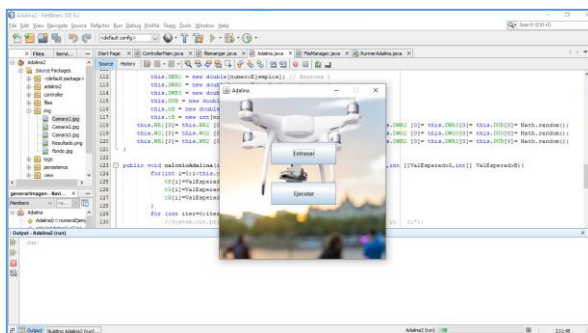


Figura 21. Adalina Procesamiento de Imagen, autoría propia generada por software.

La Adalina realizada en Java presenta una interfaz gráfica sencilla la cual posee dos botones:

- ✓ **Entrenar:** Llama el método calculo Adalina de la clase Adalina, el cual lee un archivo de texto con los ejemplos de Entrenamiento (seis entradas, valores RGB de imágenes Original y Resultado). Dicho entrenamiento retorna los pesos sinápticos para cada una de las entradas.
- ✓ **Ejecutar:** Al presionar dicho botón se llama el método generar Imagen de la clase Adalina que se encarga de leer la imagen generada por la cámara del Dron, y mediante los pesos sinápticos obtenidos en el entrenamiento se genera una imagen de Resultado la cual es guardada en una ubicación de archivo.

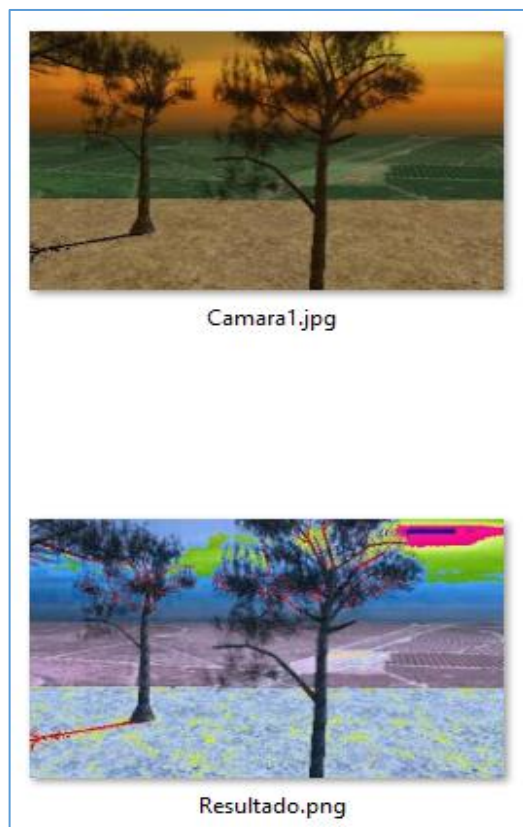


Figura 22. Resultado procesamiento mediante la Adalina, autoría propia generada por software.

Como podemos observar en la Figura 22 tenemos la imagen original (Camara1.jpg) y la imagen de salida (Resultado.png) donde se resalta en un color gris el cultivo ilícito y se diferencia de los demás objetos presentes en la imagen original.

## Conclusiones

En este artículo se ha buscado implementar un sistema de navegación autónomo con capacidad de reconocimiento de cultivos ilícitos empleando inteligencia artificial para memorizar los recorridos y con procesamiento de imágenes que permita la identificación de cultivos diferenciándolos de la demás flora encontrada en cada trayecto.

La dificultad radica, principalmente, en la construcción del prototipo capaz de recorrer grandes distancias de forma autónoma, aun con pruebas realizadas, debido a que el entorno es cambiante y el clima afecta en gran medida.

Otro de los retos es mejorar la capacidad de reconocimiento ya que las simulaciones deben ser puestas en práctica e ir comparando resultados para corregir posibles fallas.

Las simulaciones demuestran un buen desempeño en los recorridos y con capacidad de reconocimiento aceptable pero que deben ser probadas y comparadas en entorno real, lo cual implica contar con el prototipo y por supuesto el desplazamiento a zonas de riesgo para que con el conocimiento adquirido, se puedan corregir deficiencias en hardware y programación no prevista en el diseño y programación de

navegación autónoma e inteligencia artificial (neuronas digitales y su mejora a sistemas más complejos).

El rendimiento de este sistema se presenta a través de experimentos y pruebas en el software de simulación donde la tarea es reconocer cultivos ilícitos basados en la forma de la estructura.

Para futuras correcciones y mejoras el entrenamiento debe ser muy minucioso, ya que incluso el más mínimo cambio en la luz ambiental puede conducir a diferentes modelos (cultivos) que hacen que el procesamiento e identificación sea menos preciso.

## Referencias:

- [1] Angarita Noriega, J. C., Roa Prada, S., & Moncada Guayazan, C. E. (2020). Campus priority delivery system for the mail office of a university using an autonomous drone. IEEE IX International Congress of Mechatronics Engineering and Automation.
- [2] Devos, A., Ebeid, E., & Manoonpong, P. (2018). Development of Autonomous Drones for Adaptive Obstacle Avoidance in Real World Environments. IEEE 21st Euromicro Conference on Digital System Design.
- [3] Hasegawa, A., Goto, T., & Hirano, S. (2018). Accuracy Improvement of Image Recognition by Contrast Correction for Autonomous Drone Flights. IEEE 7 Global Conference on Consumer Electronics.
- [4] Kersandt, K., Muñoz, G., & Barrado, C. (2018). Self-training by Reinforcement Learning for Full-autonomous Drones of the Future. 2018 IEEE/AIAA 37th Digital Avionics Systems Conference.
- [5] Lu, J., & Smith, B. (2017). Autonomous Navigation in Drone Racecourse. IEEE MIT Undergraduate Research Technology Conference.
- [6] Pietrow, D., & Matuszewski, J. (2017). Objects detection and recognition system using artificial neural networks and drones artificial neural networks and drones. IEEE 2017 Signal Processing Symposium.

- [7] Rivas Asanza, W., & Mazón Olivo, B. (2018). Redes neuronales artificiales aplicadas al reconocimiento de patrones. Universidad Técnica de Marchala.
- [8] System of Autonomous Navigation of the Drone in Difficult Conditions of the Forest Trails. (2018). IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering.
- [9] Tusor, B., Takac, O., Molnar, A., Gubo, S., & Várkonyi-Kóczy, A. (2020). Shape Recognition in Drone Images Using Simplified Fuzzy Indexing Tables. SAMI 2020 • IEEE 18th World Symposium on Applied Machine Intelligence and Informatics.
- [10] Zientara, P., Choi, J., Sampson, J., & Narayanan, V. (2018). Drones as Collaborative Sensors for Image Recognition. IEEE International Conference on Consumer Electronics.
- [11] Pardo, Daniel (2020). Por qué la exuberante geografía de Colombia es a la vez una de sus mayores riquezas y de sus peores desdichas. [Link2](#)
- [12] Gómez Jiménez, Andrés Felipe (2020). La mata primero que el campesino: falla en las políticas de erradicación y sustitución de cultivos. [Link3](#)
- [13] Salas, Rodrigo (2004). Redes Neuronales Artificiales. Departamento de Computación. Universidad de Valparaíso.
- [14] Nacelle, Andrés (2009). Redes Neuronales Artificiales. Facultades de Ingeniería y Medicina. Universidad de la República.
- [15] Godoy, Jorge (2013). Arquitectura de Control para la conducción autónoma de vehículos en entornos urbanos y autovías. Escuela Técnica Superior de Ingenieros Industriales.
- [16] Acosta Burrelle, Luis Alberto (1997). Procesamiento paralelo con arquitecturas multi-dsp. Facultad de Informática. Universidad Nacional de La Plata.
- [17] Fernández, Juan (2013). Diseño De Un Sistema De Control Para Un Cuadricóptero. Departamento de Ingeniería de Sistemas y Automática Universidad Carlos III de Madrid
- [18] Navarro, Danilo, Ríos, Luis Hernando, Parra, Hernando (2004). Sensores De Ultrasonido Usados En Robótica Móvil Para La Medición De Distancias. Universidad Tecnológica de Pereira Pereira, Colombia
- [19] Naylamp Mechatronics. Módulo Magnetómetro 3-Ejes Hmc5883L. [Link4](#)
- [20] Baker, Bonnie (2018). Aplicar la fusión de sensores a acelerómetros y giroscopios. [Link5](#)
- [21] Mandujano, S, Mulero Pazmany, MC and Rísquez-Valdepeña, A (2017). Drones: una nueva tecnología para el estudio y monitoreo de fauna y hábitats. LJM Research Online.
- [22] Pascual, Jorge (2017). Adquisición Y Procesado De Información De Posicionamiento GPS Mediante Dispositivo Inalámbrico Basado En Arduino. Universidad Politécnica de Valencia.
- [23] Cortes, Edmundo, Palomino, Amparo & Martínez, Cesar (2019). Simulación de un Cuadróptero en el Software Webots. Memorias del Congreso Nacional de Control Automático.
- [24] Palacios, Francisco. (2003). Redes Neuronales con GNU/Linux. iBiblio. [Link6](#)
- [25] Bosque Andino en el Santuario de Fauna y Flora de Iguaque, departamento de Boyacá, Colombia [Link7](#)
- [26] Colprensa (2018), Cultivos ilícitos crecieron un 17% en Colombia: UNODC. [Link8](#)
- [27] Herramientas en GNU/Linux para estudiantes universitarios, la neurona artificial [Link9](#)
- [28] Ríos, Luis, Oportunidades Empresariales en los Bosques Colombianos [Link10](#)
- [29] Herramientas en GNU/Linux para estudiantes universitarios, El perceptron multicapa (MLP) [Link11](#)
- [30] Los tres ejes de rotación principales de una aeronave [Link12](#)
- [31] Dos sistemas de coordenadas ortogonales en el que se muestran los ángulos de Euler, 2007 [Link13](#)
- [32] Pineda, Pepe, 2020, Magnetómetro [Link14](#)
- [33] Llamas, Luis, 2016, Localización GPS con Arduino y los módulos GPS NEO-6 [Link15](#)
- [34] 5 bosques en Colombia para conocer, recorrer y amar, 2021 [Link16](#)
- [35] Baker, Bonnie, 2018, Aplicar la fusión de sensores a acelerómetros y giroscopios [Link17](#)
- [36] Feist, Jonathan, 2022, DJI Mavic 2 review – Mavic 2 Pro and Mavic 2 Zoom [Link18](#)