UNIVERSIDAD TÉCNICA DEL NORTE

Facultad de Ingeniería en Ciencias Aplicadas

Carrera de Ingeniería en Sistemas Computacionales

DESARROLLO DE UNA APLICACIÓN PARA DETECCIÓN AUTOMÁTICA DE LÍNEAS DE CULTIVO Y MALEZAS MEDIANTE EL PROCESAMIENTO DE IMÁGENES AGRÍCOLAS EN TIEMPO REAL.

Trabajo de grado previo a la obtención del título de Ingeniero en Sistemas Computacionales

Autor:

Sr. Yerson Fabricio Lucero Torres

Director:

Ing. Marco Remigio Pusdá Chulde MSc.

Ibarra - Ecuador

2023

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN

A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presentetrabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	100412162-8		
APELLIDOS Y NOMBRES:	LUCERO TORRES YERSON FABRICIO		
DIRECCIÓN:	OTAVALO - BARRIO VERDE VALLE		
EMAIL:	yflucerot@utn.edu.ec		
TELÉFONO FIJO:		**TELÉFONO MÓVIL:**	0980974354

DATOS DE LA OBRA	
TÍTULO:	DESARROLLO DE UNA APLICACIÓN PARA DETECCIÓN AUTOMÁTICA DE LÍNEAS DE CULTIVO Y MALEZAS MEDIANTE EL PROCESAMIENTO DE IMÁGENES AGRÍCOLAS EN TIEMPO REAL.
AUTOR (ES):	LUCERO TORRES YERSON FABRICIO
FECHA: DD/MM/AAAA	03/07/2023
SOLO PARA TRABAJOS DE GRADO	
PROGRAMA:	[x] **PREGRADO** [ ] **POSGRADO**
TITULO POR EL QUE OPTA:	INGENIERO EN SISTEMAS COMPUTACIONALES
ASESOR /DIRECTOR:	Ing. MARCO REMIGIO PUSDÁ CHULDE MSc.

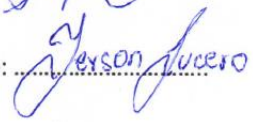
2. CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 3 días del mes de julio de 2023

EL AUTOR:

(Firma).....



Nombre: .....Yerson Jucero

CERTIFICACIÓN DEL DIRECTOR

Ing. Marco Remigio Pusdá Chulde MSc.

Director del presente trabajo de titulación certifica:

En mi calidad de tutor de trabajo de grado presentado por el Sr. Yerson Fabricio Lucero Torres para obtener el título de Ingeniería en Sistemas Computacionales cuyo tema es "DESARROLLO DE UNA APLICACIÓN PARA DETECCIÓN AUTOMÁTICA DE LÍNEAS DE CULTIVO Y MALEZAS MEDIANTE EL PROCESAMIENTO DE IMÁGENES AGRÍCOLAS EN TIEMPO REAL." Considero que el trabajo reúne los requisitos y méritos para ser sometido a la presentación pública y evaluación por parte del tribunal examinador.

Ibarra, a los 28 días del mes de junio de 2023

Ing. Marco Remigio Pusdá Chulde MSc.

DIRECTOR DE TRABAJO DE GRADO

DEDICATORIA

Este trabajo de grado está dedicado a Dios, a mis Padres Nancy y Jorge quienes son las personas más importantes de mi vida y por las cuales estoy culminando este objetivo, por ser mi modelo de vida, creer en mí y apoyarme en cualquier decisión que tome, por enseñarme a seguir adelante siendo siempre una buena persona, hacerme ver que con esfuerzo, honestidad, trabajo y responsabilidad todo lo propuesto se puede conseguir.

A mis hermanas, Dios me bendijo con 3 hermanas, Fernanda, Erika y Anahí las cuales son hermosas por dentro y fuera, cada una tan diferente, pero con el mismo buen corazón, por cuidarme y recordarme que somos un equipo, siempre estaremos para apoyarnos.

De manera especial, a mi incondicional acompañante Jessica Montenegro, a quien conocí en el primer día que inicié esta etapa y sigue estando a mi lado al finalizarla, siempre ensenándome a cumplir cada propósito, con dedicación y valentía.

AGRADECIMIENTOS

A Dios por guiarme en cada momento de mi vida a mis abuelos y padres por sus consejos, reprimendas y sacrificios, por su apoyo incondicional, por brindarnos a mis hermanas y a mí, las mejores oportunidades para luchar por nuestros sueños.

A mis demás familiares por siempre estar pendientes con sus buenos deseos y motivándome a seguir, a mis amigos por compartir alegría en cualquier momento y lugar, especialmente al grupo que conocí en la etapa universitaria “Los Tardones”, los cuales puedo llamar amigos para toda la vida, hemos vivido tantas anécdotas y espero seguirlas aumentando.

A mi director de tesis, Ing. Marco Pusdá por guiarme y apoyarme con sus conocimientos y recomendaciones, que me hicieron aprender y avanzar hasta la consecución del presente trabajo de titulación. Gracias por no solo ser un excelente director, sino un buen amigo.

A mis tutores PhD. Iván García Santillán y MSc. Cosme Ortega por el apoyo brindado, quienes supieron impartir de una manera óptima sus conocimientos sin egoísmo alguno.

A todos los profesores que me guiaron en mi formación personal y académica, desde la escuela hasta la universidad conocí a excelentes profesionales humanistas quienes supieron apoyarme en todos los niveles.

TABLA DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ABREVIATURAS

AP	Agricultura de Precisión.
ML	Machine Learning.
UAV	Vehículos Aéreos no tripulado
NDVI	Índice de Vegetación de Diferencia Normalizada (NDVI) es un simple indicador de la biomasa fotosintéticamente activa o, en términos simples, un cálculo de la salud de la vegetación.
GNDVI	Vegetación de Diferencia Normalizada Verde, es un índice del "verdor" de la planta o actividad fotosintética.
LiDAR	Tecnología de teledetección basada en láser, la idea detrás de LiDAR es apuntar un pequeño láser a una superficie y medir el tiempo que tarda el láser en volver a su fuente.
ROI	Regiones de Interés, es una parte de una imagen en la que desea filtrar u operar de alguna manera.
SVM	Support vector machine (SVM) es un algoritmo de aprendizaje supervisado que se utiliza en muchos problemas de clasificación y regresión, incluidas aplicaciones de procesamiento de señales, procesamiento del lenguaje natural y reconocimiento de imágenes y voz
CNN	Red Neuronal Convusional, es un tipo de Red Neuronal Artificial con aprendizaje supervisado que procesa sus capas con una jerarquía: esto quiere decir que las primeras capas pueden detectar líneas, curvas y se van especializando hasta llegar a capas más profundas que reconocen formas complejas como un rostro o la silueta de un animal.

RESUMEN

Con el avance tecnológico los planteamientos de ayuda a la agricultura dan paso a distintas estrategias de gestión y acción necesarias para transformar y orientar los distintos sistemas agrícolas, conformando así la AP, que es el campo que busca utilizar a las TIC en el apoyo de decisiones con el fin de mejorar el trabajo del agricultor y la calidad del producto, cuidando el medioambiente y siendo más eficiente económicamente, el presente proyecto comprende la revisión y aplicación de diferentes técnicas de visión por computador para el desarrollo de una aplicación para detección automática de líneas de cultivo y malezas mediante el procesamiento de imágenes agrícolas en tiempo real.

Se realizó una revisión bibliográfica para tener un marco teórico que conceptualizó las herramientas, técnicas y métodos de la AP que estableció la utilización del lenguaje Python con la librería OpenCv y el método Haar Cascade para el desarrollo y entrenamiento del algoritmo, este algoritmo se basa principalmente en la escala de las imágenes y los vecinos cercanos al detectar cualquier objeto, cuenta con el apoyo de Cascade Trainer GUI, que es un programa que se puede utilizar para entrenar, probar y mejorar modelos de clasificadores en cascada. El sistema utiliza el protocolo RTSP (protocolo de transmisión en tiempo real) ayudando a reducir la pérdida de datos en la obtención y transmisión de las imágenes, el algoritmo clasificador usa el marco de detección de objetos en 5 fases: Detección del área de interés, resta de fondo, segmentación, detección y conteo. El software alcanza una precisión de detección de líneas de cultivo del 92,10% y una exactitud en la detección de maleza del 80,49% promediando un valor arriba del 86,28% en la detección de malezas y líneas de cultivo, con un costo computacional de 0.92 siendo 1 lo más eficiente. Estos resultados fueron evaluados con base en mAP (Mean Average Precision) y los principios cuantitativos de diseño de computadoras.

Palabras Claves: Procesamiento de imágenes, Cultivos de cobertura, Detección de malezas, Detección de Objetos, Prácticas de precisión, aprendizaje automático, UAV, vehículo aéreo no tripulado.

ABSTRACT

With technological progress, approaches to support agriculture give way to different management and action strategies needed to transform and guide the different agricultural systems, thus forming AP, which is the field that seeks to use TICs in decision support to improve the farmer's work and the quality of the product while taking care of the environment and being more economically efficient, this project includes the review and application of different computer vision techniques for the development of an application for automatic detection of crop lines and weeds by processing agricultural images in real time.

A literature review was carried out to have a theoretical framework that conceptualized the tools, techniques and methods of the AP that established the use of the Python language with the OpenCv library and the Haar Cascade method for algorithm development and training, this algorithm relies mainly on the scale of images and nearby neighbors when detecting any object, is supported by the Cascade Trainer GUI, which is a program that can be used to train, test, and improve cascade sorter models. The system uses the RTSP (Real-Time Transmission Protocol) protocol helping to reduce data loss in obtaining and transmitting the images, the classifier algorithm uses the 5-phase object detection framework: Detection of the area of interest, subtraction of background, segmentation, detection and counting. The software achieves a crop line detection accuracy of 92.10% and a weed detection accuracy of 80.49% averaging a value above 86.28% in the detection of weeds and crop lines, with a computational cost of 0.92 being 1 the most efficient. These results were evaluated based on mAP (Mean Average Precision) and the quantitative principles of computer design.

Keywords: Image Processing, Cover Crops, Weed Detection, Object Detection, Precision Practices, Machine Learning, UAV, unmanned aerial vehicle.

INTRODUCCIÓN

1. Antecedentes

La importancia de la agricultura en el Ecuador está reflejada en el PIB (Producto interno Bruto), representando 8,4 mil millones de dólares al país, siendo los cultivos más producidos: arroz, banano, café, maíz duro, palma aceitera y papa (Yanez et al., 2018). La mayoría de las labores agrícolas se las realiza manualmente interviniendo pequeños productores, conformando una agricultura familiar, por lo que el Ecuador siendo un país en vías de desarrollo, tiene la necesidad de automatizar tareas que mejoren la producción y disminuyan la contaminación ambiental.

La informática ya genera herramientas y plataformas digitales para administrar y monitorear los procesos de cultivo, cuyas características dependen del entorno y región donde se encuentran. El campo de la automatización de los procesos agrícolas utilizando sistemas de visión por computador es el que más destaca debido a la precisión de resultados que puede llegar a tener (Danilo García Santillán, 2018). Las distintas herramientas van encaminadas al uso de sistemas automatizados que interactúen y tengan un tiempo de respuesta corto, eficaz y que resuelvan los problemas de los diferentes sectores que en su mayoría se repiten, llegando a tener variables establecidas como los tiempos de ejecución, mano de obra, contaminación y ahorro de recursos.

2. Situación actual

La inclusión de procesos tecnológicos como la visión por computador dispone de métodos y algoritmos para el análisis de imágenes captadas por un UAV (Vehículo aéreo no tripulado) o cámaras de video fijas, que sirven para extraer la información necesaria de las actividades y procesos agrícolas, la relevancia que hoy en día va teniendo la agricultura de precisión hace posible detectar líneas de cultivo y malezas con sus diferentes características,

algunas crecen 3 veces más gruesas y frondosas que otras, a veces no llegando a identificarse entre las malas hiervas propias del terreno y región, (Seijas et al., 2009) el análisis en tiempo real trata de optimizar el proceso enfocándose en términos de exactitud y velocidad de respuesta, eficiente para tratar con las condiciones adversas mencionadas.

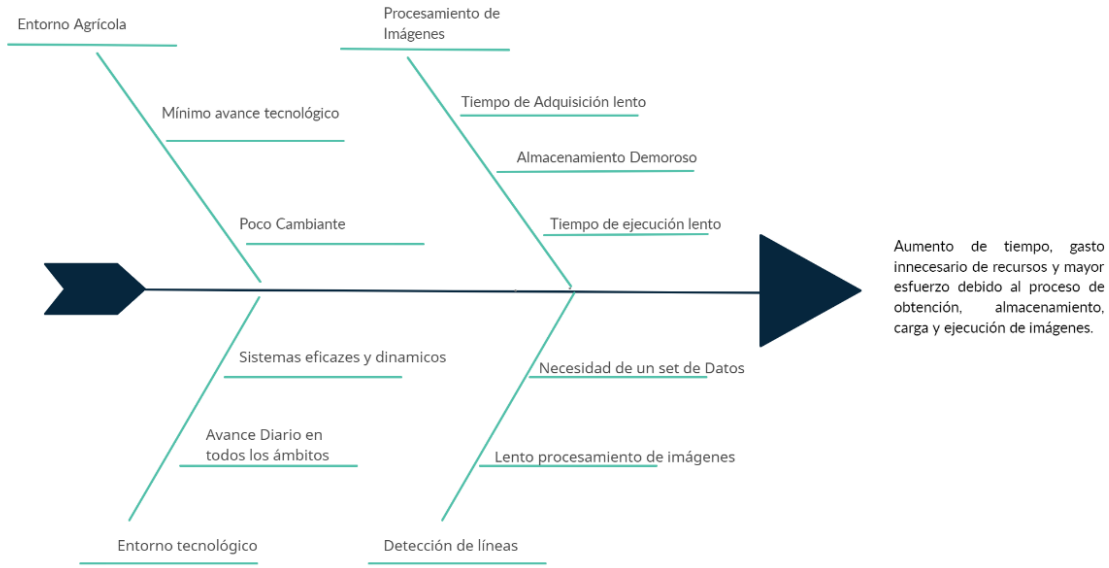
3. Planteamiento del problema

En nuestro medio cualquier actividad agrícola es realizada de forma manual y son contados los procesos agrícolas sistematizados a comparación de otras regiones del mundo, donde ya buscan una inclusión mayor de la tecnología en la agricultura, un campo de la visión artificial en el sector agrícola se enfoca en el procesamiento de imágenes de cualquier actividad con fotografías tomadas anteriormente, formando un set de datos que tiene que ser almacenado y tratado.

Esta tarea lleva tiempo y se complica aún más en grandes extensiones de terreno, ya que las fotografías deben tener una calidad suficiente, aumentando esfuerzo y tiempo en los procesos de carga y ejecución del sistema, la detección de malezas y líneas de cultivo es un área con mucho potencial y poco estudiada en el país, sus beneficios son varios, siendo un ejemplo la aplicación dedicada de fertilizantes. Como se menciona, a mayoría de las aplicaciones trabajan con fotografías ya obtenidas y previamente cargadas en el sistema, razón por la cual el presente trabajo desarrolla una aplicación que identifique en tiempo real las líneas de cultivo y malezas, del resto de malas hiervas con imágenes obtenidas en tiempo real, sin necesidad de cargar previamente un set de datos, la Figura 1 presenta el árbol de problemas del proyecto.

Figura 1

Árbol de problemas.



Nota. El Diagrama de Ishikawa es de elaboración propia con los problemas encontrados.

4. Objetivos

4.1. Objetivo general

Desarrollar una aplicación para detección automática de líneas de cultivo y malezas mediante el procesamiento de imágenes agrícolas en tiempo real.

4.2. Objetivos específicos

- Documentar la literatura existente sobre los lenguajes utilizados y aplicaciones realizadas para detección de líneas de cultivo y malezas en imágenes agrícolas.
- Desarrollar la aplicación para la detección automática de líneas de cultivo y malezas en tiempo real.
- Evaluar resultados empleando medidas cuantitativas para la comparación de detección de líneas de cultivo y malezas con diferentes fotografías a distancias establecidas.

5. Alcance

Se realizó una revisión de la literatura existente sobre los diferentes métodos y estrategias para la segmentación, detección y entrenamiento de imágenes en las bases de datos bibliográficas (Scopus, SJR, IEEE Xplore, ScienceDirect, Springer, Taylor & Francis, Scielo) a las que se encuentra subscrita la Universidad Técnica del Norte.

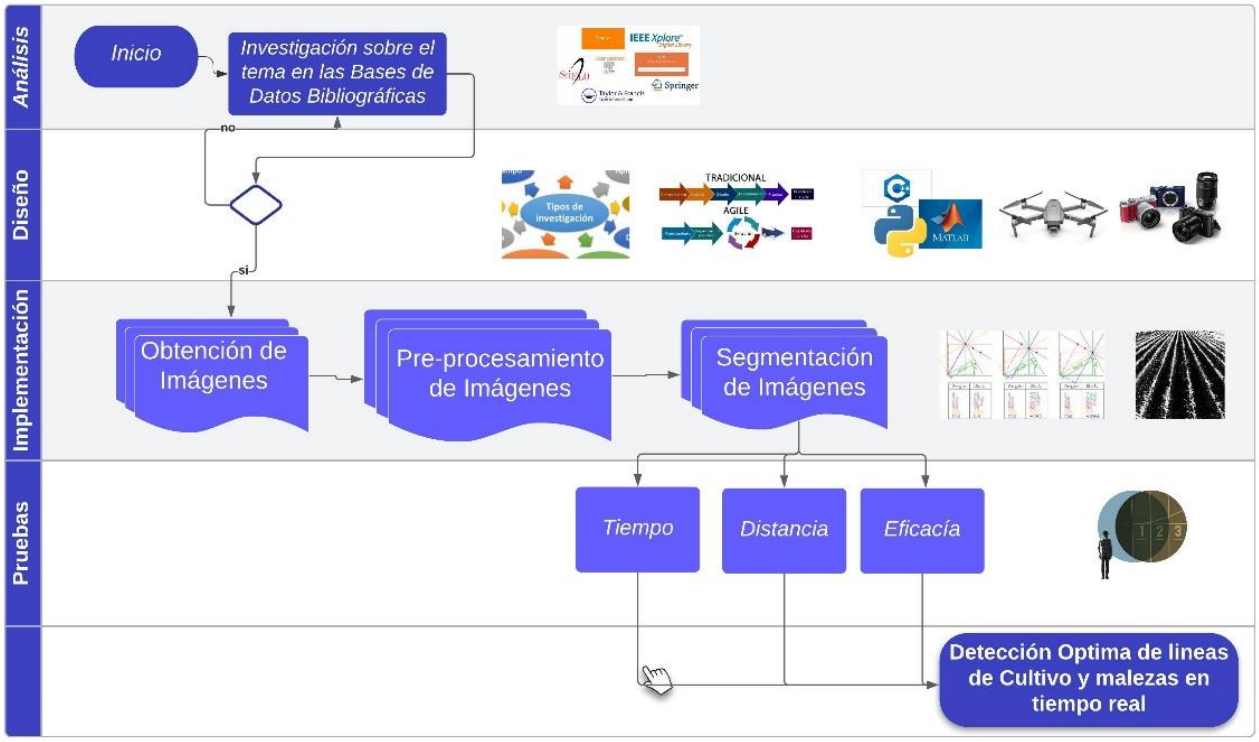
Se preparo la imagen, teniendo en cuenta la iluminación, ruido, altura, inclinación y resolución, se realizó la binarización de la imagen, se usó parámetros como el de vecinos más cercanos, estableciendo un umbral y condiciones de detección.

Para el desarrollo del software se utilizó una metodología tradicional “en Cascada”, se llevó a cabo un desarrollo secuencial, la aplicación es local, desarrollada en Python, con la librería OpenCv y el método de entrenamiento de clasificadores Haar Cascade.

Los resultados se evaluaron con métricas cuantitativas como la Mean Average Precisión (mAP), estableciendo distancias de la cámara al cultivo como 5,10 y 15 metros de altura, el cultivo de maíz tuvo un seguimiento de 4 semanas. La Figura 2 muestra un diagrama detallando los procesos que se realizaran en cada etapa de la metodología en cascada, también detalla las tecnologías a utilizar.

Figura 2

Alcance del proyecto



Nota. Alcance ordenado del proyecto, elaboración propia.

6. Justificación

El proyecto está enfocado en los ODS, el objetivo número 9 agua, Industria, innovación e infraestructura, destaca como el crecimiento del sector manufacturero a nivel mundial ha ido disminuyendo constantemente y como la innovación y el progreso tecnológico son claves para descubrir soluciones duraderas para los desafíos económicos y medioambientales (ONU, 2015). Incluso antes del brote de la pandemia de la COVID-19, la agricultura es un sector que está sufriendo alteraciones en el suministro de productos. Por lo cual proyectos innovadores de tecnología deben servir como una fuente de transformación del sector como la conocemos.

Justificación Tecnológica. - Hacer uso de nuevas tecnologías para nuestro medio que sistematicen las actividades diarias del sector, la detección en tiempo real optimizó el nivel de

respuesta, agilizo el procesamiento de imágenes, libro al sistema de una robustez, ayudo a una mejor gestión de cultivos en grandes extensiones de terreno reduciendo costes y sirvió para automatizar procesos de aplicación, como de fertilizantes y demás insumos de producción.

Justificación Teórica. - El enfoque de la presente investigación es profundizar en la teoría que trato el problema para avanzar en el conocimiento sobre la línea de investigación en el Ecuador.

Justificación Práctica. - La detección de líneas de cultivo y malezas mediante procesamiento de imágenes en tiempo real brindo una solución en conjunto de los estudios realizados por la academia en el campo de la agricultura, la cual es de vital importancia en nuestro país.

7. Contexto

Los avances para ayudar al campo de la agricultura surgen a diario, la mayoría de los proyectos e investigaciones sobre la detección de líneas de cultivo y malezas presentan métodos de discriminación con imágenes capturadas previamente y en tiempo real como se pretende realizar en este proyecto.

La Tabla 1 presenta las investigaciones más relevantes encontradas en el campo de la agricultura de precisión.

Tabla 1

Contexto de Investigaciones similares.

Tema	Diferencia	Aporte	Cita/Autor
Discriminación de cultivo y malezas en campos de papa usando técnicas de visión por computador	La detección en tiempo real no emplea imágenes previamente capturadas y almacenadas.	Se presenta un método adaptado para discriminación automática de cultivos y malas hierbas a través de imágenes capturadas en campos de papa.	García-Santillán, Pusdá, Caranqui, Landeta, Salazar, Granda (2019) Universidad Técnica del Norte
Detección automática de líneas de cultivo de papa utilizando imágenes digitales	Las imágenes serán obtenidas por una cámara ubicada en un punto estratégico que serán enviadas en tiempo real mediante una red Wifi para su análisis.	Utiliza imágenes que fueron obtenidas con una cámara instalada en el frente del tractor en proyección en perspectiva.	García, Herrera, Mina. (2016) Universidad Politécnica estatal del Carchi.
Sistema de Visión Artificial para el Análisis de Imágenes de Cultivo basado en Texturas Orientadas	La aplicación busca encontrar estructuras parametrizadas rectas/curvas en las imágenes y no texturas.	Detección de líneas basada en el análisis de texturas orientadas; tres estrategias: poca capa vegetal, rodales de mala hierba, variabilidad de la iluminación.	Sotomayor, Gómez, Cela, Fernando. Escuela Politécnica Nacional.
Métodos de visión por computador para detección automática de líneas de cultivo curvas/rectas y malas hierbas en campos de maíz.	En este caso se investigará en las Bases de Datos Bibliográficas los diferentes métodos y estrategias de procesamiento de imágenes para su aplicación.	Propone dos nuevos métodos para la detección de líneas de cultivo en curvas y rectas en campos de maíz.	Iván Danilo García Santillán. (2017). Universidad Complutense de Madrid.
Subsistemas de detección de ubicación adaptables de alta precisión y bajo costo para vehículos autónomos en agricultura de precisión	Analizará la precisión y el tiempo de respuesta en que procesa el sistema las fotografías en tiempo real	Trabajo en conjunto del procesamiento de imágenes en tiempo real y sensores de aproximación que evalúan la ubicación del vehículo en el surco.	Levo ir, Samuel J. Far ley, Peter A. Sun, Tao Xu, Chong (2020)
Enfoque de aprendizaje profundo basado en la visión para la detección en tiempo real de malezas en la agricultura orgánica.	Utilizará técnicas de visión artificial, tecnología que en la actualidad sigue siendo eficiente.	Detecta malezas en campos de zanahorias en tiempo real utilizando una red neuronal de convolución para localizar y clasificar las plantas simultáneamente.	Czymmek, Vitali Harders, Leif O. Knoll, Florian J. Hussmann, Stephan 2019.

CAPÍTULO 1

Marco Teórico

El presente capítulo realiza una recopilación de diferentes investigaciones realizadas en el campo de la visión por computador enfocada a la agricultura. Se expone un breve resumen acerca de trabajos relacionados, incluyendo los resultados obtenidos en cuanto a tiempos y eficiencia. Se mencionan los conceptos importantes en el procesamiento de imágenes, conceptos de segmentación, extracción y clasificación, enfocándose en las diferentes actividades, como el conteo de plantas, la fumigación, detección de enfermedades y plagas. También se destaca los diferentes algoritmos, técnicas y protocolos involucrados en la construcción de los sistemas de detección en tiempo real, se toma en cuenta la información de conjuntos de datos existentes; los mismos que sirven de sustento para el desarrollo del proyecto.

1.1. Agricultura de Precisión (AP)

El manejo agronómico y medioambiental de la actual producción agraria depende de la inversión en la innovación tecnológica en el sector, debido al aumento diario de la población mundial se hace necesario el incremento de la producción agrícola, según la Organización de las Naciones Unidas para la Alimentación y la Agricultura FAO, la producción agrícola deberá proliferar en un 70% para el 2050, las tendencias de investigación se centran en el cambio climático, necesidad de agua dulce y compensación a la falta de tierras necesarias para el cultivo, y otros aspectos que mejoren la productividad, el uso de la tecnología y principalmente de sistemas de visión artificial se está convirtiendo en una necesidad debido al gran potencial que presenta la AP, esta, comprende la integración de estrategias de varias ciencias e ingeniería sustentadas en tecnologías geo espaciales, sensores remotos, dispositivos tecnológicos, software especializado y aplicaciones de control automático (Jimenez et al., 2015).

Las principales tareas agrícolas que se automatiza son: La detección de malezas, monitoreo de cultivos, detección de enfermedades, detección de plagas, recuento de plantas y alimentos, características de suelo, fumigación, entre otras (Pusdá-Chulde et al., 2020).

Una de las partes fundamentales de la AP es la variabilidad espacial y la variabilidad temporal, la espacial busca identificar características de medición específicas para supervisar todas las variaciones del cultivo (la vegetación, el agua, la humedad, la composición del suelo, la topografía, el estado de las enfermedades y plagas de las plantas), la variabilidad temporal tiene como fin identificar las características particulares que afectan el rendimiento del cultivo como las propiedades del suelo (Mozaffari et al., 2019).

Con el fin de obtener una mejor gestión de suelos y cultivos agrícolas amigables al medioambiente, exístete la combinación de distintos procesos de análisis y tecnologías relacionadas con todas las etapas de la producción de un cultivo, desde la siembra hasta la cosecha, teniendo como las principales:

a) Sistemas GPS
b) Sistemas GIS
c) Mapeo de producción
d) Mapeo de suelos
e) Mapeo de la conductividad eléctrica del suelo (EC)
f) Tecnologías RS

El elemento integrado más utilizado en el desarrollo de un sistema de visión por computador es una cámara, comúnmente estas van fijas o en un UAV, para estos, existen tres tecnologías principales de cámaras, multiespectrales, hiperespectrales y térmicas. Las cámaras multiespectrales integran cinco bandas (rojo, verde, azul, borde rojo e infrarrojo cercano), la cámara hiperespectral incluye bandas en contraste con el primer caso, llegando al número de 2000 nm y la cámara térmica emplea la radiación infrarroja para formar una imagen de zona de calor, operando a longitudes de onda de 14000 nm aproximadamente (Mozaffari et al., 2019).

La utilización de los análisis y las tecnologías en un ámbito de tiempo real, el cual es el propósito del proyecto, permite reducir costos, recursos y combinadas con las adecuadas técnicas, presenta un desafío para resolver problemas agrícolas con mejores resultados, razón por la cual esta investigación se desarrollará con una videocámara accesible y eficiente sin necesidad de algún vehículo aéreo no tripulado.

Las aplicaciones de la Agricultura de Precisión generalmente utilizan varias técnicas de procesamiento y análisis de imágenes, la Tabla 2 resume los métodos de procesamiento y análisis de imágenes, teniendo en cuenta los porcentajes de utilización, siendo la más utilizada la técnica de segmentación (Pusdá-Chulde et al., 2020).

Tabla 2

Técnicas de procesamiento y análisis de imágenes

Técnica	**Descripción**	**% de Utilización**
Segmentación.	Método donde se utilizan imágenes para paralelizar tecnologías heterogéneas en tiempo real, proporcionando un umbral automáticamente, con comportamientos apropiados, incluso en condiciones de iluminación cambiantes o adversas	43,8
Basadas en índices de Vegetación	Permite la extracción de características espectrales mediante la combinación de dos o más bandas espectrales basadas en la adecuada reflectancia producida por la vegetación.	25
Basadas en Aprendizaje.	Estas técnicas se basan en procesos de aprendizaje supervisados y no supervisados; aplicando deep learning, usando una red neuronal profunda, proponiendo métodos heurísticos, métodos automáticos para la identificación de malezas, etc.	21,9
Basado en Wavelet.	Resalta los detalles (bordes) y detecta las texturas, analizando la imagen desde diferentes ángulos.	9,3

Nota. Fuente extraída de (Pusdá-Chulde et al., 2020).

1.2. Sistematización de Actividades Agrícolas

La sistematización de actividades agrícolas comprende la aplicación de técnicas modernas con la adecuada gestión y organización de datos utilizando herramientas y tecnologías aplicadas en diferentes labores agrícolas. Los sistemas generalmente están basados en Sistemas de Posicionamiento Global (GPS), Sistemas de Información Geográfica (SIG) y teledetección que provee una enorme cantidad de datos sin necesidad de contacto físico derivando en información valiosa (Comba et al., 2018).

La sistematización de actividades presenta una agricultura tecnificada, mejorando la rentabilidad, aumentando el rendimiento de los cultivos y reduciendo costos de los insumos, con una protección ambiental, mayor eficiencia en el uso de los nutrientes de la planta, técnicamente denominado 'conservación de precisión' que es el uso de tecnologías y procedimientos de precisión, a través de la variabilidad espacial y temporal, para lograr los objetivos de conservación del suelo y el agua (Yost et al., 2017). La Tabla 3 presenta 5 casos de sistematización de actividades agrícolas con 3 diferentes métodos de procesamiento de imágenes.

Tabla 3

Sistemas agrícolas con diferentes técnicas de procesamiento

Sistema	Técnica de Procesamiento	Funcionamiento	Resultados
Procesamiento digital de imágenes de sensores remotos para aplicaciones de agricultura de precisión.	Índices de Vegetación	Una hoja seca y amarilla es colocada en una caja negra con dos cámaras en el interior, una de espectro visible y otra de infrarrojo, construida para el estudio, se extraen las bandas azul, verde y roja y la imagen de infrarrojo se convierte a escala de Grises y se determina el NVDI.	Las bandas azul, verde y roja tienen niveles altos de NDVI para regiones con maleza o enfermas y el infrarrojo cercano presenta valores mínimos, por lo que lo óptimo sería tener un NDVI con valores altos para tener un cultivo sin enfermedades ni maleza.
Software para el estudio de coberturas vegetales con conceptos de agricultura de precisión	Índices de Vegetación	El procesamiento busca realizar un recorrido de la circunferencia obtenida por las dos posiciones marcadas (centro y borde) en sentidos verticales y horizontales, enmarcando a todos los bordes de la planta en una sola circunferencia.	Se obtiene una imagen estudiada en las posiciones (x, y) un índice promedio de NDVI y el cálculo del Índice por área concluyendo que las hojas de cultivo poseen mayor índice de vegetación que las hojas de maleza.

Sistema De Agricultura de Precisión (PAS)	Segmentación	Los datos se dividieron para 3 zonas de interés, se evaluaron la profundidad de la capa superior del suelo, (TD) derivada de la elevación, la conductividad eléctrica aparente y el agua disponible de la planta, análisis que sirvió para elaborar cuadros estadísticos que determinaron los grados día de crecimiento acumulativo anual del cultivo.	El PAS no mejoró la producción general de granos en gran parte del campo, pero, logró aumentar la producción relativa de cereales en posiciones vulnerables y erosionadas en laderas, en 11 años de funcionamiento se documentó cambios significativos que ayudan al agricultor a tener una mejor idea del clima, del suelo y del desarrollo del grano en la producción.
Modelización de mapas de nubes de puntos tridimensionales (3D)	Índices de Vegetación	Las zonas de cultivo vitícolas son extensas, con terrenos escarpados, distribución espacial dispar e irregular, un gran conjunto de datos de puntos se convierte en un modelado de nubes de puntos en 3D, siendo esencialmente información espacial.	El algoritmo detecta automáticamente las zonas ocupadas por parcelas de viñedo dentro de un mapa de nubes de puntos determinado, evaluando la orientación local de las hileras de la vid y el espacio entre ellas, el índice de detección era mayor al 90,0%.
Reconocimiento de imágenes basado en red de cápsulas CapsNet.	Aprendizaje	plantea CapsNet que es una Red neuronal convolucional que procesa las imágenes en 5 capas: Capa de Entrada, Capa de convolución (Transforma dos funciones en una nueva), capa de cápsulas primarias, capa de cápsulas digitales y la capa de salida.	Las CNN (Redes neuronales convolucionales) tienen fallas al reconocer imágenes superpuestas cuyo fondo se superpone al objetivo de la imagen, afectando la precisión de reconocimiento y la información de posición y actitud de los cultivos,

La primera técnica encuentra los índices de vegetación que son combinaciones de bandas espectrales de una imagen donde el espectro visible representa los pigmentos de clorofila que absorben la luz violeta, azul y roja para la fotosíntesis, debido a esto las plantas en sus distintos procesos biológicos de aspectos térmicos, fotosintéticos y foto morfológicos, resultan con colores del espectro, como el azul, violeta, verde y roja. Los índices de vegetación son la combinación algorítmicamente de dos o más bandas del espectro visible e infrarrojo, el infrarrojo cercano resulta cuando se empieza a trabajar alrededor de los 800 nm de onda (Jimenez et al., 2015).

Los principales Índices de vegetación son: RVI (Índice de vegetación de proporción), NDVI (Índice de vegetación de diferencia normalizada), PVI (Perpendicular VI) y el TVI (Índice de vegetación transformada) (Jimenez & Jiménez, 2013).

La segmentación es la segunda técnica analizada, el proceso consiste en delinear regiones de interés para un objetivo dado, la conservación de cultivo / suelo ha sido evaluada en entornos a corto plazo, pero no a un largo plazo, los factores limitantes para un cultivo varían de un año a otro ya sea por el clima o por factores de suelo que se puede corregir fácilmente como el pH (nutrientes esenciales de las plantas) (Yost et al., 2017).

A partir de 1993 un campo de 36 hectáreas en el centro de Missouri Estados Unidos con características arcillosas fue estudiado, la segmentación ayudo a investigar el rendimiento de grano y otros datos espaciales como características de suelo, planta y agua, para el 2003 se desarrolló un plan de manejo del cultivo llamado PAS (Sistema de agricultura de Precisión), el manejo de PAS consiste en delimitar zonas para aumentar la producción y la rentabilidad de los cultivos, disminuir la variabilidad de la producción de cultivos y mejorar la calidad del suelo y el agua sobre el manejo uniforme convencional de los años anteriores al PAS (pre-PAS). El PAS se desarrolló sobre la hipótesis de que la información espacial de cultivos y suelos era fundamental para decidir que cultivos y prácticas de manejo adoptar, dividiendo el campo y evaluando sus datos obtenidos mediante imágenes satelitales, al revisar la información espacial y temporal existente del cultivo se adoptaron varias prácticas para diferentes características: Condiciones Climáticas y Rendimiento, el sistema PAS detecto con mayor efectividad el cambio de las condiciones climáticas (desviaciones cálidas – húmedas) en ciertas zonas del campo y además que el suelo no varía en condiciones dependiendo el cultivo de maíz o soja (Yost et al., 2017).

Las dos técnicas anteriores muestran la búsqueda de métodos de reconocimiento de imágenes, para, proporcionar la información necesaria y determinar los Índices de vegetación, además de características que faciliten realizar una discriminación de cultivos, hallar una medición de la altura en hileras curvilíneas, evaluar a través de etapas el estado del agua, índices de estrés, plagas y enfermedades de una planta, para, así llegar a tener un monitoreo preciso en tiempo real y en forma remota.

Sin embargo, la calidad de imagen y el clima son unos limitantes que hacen mejorar diariamente los métodos, reduciendo costos y estableciendo nuevos modelos de procesamiento, uno de estos es la utilización de Redes neuronales Convulsiónales (CNN) donde interviene el aprendizaje que es la tercera técnica de procesamiento a evaluar (Li et al., 2019).

En el caso de CapsNet que es una CNN, el proceso de aprendizaje consiste en realizar la adquisición de las imágenes para dos conjuntos (de entrenamiento y de prueba), las imágenes en tiempo real son analizadas para ser seleccionadas y clasificadas: por color de hoja, forma, textura, etc. El preprocesamiento consiste en dos métodos: la ecualización del histograma que se utiliza para mejorar el contraste de la imagen y la eliminación de ruido con el algoritmo de superpíxeles que segmenta la imagen del cultivo y posteriormente ejecuta el reconocimiento. (Li et al., 2019).

1.2.1. Algoritmo Viola-Jones

El algoritmo Viola-Jones es desarrollado por Paul Viola y Michael Jones en 2001. Viola y Jones propusieron un algoritmo de detección de objetos basado en el aprendizaje en tiempo real, consolidándolo como un método de detección de objetos ampliamente utilizado para la detección de rostros (Tran et al., 2021). En el caso de detección de rostros Viola-Jones se basa en tener muestras positivas que son de rostros y muestras negativas (objetos que no son rostros) que incluyen imágenes térmicas que no son rostros e imágenes visibles que no son rostros denominados "ejemplos cruzados", lo que significa que las muestras negativas son un conjunto de imágenes térmicas e imágenes visibles. Los ejemplos cruzados pretenden aumentar la discriminación entre las muestras positivas y las muestras negativas, y mejorar el rendimiento (Tran et al., 2021).

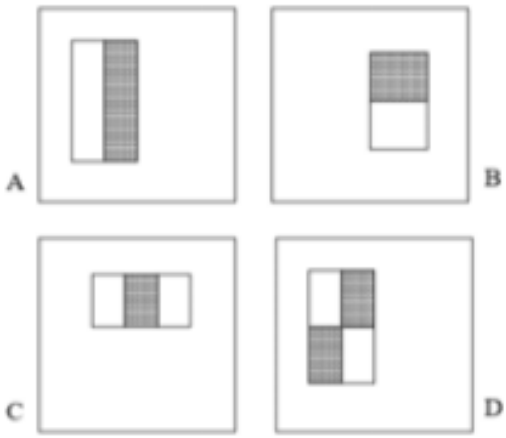
Algunas investigaciones aplican el algoritmo en 3 etapas, en la primera etapa se realiza una transformación de la imagen generando una nueva, llamada imagen integral, en el segundo bloque se realiza la extracción de características usando filtros con base haar, y por último se usa boosting para la construcción de clasificadores en cascada (Elizabeth Rodríguez, 2017). Otras Investigaciones forman al algoritmo de 4 etapas:

a) Selección de características Haar
b) Crear una imagen integral
c) Entrenamiento de Adaboost
d) Clasificadores en cascada

La Figura 3 ilustra los tres tipos de características rectangulares utilizados. El valor de una característica de dos rectángulos es la diferencia entre la suma de los píxeles dentro de dos regiones rectangulares. Las regiones tienen el mismo tamaño y forma y están adyacentes horizontal y verticalmente. Una característica de tres rectángulos calcula la suma dentro de dos rectángulos externos que se restan de la suma en un rectángulo central. Finalmente, una característica de cuatro rectángulos calcula la diferencia entre pares diagonales de rectángulos (Rocha et al., 2019).

Figura 3

Clasificador en cascada algoritmo Viola-Jones



Nota. Fuente extraída de (Rocha et al., 2019).

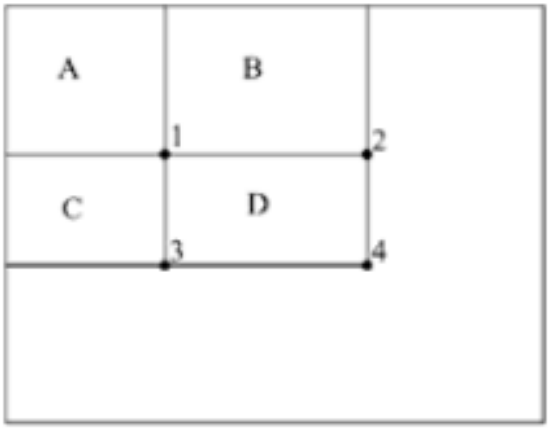
Con el objetivo de reducir el tiempo de cálculo necesario para evaluar características rectangulares de distintas zonas de la imagen a diferentes escalas, el método utiliza una nueva representación de la imagen denominada imagen integral que se calcula en la siguiente ecuación (Rocha et al., 2019).

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y')$$

Donde $ii(x, y)$ es el valor de la posición (x, y) de la imagen integral e $i(x', y')$ es el valor del píxel de la imagen original en (x', y'). La imagen integral puede calcularse en una única pasada de la imagen original si se acumula la suma por filas en otra matriz auxiliar (Rocha et al., 2019). La Figura 4 muestra de una forma gráfica el mencionado cálculo de imagen integral.

Figura 4

Imagen integral de Viola-Jones



Nota. Fuente extraída de (Rocha et al., 2019).

El valor de la imagen integral en la ubicación 1 es la suma de los píxeles en el rectángulo A. El valor en la ubicación 2 es A + B, en la ubicación 3 es A + C, y en la ubicación 4 es A + B + C + D. Luego, la suma dentro de D se puede calcular como 4 + 1 - (2 + 3) (Rocha et al., 2019).

El entrenamiento de adaboost propone entrenar una serie de clasificadores débiles de manera iterativa, de modo que, cada nuevo clasificador “weak learner” se enfoque en los datos que fueron erróneamente clasificados. La Figura 5 presenta un diagrama general del clasificador de rostros analizado.

Figura 5

Esquema del entrenamiento en cascada.

Nota. Fuente Extraída de (Elizabeth Rodríguez, 2017).

La cuarta etapa que es la clasificación en cascada consiste en asignar un conjunto de características dado a una clase con la que se encuentra una mayor similitud, de acuerdo con un modelo inducido durante el entrenamiento (Elizabeth Rodríguez, 2017).

1.3. Lenguajes de programación para el procesamiento de imágenes

Hoy en día la popularidad de la inteligencia artificial, especialmente la rama de Machine Learning (ML), ha crecido de forma exponencial, siendo el procesamiento de imágenes parte de ella, a su vez, los lenguajes de programación que se pueden utilizar, algunos de los lenguajes de programación utilizados en ML son R, Python, Julia y Matlab, entre otros. En la industria del desarrollo de software cada lenguaje busca dar respuesta a la complejidad de manejo de datos para generar información oportuna y aportar innovación tecnológica, el lenguaje se decide dependiendo el nivel de comodidad y la experiencia previa del desarrollador (Rojas, 2020).

Al construir un modelo de machine learning, es necesario conocer la clasificación de los tipos de algoritmos de aprendizaje de ML y el proceso que implica construirlo para entender que la selección del lenguaje es fundamental.

- Aprendizaje supervisado: se enseña al algoritmo cómo realizar su trabajo con un conjunto de datos clasificados bajo una cierta apreciación o idea para encontrar patrones que puedan aplicarse en un análisis y producir una salida que ya se conoce (Rojas, 2020).
- Aprendizaje no supervisado: consiste en un entrenamiento de manera similar al aprendizaje supervisado, pero la diferencia es que la comprensión se da en datos no clasificados o etiquetados y se encuentra patrones de ejemplos similares entre grupos de datos (Alfons Crespo & Alejando Alonso, 2006).
- Aprendizaje reforzado: no existe capacitación con datos clasificados o no clasificados; el sistema aprende en un entorno donde no hay información sobre la posible salida, a través de acciones y los resultados obtenidos, el modelo se refuerza al resolver el problema de la mejor manera (Rojas, 2020).

En la Figura 6 se puede observar detalladamente los pasos del proceso para la construcción de un modelo predictivo, generalmente implica al menos 6 pasos.

Figura 6

Proceso de construcción del modelo machine learning

Nota. Fuente Extraída de (Rojas, 2020).

Al tener estos fundamentos los lenguajes de programación han evolucionado, a continuación, la Tabla 4 presenta las características de los lenguajes más comunes para este tipo de desarrollo:

Tabla 4

Análisis de los principales lenguajes de programación para ML.

Lenguaje	Características	Librerías Principales	Ventajas	Desventajas
R	- Herramientas estadísticas. - Modelos lineales o no lineales. - Programación POO - Algoritmo de clasificación y agrupación. - Integración BDD	RandomForest - Rpart - Igraph - Outliers - Survival - Forecast - Nnet	- Mayor análisis de datos y soporte estadístico	- Puede ser un poco lento en comparación con otros lenguajes de programación.
Python	- Gestión de memoria, no se sobrecarga con las tareas de liberar y asignar memoria, gestiona la basura. - Agilidad (Sintaxis simple) permite ser rápido en desarrollo. - Tipado Dinámico	- Scikit-Learn - Statsmodels - PyMC - NTLK - Pandas - Numpy - TensorFlow	- Python tiene una comunidad grande en internet por lo cual se encuentra soporte del lenguaje. - Lenguaje fácil de aprender. - Depende mucho de sus librerías o paquetes - Paralelismo, técnica de programación fundamentada en la idea de dividir los inconvenientes de gran magnitud en apartados de menor tamaño para, de este modo, poder resolverlos en paralelo o de manera concurrente.	- Desarrollo móvil.
Julia	- Buen rendimiento y soporte para usos interactivos. - Útil en análisis Matemáticos. - Control de procesos - Aceleración en el GPU	- Flux, librería que convierte el código en modelos entrenables con diferenciación automática.	- Velocidad - Gestión automática de la Memoria como la de Python. - Sintaxis orientada a las Matemáticas. - Paralelismo mejor que Python.	- Primera ejecución lenta ya que el compilador tiene que analizar y compilar el código
Matlab	- Ejecución más rápida que con código abierto en la mayoría de los cálculos estadísticos y el aprendizaje automático. - Posibilidad de usar el mismo código para escalar el procesamiento de big data y clusters.	- Math Library - Math Toolbox	- Amplio soporte de funciones ya desarrolladas. - Rápido prototipado - Integración	- Lenguaje de paga que necesita una licencia para su uso.

Nota. Elaboración Propia

El uso de lenguajes específicos de tiempo real presenta un enfoque con ventajas en portabilidad y fiabilidad, lenguajes como Ada y Java proporcionan mecanismos fundamentales para este tipo de programación. Ambos lenguajes cuentan con una programación concurrente que les permite ser rápidos, además de contar con extensiones maduras e innovadoras contando con un gran número de soporte por parte de la comunidad (Alfons Crespo & Alejando Alonso, 2006).

En los sistemas de guía UAV en tiempo real, se repite un modelo estándar donde la arquitectura del software está dividida en tres módulos, el primero es el sistema operativo, con una interfaz GUI deshabilitada para un mejor rendimiento, este SO es elegido debido a la compatibilidad con las librerías de software necesarias para el sistema desarrollado. El segundo módulo está formado por las librerías de software, utilizadas como: la cámara Raspberry Pi (Raspicam) permiten la comunicación con la interfaz para la adquisición de imágenes, Open CV, proporciona recursos para la implementación de los algoritmos desarrollados, el protocolo de comunicación de UAV micro (MAVLink), permite enviar mensajes / comandos al controlador de vuelo del UAV y el tercer módulo corresponde al software que implementa los algoritmos propuestos, estos algoritmos están desarrollados en el lenguaje que presentan un mejor rendimiento (Basso & Pignaton de Freitas, 2020).

Se pueden realizar aplicaciones en diferentes lenguajes de programación, la implementación en Python como en C ++ se considera una buena práctica de codificación para el desempeño, los dos ayudan al cómo evitar estructuras complicadas y evitar parámetros excesivos. Las funciones de la biblioteca OpenCV para operaciones matemáticas en matrices se utilizaron para evitar las estructuras "para" y están disponibles en los dos lenguajes, además, la biblioteca OpenCV se configura para usar el hardware de la GPU a través de componentes del lenguaje informático abierto (OpenCL), también, se utilizó MATLAB para procesar imágenes de video de semillas de trigo, se analizaron los resultados y se concluyó que la imagen del marco de la semilla de trigo procesada era clara, MATLAB mejoró la

precisión y velocidad de detección en comparación de C y C ++, mientras que tenía un proceso de operación simple y una amplia perspectiva de aplicación (Viera, 2018).

1.4. Sistemas en tiempo real

En el estudio de detección de sistemas en tiempo real se identifican distintos métodos o algoritmos que son fundamentales en el proceso de plantear el cómo se va a desarrollar el proyecto, a continuación, se realiza un análisis de algunos proyectos y una descripción de la eficiencia de los métodos tradicionales y la de los métodos basados en aprendizaje profundo.

1.4.1. Detección de malezas/cultivos

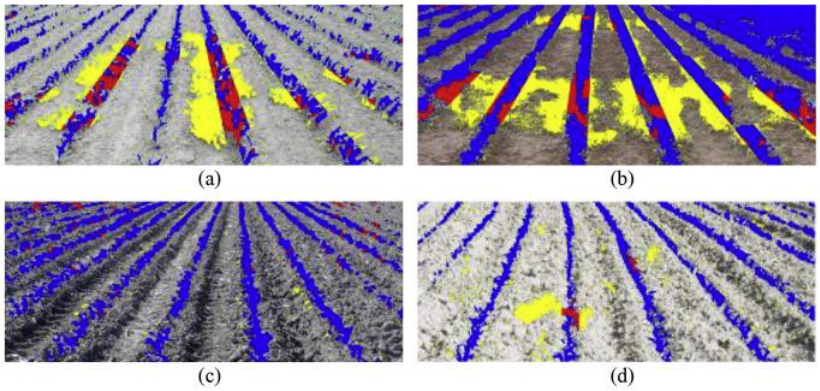
Las malezas compiten con las plantas de cultivo y pueden tener un impacto negativo en el rendimiento, como los herbicidas tienen efectos dañinos, las prácticas de manejo que utilizan herramientas químicas sugieren aplicar herbicidas localmente dentro de la dosis estrictamente necesaria, a escala de campo, esto requiere herramientas capaces de identificar y localizar las malas hiervas entre plantas de cultivo, entre hileras o dentro de hileras (Ahmad et al., 2018).

Distancia de mahalanobis

Un método para discriminar maleza en campos de maíz para las etapas iniciales de crecimiento (hasta 40 días) es el criterio de similitud, basado en la distancia de mahalanobis, el método propuesto consta de tres fases principales: segmentación, capacitación y pruebas, todas las fases se ejecutan en línea, todos los procesos correspondientes ejecutándose en tiempo real, de modo que los datos de entrada de cada imagen se adquieren, actualizan y procesan inmediatamente. La Figura 7 muestra como el método, ha demostrado ser adecuado en condiciones de iluminación incontroladas mediante el uso de una cámara a color instalada en la parte delantera de un tractor con proyección de perspectiva de imágenes (García-Santillán & Pajares, 2018).

Figura 7

Identificación de cultivo / maleza con la distancia de mahalanobis.



Nota. Las plantas de cultivo están etiquetadas en azul, las malezas entre hileras en rojo y la distancia entre hileras en amarillo. Fuente extraída de (García-Santillán & Pajares, 2018).

Mahalanobis identifica las plantas de los cultivos y también las malezas tanto entre hileras como dentro de ellas, logrando una precisión del 91,8% y un coeficiente Kappa (medida estadística que ajusta el efecto del azar en la proporción de la concordancia observada) de 0,77, cumpliendo los requisitos de valores comúnmente aceptados y está de acuerdo con las prestaciones de tres métodos existentes con mejora, así como con un tiempo de procesamiento adecuado siempre inferior a 280 ms (García-Santillán & Pajares, 2018).

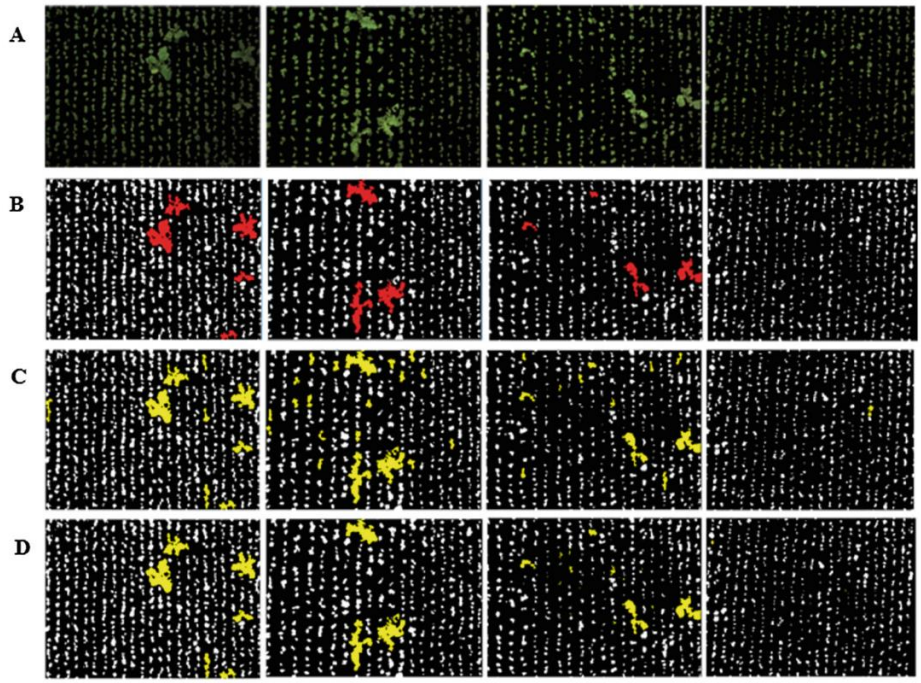
Método de MExG (verde de exceso modificado)

Este trabajo presenta un sistema de procesamiento de imágenes basado en las características e intensidad de las plantas extraídas de imágenes RGB, las adquisiciones de imágenes se realiza en condiciones reales, en un invernadero de plantas de lechuga a condiciones diferentes de luz, son capturadas por una cámara de 20 mpx, con una resolución espacial de 5120x3840 px, la distancia de la cámara al cultivo es de 1,20 cm. Se implementa un algoritmo específico MExG (Verde en exceso modificado), consiste en favorecer el componente de los objetos a detectar, se establece un tamaño del radio dependiendo del tamaño de las plantas, en este caso 5 px a un umbral de 0, también se establece restricciones

donde G<R, G<B y G<a, donde a representa un valor de intensidad entre 0 y 255. La Figura 8 muestra como el algoritmo clasificador procede a utilizar los datos de entrenamiento y establece un modelo de entrenamiento, como la función es tener un método en tiempo real debe ser rápido y requerir una memoria mínima para almacenar las características obtenidas en el proceso de entrenamiento (Ahmad et al., 2018).

Figura 8

Aplicación del método de clasificación MExG



Nota. En la Figura 8 se utiliza un set de 13 imágenes A muestra las imágenes iniciales al extraer las características con el algoritmo de entrenamiento, el cual localiza la intensidad de verde en el cultivo, B presenta las imágenes reales del suelo, con la aplicación del algoritmo SVM detectando de color rojo las malas hierbas, en C se detecta 1672 objetos de vegetación, teniendo una tasa de detección de 97,3%, de acuerdo con la verdad del terreno la tasa de detección es correcta y en D esto se comprueba porque la tasa de detección obtenida es de 98,8% siendo en ambos casos porcentajes parecidos con una tasa muy buena y en un tiempo de cálculo relevante. C y D presentan maleza como objetos amarillos, teniendo falsos positivos. Fuente extradida de (Ahmad et al., 2018).

El algoritmo de segmentación exceso de verde modificado evalúa los resultados con una validación cruzada dejando claro que los datos obtenidos son independientes del entrenamiento y prueba, también mejora el algoritmo SVM (algoritmo de aprendizaje supervisado) con una función de núcleo polinomial, es decir, tiene una función de núcleo que permite analizar en una imagen características implícitas y de alta dimensión sin tener que calcular nunca las coordenadas de los datos en esta imagen, esta operación conviene más que el cálculo de las coordenadas. El desafío del método es la superposición de objetos de vegetación en el procesamiento de las imágenes, ya que separar estos objetos superpuestos hace necesario un análisis de texturas (Ahmad et al., 2018).

Modelo Linear SVM

Este método de clasificación lineal se realiza con el entrenamiento y validación de imágenes hiperespectrales de cultivos de cacao, el objetivo es clasificar los granos de cacao según su color interno, para ello, se utiliza un conjunto de entrenamiento y otro conjunto de validación del mismo día de cosecha, el total de imágenes hiperespectrales de granos de cacao que conformó el conjunto de entrenamiento y el de validación fue 300 obtenidos de 8 bellotas, utilizando los espectros de los granos de cacao y sus respectivos índices espectrales, se crearon, entrenaron y validaron distintos modelos de clasificación, de estos se escogió el de mejor rendimiento resultando el modelo Linear SVM con un porcentaje promedio de acierto de 86.8%. La aplicación de Matlab llamada “Classification Learner” comprobó el porcentaje de acierto de los distintos modelos de clasificación con la estrategia de validación cruzada, para tener una idea del rendimiento de cada uno. Para el uso correcto de la información obtenida de imágenes hiperespectrales, es importante realizar un procesamiento adecuado de la imagen, existen distintas herramientas de software para el procesamiento y análisis como ENVI y MATLAB, pero los problemas en el análisis es la baja velocidad de cálculo y la redundancia de información en algunas aplicaciones se usan como referencia para desarrollar sistemas de imágenes

multiespectrales utilizando 3 - 4 bandas con un costo relativamente bajo y alta velocidad analítica, las bandas espectrales utilizadas en los índices hiperespectrales permitirían diseñar un sistema multiespectral (Viera, 2018).

1.4.2. Comparativa de métodos tradicionales de detección de malezas con métodos de deep learning.

La mayoría de los estudios utilizan algoritmos de aprendizaje automático combinados con características de imagen para realizar tareas de identificación de malas hierbas, estos métodos de ML requieren una muestra y un tiempo, son de bajo coste y pueden usarse en una variedad de maquinaria agrícola, siendo un método eficaz y útil. Básicamente, estas tecnologías utilizan una serie de métodos de procesamiento de imágenes para extraer características superficiales (textura, forma, color, o imágenes espectrales) de las malezas para después enviarlas a un clasificador para su detección. El uso de imágenes de drones para la detección es una práctica común, los UAV están menos limitados por las condiciones del cultivo, y tienen la ventaja de observar zonas de malezas a gran escala, además las imágenes ya ofrecen alta resolución y flexibilidad al momento de la toma (Wu et al., 2021).

En la Tabla 5 se realiza un análisis de la precisión de algunos métodos tradicionales de aprendizaje automático para la detección de malas hierbas, con las desventajas en el proceso.

Tabla 5

Ventajas y desventajas de los métodos tradicionales de aprendizaje automático.

Método	Precisión	Desventajas
Combinación de la característica HOG con la máquina de vectores de apoyo SVM para identificar hojas de uva.	83,50%	La detección de una sola característica no es suficiente.
Identificación de diferentes hojas de plantas a partir de LBP mejorados.	79,35%	La estabilidad y baja precisión de los LBP (patrones binarios locales), no se formula bien la imagen binaria.
Combinación de matrices de Gabor y de concurrencia de grises (GLCM) para clasificar hojas de plantas.	91,60%	No se incluyen imágenes reales sobre el terreno, y el conjunto de datos se compone por imágenes sin fondo.

SVM o red neuronal artificial ANN en la detección de remolacha azucarera y malas hierbas.	93,33%	Falta un análisis sobre la selección de características.

Nota. Fuente extraída de (Wu et al., 2021)

La detección de objetos y principalmente de malas hierbas con aprendizaje profundo nace con la facilidad de captura de imágenes, el coste del hardware reducido y la potencia de cálculo de una GPU mejorada, notablemente el deep learning se basa principalmente en el uso de redes neuronales de varias capas, la Tabla 6 presenta trabajos de deep learning con su precisión en la detección de malezas.

Tabla 6

Trabajos de deep learning en la detección de objetos

	Método	**Precisión**
Detección unificada de objetos en tiempo real.	YOLO Sistema de entrenamiento con imágenes completas que optimiza directamente el rendimiento de detección.	89%
Aprendizaje profundo basado en tiempo real para la detección de malas hierbas	Red Neuronal convolucional CNN	75% mAP
Sistema de identificación de malezas WIS de bajo coste usando imágenes RGB tomadas por drones.	CNN	98,8%

Nota. Fuente extraída de (Redmon et al., 2016), (Czymmek et al., 2019)

1.4.3. Protocolo de Transmisión en tiempo real RSTP

El protocolo de transmisión en tiempo real, o RTSP es un protocolo de nivel de aplicación para controlar la entrega de datos con propiedades en tiempo real. Proporciona un marco extensible para permitir la entrega controlada y bajo demanda de datos en tiempo real, como audio y video. Las fuentes de datos pueden incluir datos en vivo y clips almacenados, está diseñado para controlar múltiples sesiones de entrega de datos, proporcionar un medio para elegir canales de entrega como UDP, de multidifusión y TCP, además de proporcionar un

medio para elegir mecanismos de entrega basados en RTP (Schulzrinne, 1998).

RTSP originalmente estaba destinado a ser utilizado para ofrecer televisión de entretenimiento, ahora solo se puede encontrar en cámaras IP, se le comparará con un control remoto de TV, ya que se puede usar para iniciar, pausar, detener, etc. el video en el servidor de medios, en otras palabras, RTSP actúa como un "control remoto de red" para servidores multimedia (Mutchima & Sanguansat, 2010).

Porque no usar UDP, porque los datos simplemente se pierden en algún lugar del camino en un ejemplo de videovigilancia al robarse la video grabadora se roban todos los datos que contiene, usando el protocolo RTSP, se puede enviar la información a un servicio de nube remoto y así los datos no se perderán, además RTSP reutiliza mecanismos de seguridad web, es compatible a todos los mecanismos de autenticación HTTP (Schulzrinne, 1998).

1.5. Detección de Plagas / Fumigación de Agroquímicos

Los Insectos son una de las mayores causas de la perdida de los productos durante las operaciones de postcosecha, los métodos tradicionales de detección (infrarrojo cercano, métodos acústicos, conductividad eléctrica) ofrecen desventajas tales como la dificultad del muestreo, velocidad lenta y trabajo manual que supone tiempo, los métodos de procesamiento de imágenes basado en aprendizaje profundo proponen una manera efectiva que reduce costos y tiene mayor rendimiento. Uno de los mayores intereses de la informática en el campo de la agricultura es la detección de enfermedades causadas en su mayoría por insectos, Para un agricultor es muy difícil prestar atención e identificar las diferentes enfermedades de los cultivos, además tiene que buscar el remedio para cada una de ellas, este enfoque manual lleva mucho tiempo y requiere precaución al escoger plaguicidas. La captura y procesamiento de imágenes en hojas infectadas mediante un sistema automatizado con técnicas de procesamiento y de aprendizaje profundo puede brindar varias alternativas en diferentes estados de identificación de enfermedades de las plantas (Shi et al., 2020).

En el campo de detección de insectos basada en la visión por computador se tienen en cuenta características como la textura y forma, existen estudios donde la identificación de los insectos es por la estructura de las alas, pero, el problema se da al querer detectar insectos de grano, estos son fáciles de confundirse por las particularidades de cada grano, debido a esto se propuso un modelo de red neuronal de aprendizaje profundo para clasificar y contar el número de polillas teniendo como resultado un 88,02% de precisión. Para una detección rápida y precisa de insectos se propone una arquitectura de red neuronal mejorada R-FCN, basada en una CNN para la extracción de características, una red de regiones propuestas ROI y un mapa de puntuación sensible a la posición de los objetivos (Shi et al., 2020).

Como un ejemplo para la detección de enfermedades, en la vid con imágenes obtenidas por UAV se combina la fuerza del enfoque de aprendizaje profundo de las CNN en diferentes espacios de color e índices de vegetación teniendo el cálculo de los diferentes espacios de colores (RGB, HSV, LAB, YUV) y los índices de vegetación (ExG, ExR, ExGR, GRVI, NDI, RGI) para obtener una combinación relevante de los espacios de características que conduzca a una alta precisión: Primero, la imagen del viñedo se divide en varios bloques, los índices de espacio, color y vegetación se extraen de cada bloque de imágenes y esta información alimenta al modelo de CNN, teniendo similares tipos de datos de entrada, para determinar las clases de pertenencia de los bloques, el mapa de enfermedades se genera a partir del proceso de clasificación, después que, el mapa de enfermedades se refina con pos procesamiento utilizando morfología matemática, eliminación de áreas pequeñas y detección de contornos (Kerkech et al., 2018).

El uso de agroquímicos en los cultivos es fundamental para mantener la calidad y escalabilidad de la producción al combinar el uso de vehículos aéreos no tripulados con un sistema de localización para la pulverización evita la compactación del suelo, en comparación con el uso de maquinaria pesada, y también reduce el desperdicio de agroquímicos, en comparación con la pulverización ordinaria con un avión convencional, teniendo como

desventajas las condiciones climáticas que pueden desviar el UAV o un fallo del sistema que puede provocar que se invada campos de cultivos vecinos donde la aplicación de estos agroquímicos no se recomienda (Basso & Pignaton de Freitas, 2020).

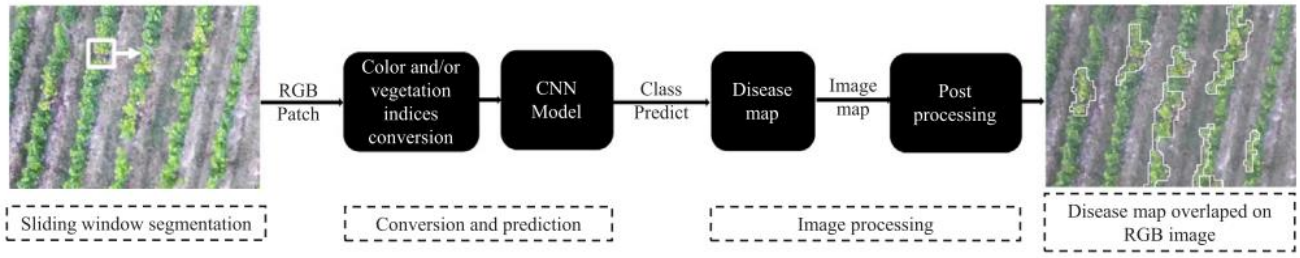
La detección de Plagas y la fumigación de agroquímicos son dos ámbitos que tienen que ir de la mano porque lo óptimo es lograr combinar el hardware con el software para obtener un sistema que detecté plagas y demás amenazas de la planta, las fumigué sin que haya una afectación grande de suelo ni de los cultivos, para lograr un sistema capaz de detectar y fumigar plagas y amenazas de la planta se tiene que pasar por un proceso de conteo de plantas y alimentos, teniendo en cuenta las diferentes técnicas de procesamiento de imágenes, las metodologías propuestas para un conteo eficaz de plantas y alimentos utilizan operadores de morfología matemática para identificar las fallas en los cultivos en hileras, es decir busca aplicar diferentes operadores matemáticos secuencialmente, para resaltar fallas que de otro modo estarían ocultas, las operaciones morfológicas se basan en la teoría matemática de conjuntos estableciendo las primeras nociones teóricas de dilatación y erosión (Oliveira et al., 2018).

La dilatación es lo opuesto al operador de erosión, centrándose en cambio en la expansión de objetos a partir de una imagen binaria, en otras palabras, la dilatación aumenta o extiende los objetos en una imagen binaria, como una forma particular, también es útil para unir partes rotas de un objeto en la imagen. La erosión tiene como objetivo principal la reducción de elementos en una imagen, A través de la erosión, es posible eliminar pequeños ruidos, separar objetos conectados y eliminar componentes más pequeños (Oliveira et al., 2018).

La Figura 9 presenta la estructura del proceso de detección de características como las enfermedades en las hileras de cultivo con una red neuronal CNN.

Figura 9

Arquitectura del sistema de detección de enfermedades con CNN



Nota. Fuente extraída de (Kerkech et al., 2018).

Con las imágenes obtenidas mediante una cámara fotográfica profesional de las distintas enfermedades de la planta se procede al procesamiento previo de Imágenes, donde se elimina el ruido (polvo, gotas de roció, excrementos de insectos, etc.) y demás acciones como el tamaño, definición, etc. que dejen en óptimas condiciones la imagen.

El siguiente paso es la segmentación que divide la imagen en regiones u objetos particulares para analizar los datos de la imagen y poder extraer las características útiles de los datos. Existen dos formas de realizar la segmentación: basada en discontinuidades, en la cual una imagen se divide en función de cambios repentinos en los valores de intensidad, por ejemplo, mediante la detección de bordes. Basado en similitudes, donde las imágenes se dividen en función de los criterios predefinidos específicos, ejemplo, el umbral realizado mediante el método de Otsu (Shah et al., 2016).

La Tabla 7 presenta una comparación de varias técnicas de segmentación utilizadas en la detección de enfermedades de las hojas con algunas ventajas y desventajas que tienen en su implementación.

Tabla 7

Técnicas de segmentación más utilizadas

Técnica	Tipo Umbral	Tipo Segmentación	Complejidad	Efecto	Ventajas	Desventajas
Método de Otsu	Global	Umbral	Muy Alto	Bueno/Estable	Funciona con imágenes reales	Mayor tiempo de Procesamiento
Basado en energía FERMI	Global	Umbral	Bajo	Mejor en comparación con Otsu y K-means	Supera la limitación de la selección adecuada de valor umbral	Funcionamiento solo en iluminación no uniforme
K-means	Global	Agrupación	Bajo	Distingue con precisión las regiones infectadas o no de la planta	Minimiza la suma de la distancia cuadrada entre el objeto y el centro	Es difícil predecir k con un número fijo de grupos.
Umbralización con nivel de Grises	Global	Umbral	Normal	Mas preciso comparado con Otsu	Proporciona contraste por enfermedad región por y antecedentes	Cada vez es necesario seleccionar el valor de umbral adecuado para obtener un mejor resultado en la segmentación
Difuso C-means	Global	Agrupación	Elevado	Mejor en comparación con Otsu y K-means	Utiliza membresía parcial; Util para problemas reales	Sensible a la condición de inicialización del clúster número y centro del grupo

Al identificar las características inherentes o características de los objetos presentes en la imagen (color, forma y textura), se utiliza la extracción de funciones, se puede utilizar para describir el objeto y el color porque es una característica importante porque ayuda a diferenciar una enfermedad de la otra, cada enfermedad puede tener una forma diferente y la textura significa como se dispersan los patrones de color de la imagen (Shah et al., 2016).

Generalmente para que un sistema realice un aprendizaje automático se distinguen tareas como la clasificación y agrupación.

La Clasificación mapea los datos en grupos o clases específicos, generalmente se denomina enfoque de aprendizaje supervisado, la clasificación es un proceso de dos pasos: fase de Aprendizaje (Entrenamiento, donde el algoritmo aprende a clasificar con base en las funciones extraídas) y modelo de clasificación, los datos de prueba se utilizan para estimar la precisión del modelo entrenado y evaluando qué tan bien se desempeña (Li et al., 2019).

La agrupación en clústeres o también denominado segmentación de datos, es un proceso de agrupación de datos en diferentes grupos en función de la similitud que estos tienen, los puntos de datos con los objetos similares se agrupan en un grupo y los objetos diferentes se agrupan en otro grupo, la agrupación es un enfoque de aprendizaje no supervisado y puede agrupar píxeles de intensidad similar en un grupo y otros píxeles de intensidad diferente en otros grupos (Li et al., 2019).

El aporte del trabajo es utilizar tecnologías informáticas y de comunicación para la construcción de sistemas automatizados que puedan notificar tempranamente las enfermedades y su tipo, para un tratamiento a tiempo y lograr disminuir pérdidas.

1.6. Otras tareas

La robótica en la agricultura ha jugado un papel fundamental para superar el problema de la escasez de mano de obra y tiempos de recolección de productos hortícolas, las técnicas como el sistema de reconocimiento de frutas y control basado en la visión han sido empleados para resolver los dos principales problemas de detección de objetos en las copas de los árboles y recolección de objetos usando información de localización en la ciencia de la horticultura, la tecnología y los negocios envueltos en la producción de hortalizas con destino de consumo requieren investigación adicional basada en varias necesidades, por lo cual se presentan técnicas de agricultura de precisión para horticultura (Tiwari et al., 2019).

En la Tabla 8 se describe algunos proyectos de la agricultura de precisión que tienen resultados óptimos en su aplicación.

Tabla 8

Diferentes sistemas de la agricultura de precisión

Sistema	**Funcionamiento**	**Componentes**	**Resultados**
Técnicas de aplicación de herbicidas	Un sistema basado en sensores láser operado por tractores de dos hileras aplicador de herbicida fue desarrollado en ICAR-CIAE, rociará herbicida entre hileras malas hierbas.	Un sensor láser para detectar las malas hierbas entre hileras y parte del aplicador, que tiene controlador, válvula y boquillas para la aplicación de herbicidas	El sensor láser está entrenado para detectar el color verde con un umbral de + 10% (color verde para el que está entrenado) el porcentaje de no localización varió entre el 5 y el 26%.
Sistema de recolección automatizado			Se han propuesto, investigado y analizado diferentes métodos practicados desde principios de la década de 1960 para la recolección mecánica de frutas, pero muchos de los sistemas automatizados de cosecha no están listos para la comercialización debido a su baja eficiencia, baja inteligencia, y alta inversión inicial.
Espectroscopia para la identificación de enfermedades en cultivos hortícolas	Una técnica de visión artificial y aprendizaje automático para la clasificación de hojas infectadas y sanas usando espectroscopia de imagen de fluorescencia, usa las imágenes para segmentar con un gráfico normalizado y extraer las características de textura con una matriz de concurrencia.	El extraído de las características se utilizaron como entrada en el clasificador, soporte máquina de vectores, para la identificación de enfermedades de cítricos	En las enfermedades incluye las propiedades de las plantas o técnicas de imágenes de detección de enfermedades basadas en el estrés como imágenes hiperespectrales y de fluorescencia.
Aplicador de pesticidas basado en sensor de dosel	El sensor rociador automático basado en huertos, entrega pesticida en aerosol solo cuando detecta el dosel de los objetivos. (El dosel da nombre al hábitat que comprende la región de las copas y regiones superiores de los cultivos)	Las señales son procesadas por una computadora personal y alimentadas en tiempo real a las boquillas de pulverización controladas por solenoide, que abre y cierra en relación con la estructura del dosel.	La aplicación precisa de los aerosoles de pesticidas apoya una disminución en la cantidad de espray entre un 45 y un 50% de insecticida / pesticida, reduciendo así tanto los costes como contaminación ambiental por productos fitosanitarios.

CAPÍTULO 2

Desarrollo

Este capítulo explica el desarrollo del software siguiendo una metodología de construcción de software, para la detección automática de líneas de cultivo y malezas mediante el procesamiento de imágenes agrícolas en tiempo real, para ello se realiza un proceso de entrenamiento, pruebas y mejoras. El análisis en tiempo real comprende la correcta sincronización del hardware con el software, a continuación, se presentan detalladamente los parámetros externos e internos que intervienen en el desarrollo.

2.1. Metodología de desarrollo

La metodología de construcción de software seleccionada fue el modelo en cascada, esta metodología se la escogió porque una aplicación de detección de objetos (maleza y líneas de cultivo) necesita concebir el trabajo como un conjunto de etapas que deben ejecutarse una tras otra, teniendo con cada fase terminada una comprobación de las exigencias y especificaciones planteadas, al realizar el entrenamiento donde el software tiene un mejor aprendizaje, necesariamente realizamos pruebas para posteriormente implementar mejoras.

2.1.1. Características de la metodología en cascada

Las principales características de la metodología en cascada que se aplica en este proyecto son las siguientes:

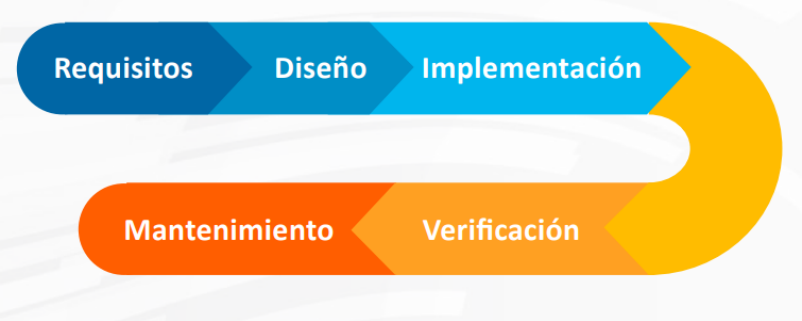
1) Modelos secuenciales
2) Cada fase tiene entregables específicos y un proceso de revisión.
3) Planificación y programación claras.
4) Es fácil medir el progreso a medida que se avanza y se cumplen los objetivos.
5) Si los requisitos no están claros al inicio, es una metodología poco efectiva, ya que se dificulta ir atrás en el proceso y hacer cambios.

6) El proceso de prueba comienza una vez finalizado el desarrollo, hay muchas posibilidades de encontrar errores que pueden ocupar tiempo en solucionar.

Todas las etapas de la metodología en cascada están resumidas en la Figura 10, teniendo en cuenta que el proyecto llegará hasta la etapa de verificación.

Figura 10

Esquema de fases de la metodología en cascada



Nota. Imagen extraída de (Jesús Demetrio Velázquez Camacho, 2012)

Los siguientes apartados comprenden el análisis, proceso y explicación detallada de las fases de la metodología en cascada utilizadas para un óptimo desarrollo del software.

2.2. Análisis y especificación de requisitos

En esta etapa se tiene un estudio de viabilidad del sistema y una definición de los requisitos, aclarando las funciones y características que debe ofrecer el producto de software para cumplir con los correspondientes objetivos.

2.2.1. Propuesta del software

El presente proyecto propone una aplicación para detección automática de líneas de cultivo y malezas mediante el procesamiento de imágenes agrícolas en tiempo real, que incluirá las siguientes funcionalidades:

1) Captura de video por medio de una cámara móvil, las imágenes se envían por una aplicación de terceros Free, a través de la red.
2) Imágenes en tiempo real, para realizar la detección de malezas y líneas de cultivo,

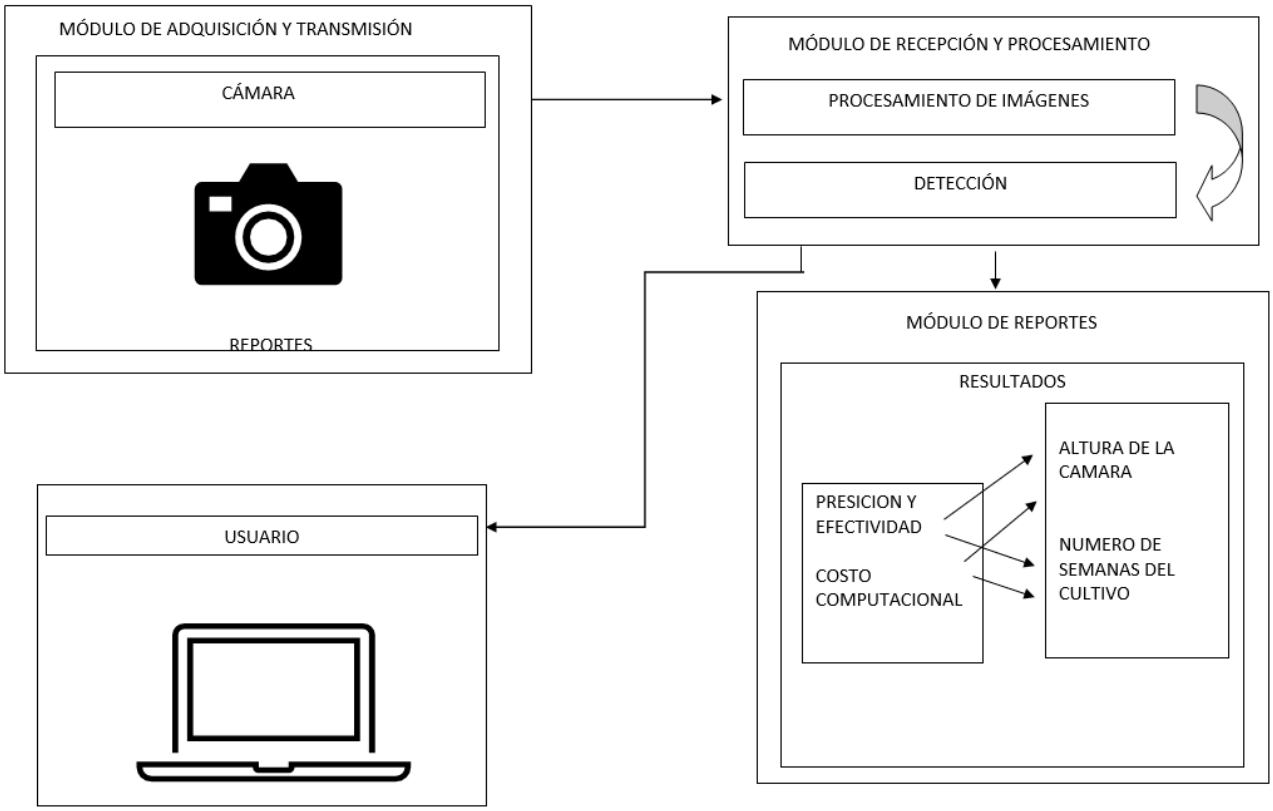
mediante un algoritmo desarrollado en Python.

3) Evaluar resultados utilizando mediciones cuantitativas sobre usabilidad, calidad, eficiencia y productividad, realizando así una medición del software.

Teniendo en cuenta las funcionalidades mencionadas, el diagrama modular específico propuesto del proyecto se muestra en la Fig. 11.

Figura 11

Diagrama del sistema propuesto con módulos, resultados y comparaciones.



Nota. Elaboración propia.

2.2.2. Requisitos Funcionales

Los requisitos funcionales son aquellos que describen el comportamiento o función del software, para esto se ha identificado los siguientes requerimientos:

Imágenes

El proceso inicial de en un sistema de visión por computador es la adquisición de imágenes. Las imágenes para el entrenamiento como las transmitidas en tiempo real deben tener una calidad aceptable, tomadas con una buena iluminación.

Clasificadores

Para realizar una detección automática el software necesita de la operación de un clasificador el cual es un algoritmo usado para asignar un elemento de entrada que no pertenece a ninguna clase ni tiene algún tipo de etiqueta en una categoría específica, dichos algoritmos ayudan a ordenar los elementos que ingresen en el sistema siguiendo alguna característica extraída.

Detección Automática

El software debe tener detección automática que comprende al aprendizaje automático refiriéndose a una de las ramas de la inteligencia artificial cuya finalidad es desarrollar técnicas que permitan a los computadores simular un aprendizaje, básicamente trata de la creación de programas y algoritmos que realicen algún tipo de tarea a partir de una serie de ejemplos suministrados previamente, si al pasar el tiempo de prueba, la información aprendida no varía, se concluye que el software puede realizar predicciones que, en función de este proyecto, sería la detección de líneas de cultivo y maleza.

2.2.3. Requisitos no Funcionales

En la Tabla 9 se encuentran los requisitos no funcionales del sistema.

Tabla 9

Requisitos no funcionales del software.

	REQUISITOS NO FUNCIONALES
1	Utilizar el lenguaje de programación Python
2	Utilizar la librería OPENCV que es una librería de código abierto para visión artificial.
3	Sistema Operativo (Linux, macOS, Windows)
4	Conocimientos básicos en teoría de la visión por computadora
5	Desarrollar la aplicación en visual Studio Code como un editor de código fuente
6	Utilizar una aplicación free para el streaming

2.3. Diseño del software

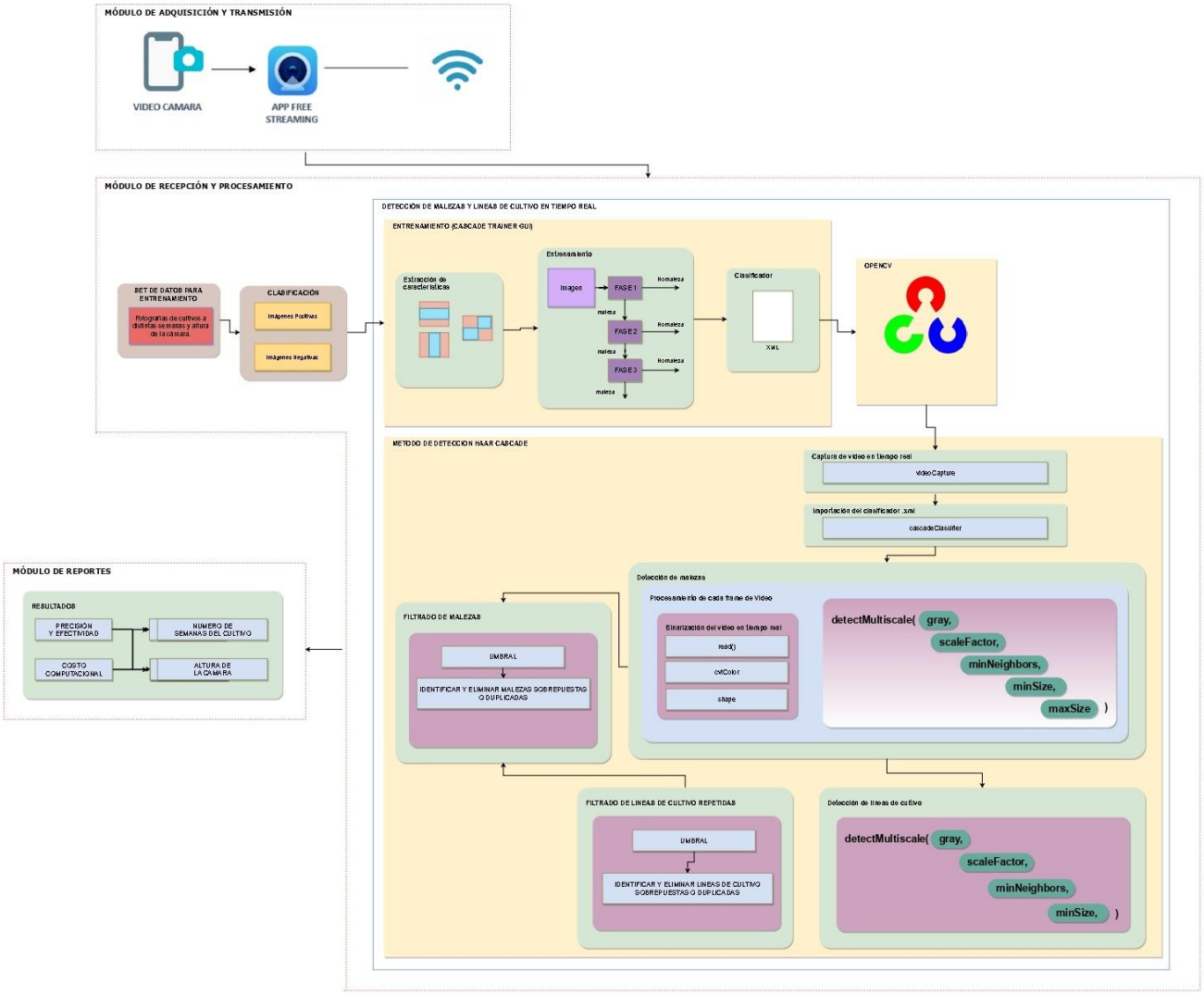
Para definir mejor la solución a las exigencias se tiene que realizar una arquitectura del software, así como un plan de diseño detallado del mismo, para esto se describe cada uno de los módulos tal como se estructura previamente en la Figura 11.

2.3.1. Arquitectura del software

Los módulos que forman el proceso de construcción del software se presentan en un diagrama secuencial general del software. En la Figura 12 se muestra un diagrama con la arquitectura del software, que busca establecer la estructuración ideal del sistema para identificar el impacto directo de este sobre la consecución del objetivo, teniendo así los atributos de calidad del sistema.

Figura 12

Arquitectura del software.



Nota. Fuente, elaboración propia.

2.3.2. Módulo de Adquisición y Transmisión de video

El módulo está constituido por cámaras IP que se encargan de capturar y enviar el video en tiempo real, con la ayuda de Camo Studio una aplicación free y a través de una red de Internet o red interna se enviaron las imágenes para que el software las analice. El proceso de entrenamiento necesitó un conjunto 4200 imágenes, positivas y negativas, las muestras de imágenes positivas son las imágenes del objeto que se desea entrenar al clasificador para su

detección, entonces se debe tener características de cultivos con plantas a diferente tamaño y diferente forma. Se debe considerar parámetros intrínsecos mínimos requeridos para la Cámara, los cuales se muestran en la Tabla 10.

Tabla 10

Parámetros intrínsecos de la Cámara

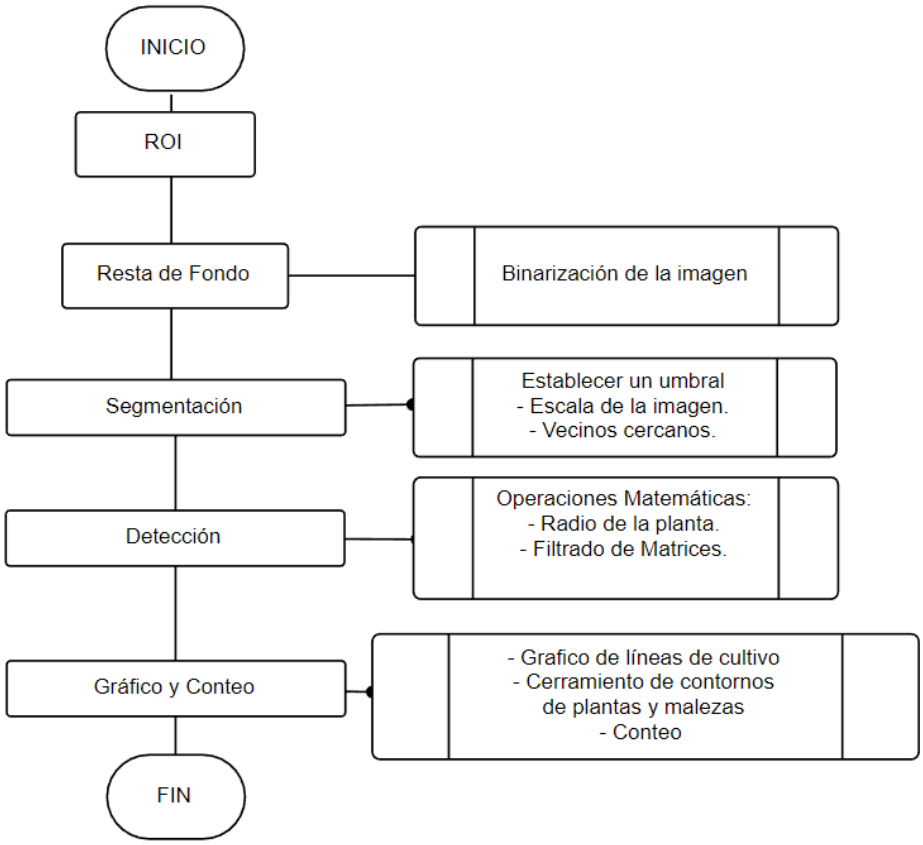
Característica	**Valor**
Resolución	VGA o superior
Altura	5 a 15 metros
Angulo	45 a 90 grados por debajo del eje X
Soporte RTSP	si

2.3.3. Módulo de Recepción y Procesamiento

Analiza los datos entrenados con el banco de imágenes previamente obtenidas y recibe el video transmitido en tiempo real desde la cámara, posteriormente ejecuta el algoritmo de detección de malezas y líneas de cultivo en tiempo real. Para una comprensión clara y precisa, se realiza un diagrama general del algoritmo con todos los procesos y después se detalla cada uno con los subprocesos específicos. En la Figura 13 muestra el diagrama general del proceso de detección de malezas y líneas de cultivo.

Figura 13

Proceso para detección de malezas y líneas de cultivo



Nota. Elaboración propia

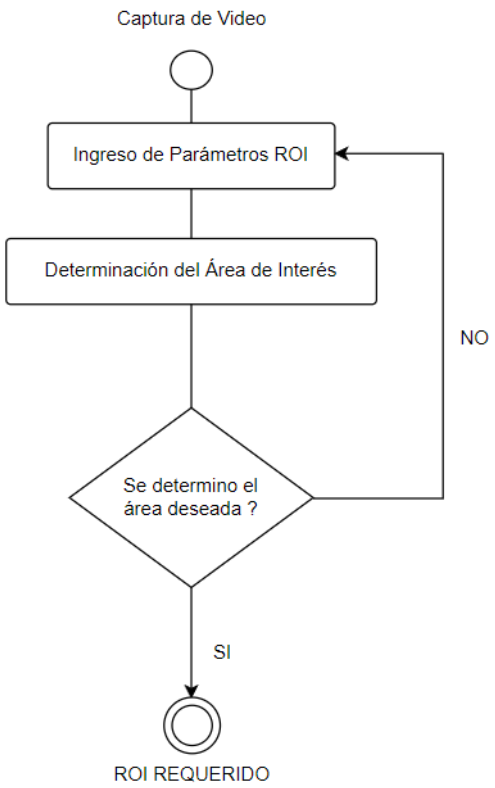
A continuación, se describe cada uno de los subprocesos que intervienen en el desarrollo.

ROI o Área de Interés

Proceso donde se establece el área de interés en el video de tiempo real, el objetivo es no procesar áreas que no influyan en nada referente a la detección de malezas y líneas de cultivo, evitando generar un exceso en el consumo de recursos de procesamiento, estas métricas fueron tomadas del apartado se construyó un proceso eficiente utilizando los menores recursos posibles. La Figura 14 detalla el diseño del proceso que se sigue para encontrar el ROI.

Figura 14

Proceso de búsqueda del área de interés.



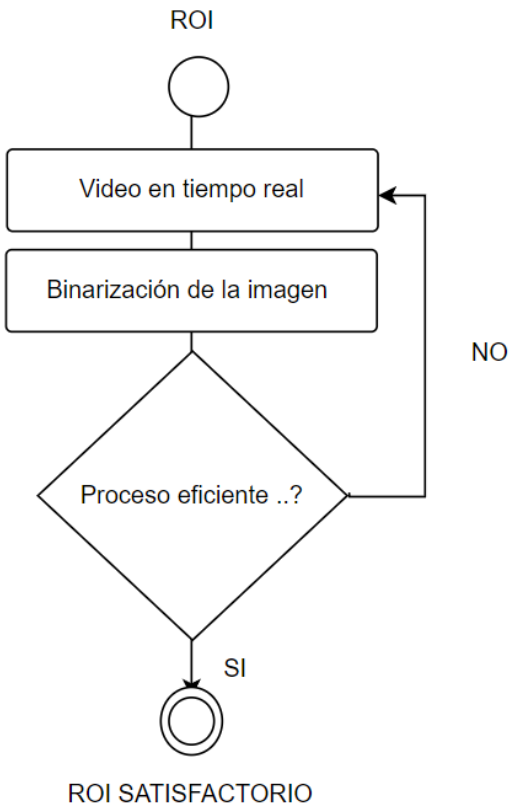
Fuente: Elaboración Propia.

Resta de Fondo

Método usado para detectar objetos, comúnmente detecta aquellos que están en movimiento en cámaras estáticas, su funcionamiento es al restar los objetos del segundo plano o fondo (objetos en movimiento) del primer plano. Esto genera una imagen binaria (de blancos y grises) donde los píxeles clasificados como fondo presentan un valor de gris igual a 0 y los píxeles de las plantas están entre 1 y 255. En la Figura 15 se muestra el proceso para localizar los objetos de interés en las imágenes en movimiento.

Figura 15

Proceso de resta de fondo.



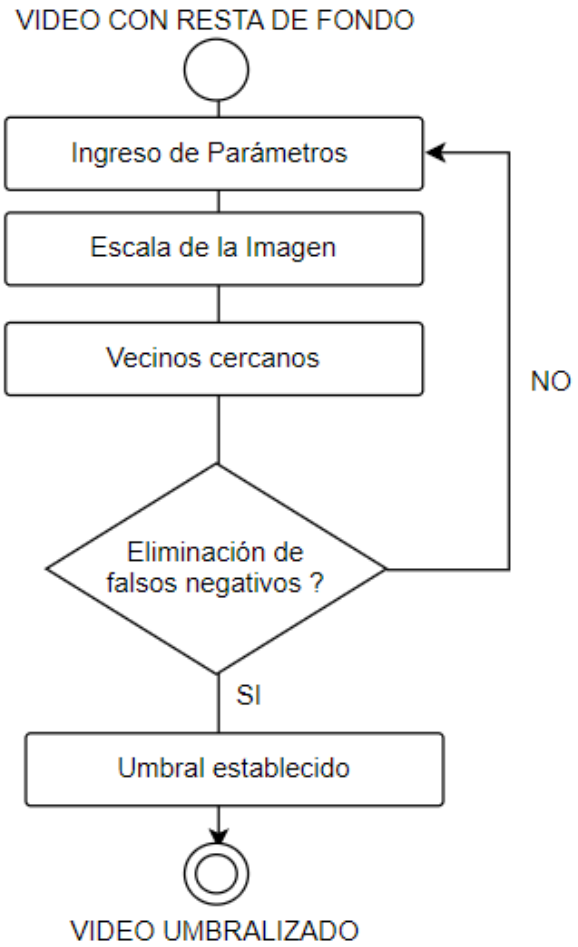
Fuente: Elaboración propia.

Segmentación

La segmentación consta de aplicar filtros al ROI satisfactorio, eliminar los falsos negativos y dejar los verdaderos positivos que se encuentran al comparar los parámetros obtenidos en el clasificador (archivo XML) con la imagen de video que se va transmitiendo en tiempo real, el objetivo final es obtener un valor óptimo del umbral y así poder identificar si es planta o no. La Figura 16 muestra el proceso para encontrar el umbral, es decir, encontrar la distancia mínima en el cual se encontrará otra planta o línea de cultivo con similares características.

Figura 16

Proceso de segmentación.



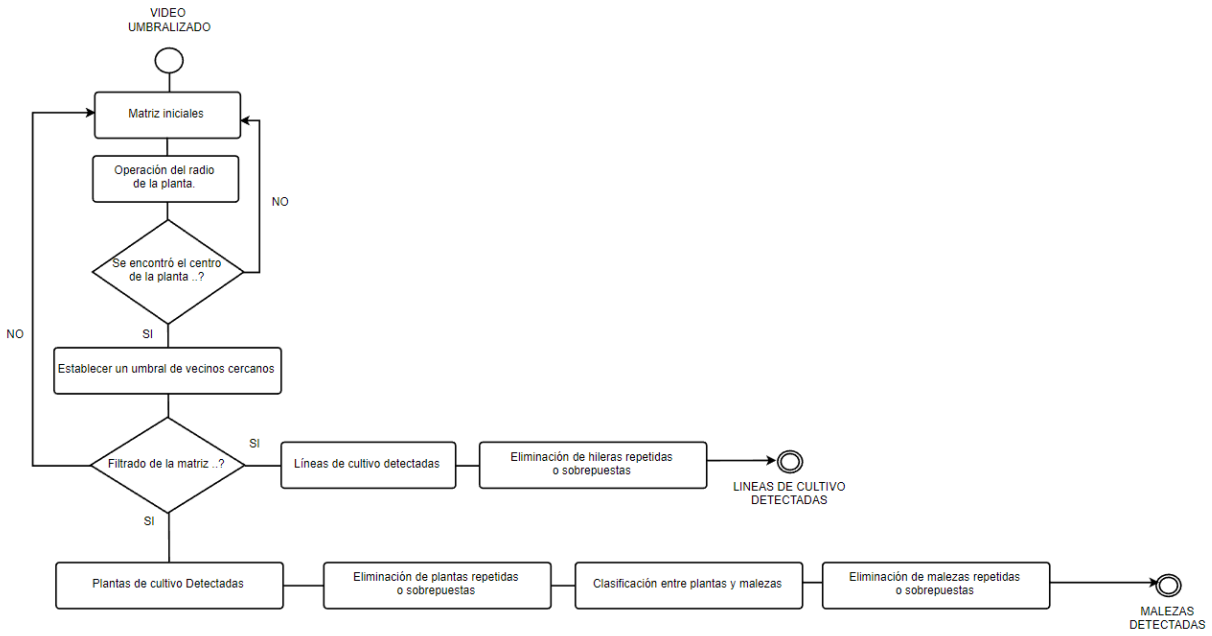
Nota. Elaboración propia.

Detección

En esta etapa se aplica operaciones matemáticas para simplificar las imágenes recibidas en tiempo real, se trata de encontrar el radio de las posibles plantas para realizar una detección a distancias establecidas y tener matrices con las posibles posiciones de las hileras y plantas de cultivo que posteriormente se someten a un filtro de eliminación, conforme al vecino más cercano. Al comprobar nuevamente que las plantas de las matrices filtradas cumplen con los parámetros del clasificador, se tiene dos matrices ya clasificadas con datos confirmados, una como referencia para identificar las líneas de cultivo y otra de las plantas, luego se realiza un descarte donde la matriz de plantas es sometida a un nuevo proceso, estableciendo un que toda característica que no se encuentre a la distancia establecida por el umbral es maleza. La Figura 17 explica detalladamente el proceso realizado, desde, encontrar el centro de la planta a la selección de malezas y líneas de cultivos.

Figura 17

Proceso de detección.



Nota. Elaboración propia

2.3.4. Módulo de Reportes

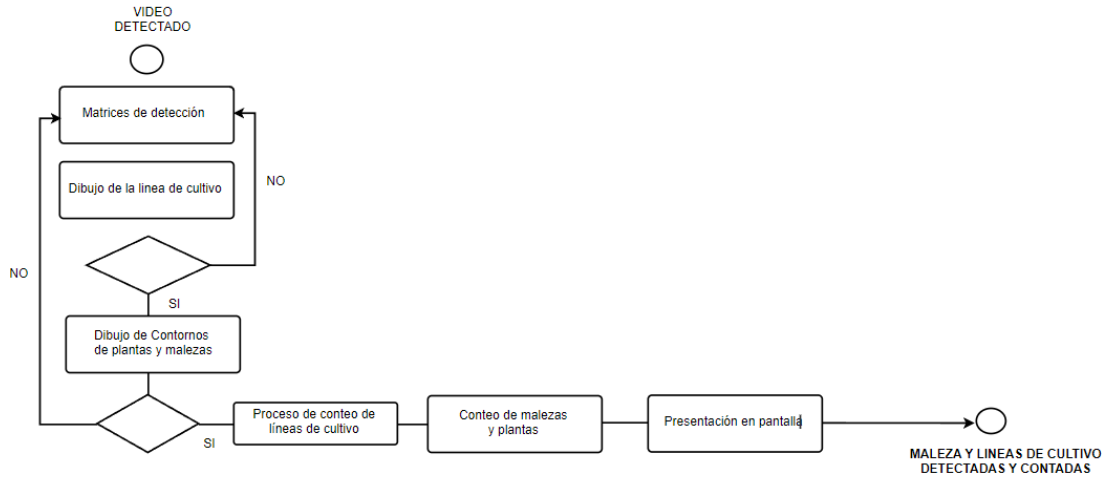
El módulo de reportes consta de dos partes, al tener los objetos detectados, se procede a graficar y contar para una mejor identificación, además se evalúa el software mediante tres métricas establecidas que son, usabilidad, corrección de errores o defectos y productividad. A continuación, el proceso final del algoritmo.

Gráfico y conteo

Los procesos anteriores ayudaron a obtener 3 nuevas matrices, donde: la primera contiene los datos de las líneas de cultivo, la segunda guarda los datos de las plantas en la imagen y la tercera las malezas existentes. Para poder comprender estos datos se presenta por pantalla cada objeto identificado encerrado en una figura geométrica de diferente color, para la identificación de las hileras de cultivo se logra graficar líneas que las visibilice. El conteo consta de una presentación del número de objetos identificados, el cual cambia constantemente conforme las imágenes en tiempo real sean leídas por el software. La Figura 18 explica el proceso de detección y conteo logrando el objetivo del proyecto.

Figura 18

Gráfico y conteo de líneas de cultivo.



Fuente: Elaboración propia.

2.4. Implementación

La arquitectura de software con todos sus módulos concebida en la fase de diseño se ejecuta, en la que se incluye la programación del software, la búsqueda de errores y las pruebas realizadas. La implementación da como resultado un producto de software que se comprueba por primera vez y como producto final en la fase de verificación.

2.4.1. Herramientas

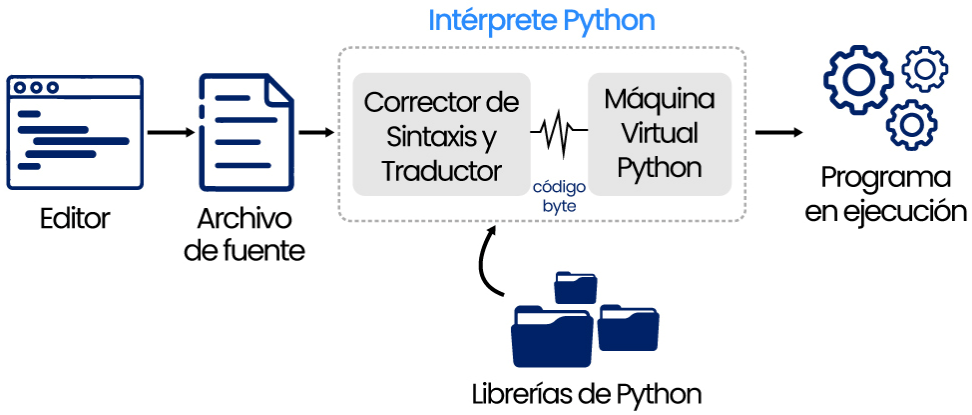
Para el desarrollo de este módulo se optó por las siguientes tecnologías.

Python. – El lenguaje Python fue escogido siguiendo el análisis del aparatado 1.3 del capítulo 1 y dado que su código permite que sea un lenguaje didáctico, versátil y es uno de los líderes en el campo de la inteligencia artificial, la versión 3.10.7 posee librerías de fácil integración compatibles con el sistema operativo Windows 10 Pro, generalmente se agrega Python al PATH para poder ejecutar scripts y programas con más facilidad utilizando el intérprete directamente.

La programación en el lenguaje Python es llamado código de byte, ya que este no puede ser interpretado por una CPU, entonces el intérprete es una máquina virtual de Python (PVM) que compila el código. La Figura 19 presenta el funcionamiento del intérprete de Python, el intérprete procesa poco a poco, alterna la lectura de líneas de código y la realización de cálculos.

Figura 19

Estructura del lenguaje Python



Fuente: Elaboración propia

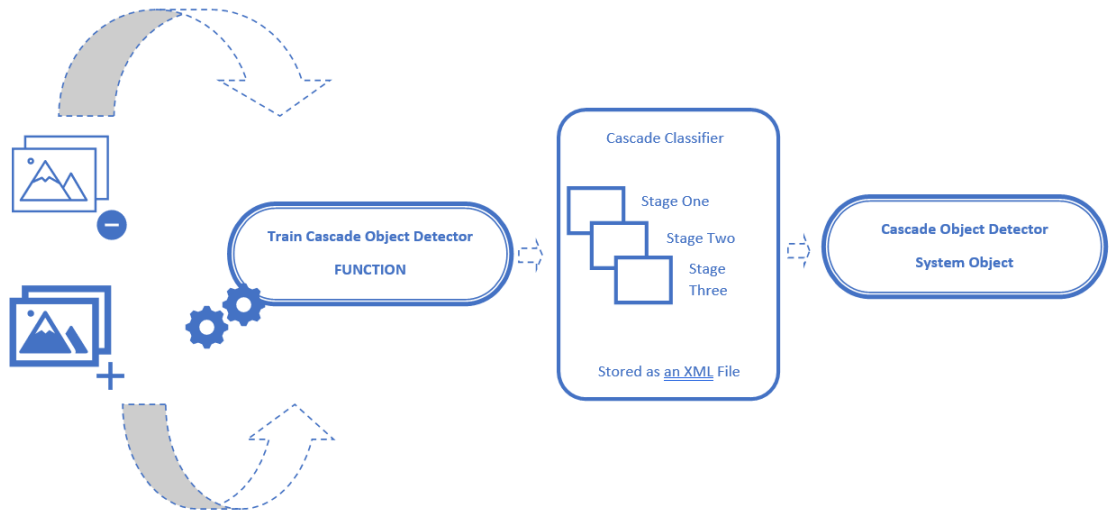
Opencv. – Se empleó las librerías y métodos contenidos en Opencv 3.0, estas librerías están enfocadas al desarrollo de aplicaciones de visión artificial. Opencv se compiló desde cero en la plataforma seleccionada.

Entorno de desarrollo integrado (IDE). – El IDE para el desarrollo de este módulo fue Visual Studio Code en su última versión, el objetivo al usar estas herramientas es que sean las más prácticas y sencillas posibles, debido a los recursos de CPU que se necesitan al momento del entrenamiento del clasificador.

Cascade Trainer GUI. - Cascade Trainer GUI es un programa utilizado para entrenar, probar y mejorar modelos de clasificadores. Este programa utiliza el marco de detección de objetos Viola Jones con un proceso de clasificación basado en Haar cascade, es un algoritmo de detección de objetos basado en el aprendizaje automático. La Figura 20 presenta la estructura del proceso de entrenamiento que realiza Cascade Trainer GUI.

Figura 20

Estructura Cascade Trainer GUI



Nota. Elaboración propia

2.4.2. Adquisición de Imágenes

Se utilizó un banco de 4200 fotografías de cultivos de maíz de la zona norte del país en diferentes etapas de crecimiento, las fotografías fueron tomadas con un enfoque superficial y bajo condiciones naturales de iluminación, con una resolución de 24 Mpx a 5, 10 y 15 metros de distancia.

Como se explicó anteriormente al describir el módulo de adquisición y transmisión de video, las fotografías fueron divididas en 3800 imágenes negativas y 400 positivas, empleadas en el entrenamiento del clasificador, el formato utilizado para almacenar las imágenes es JPG. Este banco de imágenes serán los datos ingresados en el entrenamiento. Para la identificación en tiempo real, la adquisición de imágenes se la puede realizar por diferentes medios, el principal medio que se utilizó es el de una cámara de un teléfono móvil, otros medios son cámara fija o de un dron. Las características de la cámara del teléfono móvil se presentan en la Tabla 11.

Tabla 11

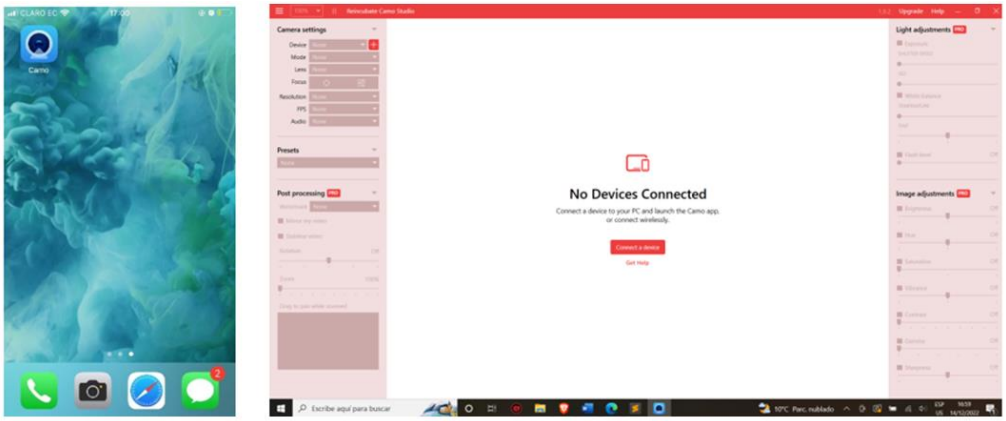
Especificaciones de la Cámara

Marca teléfono móvil	Apple
Modelo	IPhone 8
Sistema Operativo	IOS 16.0
Características de la Cámara	Cámara de 12 Mpx, gran angular apertura de f/1,8 Zoom digital hasta x5 Estabilización óptica de imagen Lente de seis elementos Flash True Tone con LED y sincronización lenta Fotos panorámicas (hasta 63 Mpx) Live photos con estabilización Gama cromática amplia para fotos y live photos HDR automático para fotos Estabilización automática de imagen Modo ráfaga Geoetiquetado de fotos
	Captura de imagen en formato HEIF y JPEG

La transmisión se realiza por “Camo Studio”, esta aplicación fue utilizada porque es una app para móviles y PC con un gran potencial para integrar la AR (realidad aumentada) a sus transmisiones y vídeos, sin la preocupación de ralentizar la máquina, además ofrece una integración fácil y sencilla porque se la debe descargar tanto en el teléfono móvil como en la PC, con un cable USB o escaneando un código QR se conecta la primera vez para reconocimiento de los dispositivos, posteriormente la transmisión en tiempo real mediante una red wifi funciona perfectamente. La Figura 21 muestra capturas de pantalla del funcionamiento de la aplicación de transmisión de imágenes de video en los dos dispositivos.

Figura 21

Funcionamiento de Camo Studio.



(a) *(b)*

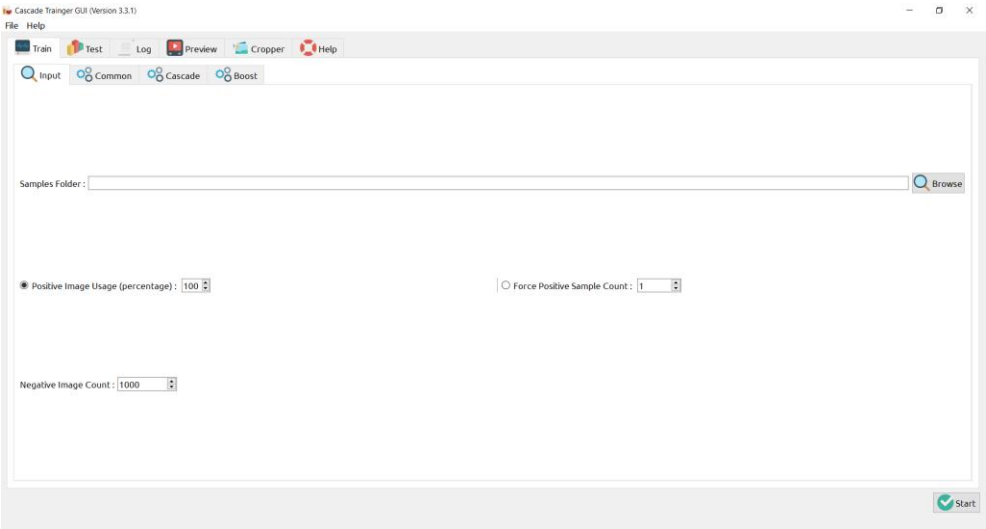
Nota: (a) Captura de pantalla del funcionamiento de Camo en el teléfono móvil. (b) Captura pantalla del funcionamiento de Camo en la PC.

2.4.3. Algoritmo de Entrenamiento

En el uso, la Figura 22 es la captura de pantalla de inicio para entrenar el clasificador, generalmente se necesita proporcionar miles de muestras de imágenes positivas y negativas, pero hay casos en los que se puede lograr una excelente detección con menos muestras (Amin Ahmadi, 2017).

Figura 22

Pantalla de inicio Cascade Trainer GUI.

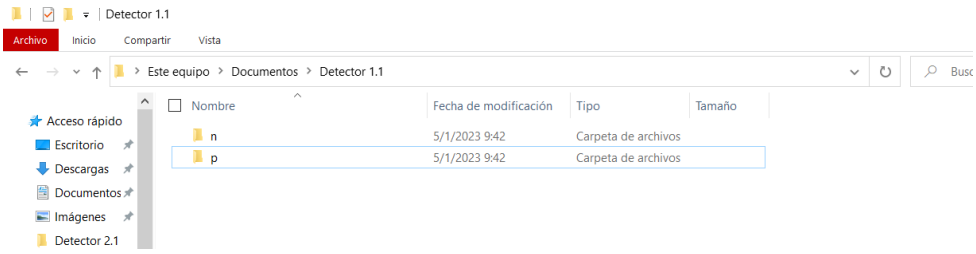


Fuente: Elaboración propia

Se debe crear una carpeta para el clasificador, dos carpetas más dentro de esta, una debe ser “p” (para imágenes positivas) y la otra “n” (para imágenes negativas), las imágenes negativas no deben incluir imágenes positivas ni siquiera parcialmente, la Figura 23 muestra este proceso.

Figura 23

Carpeta principal del clasificador con las subcarpetas "p" y "n".



Nota. Elaboración propia.

En el desarrollo del algoritmo se importan las librerías que se observan en la Figura 24.

Figura 24

Librerías OpenCV

```
import cv2
import numpy as np
import imutils
import os
```

Nota. Código desarrollado en Visual Studio Code.

- **cv2,** llamamos a la biblioteca de código abierto OpenCv, que contiene implementaciones que abarcan más de 2500 algoritmos.
- **numpy,** es una librería en la que se define un tipo de dato que representa matrices multidimensionales.
- **imutils,** conjunto de funciones para el procesamiento de imágenes (cambio de tamaño, clasificación de contornos, rotación, visualización, entre otras).
- **os,** el módulo os nos permite acceder a funcionalidades dependientes del sistema operativo. Sobre todo, aquellas que nos refieren información sobre el entorno de este y permite manipular la estructura de directorios: leer y escribir archivos.

Como se explicó en la Fig. 23, se necesita crear dos carpetas para las imágenes negativas e imágenes positivas, en la Figura 25 se desarrolla las líneas para guardar las imágenes seleccionadas en las caretas requeridas.

Figura 25

Almacenamiento de Imágenes.

```
Datos_p = 'p' #Carpetas p=Plantas
if not os.path.exists(Datos_p):
    print('Carpeta Creada: ', Datos_p)
    os.makedirs(Datos_p)

Datos_n = 'n' #Carpetas n=Malesa
if not os.path.exists(Datos_n):
    print('Carpeta Creada: ', Datos_n)
    os.makedirs(Datos_n)
```

Nota. Código desarrollado en Visual Studio Code.

Para obtener las imágenes positivas y negativas, se tiene las fotografías originales del cultivo, estas son duplicadas para en la copia realizar la selección dibujando un cuadrado, con `resize` redimensionamos el tamaño de la imagen a una calidad manejable. La Figura 26 muestra el código desarrollado para realizar el proceso detallado.

Figura 26

Selección de imagen raíz y copia de esta.

```
imagen=cv2.imread("fotos/s4_5m (2).JPG")    #abre la imagen
#copiamos imagen nueva
#es la imagen donde se va a graficar los recuadros sin alterar la original
imagen_seleccion=imagen.copy()
#redimensiona la imagen que no sea full hd que baje la calidad
imagen_seleccion = imutils.resize(imagen_seleccion, height=1000)
imagen_limpia = imagen_seleccion.copy() #imagen de la que se obtendra la captura

cv2.namedWindow('imagen_seleccion')
cv2.setMouseCallback('imagen_seleccion',select)
```

Nota. Código desarrollado en Visual Studio Code.

El comando `setMouseCallback`, permite al puntero del ratón interactuar en una interfaz gráfica, la interfaz gráfica inicia con `namedWindow`.

En la Figura 27, instanciamos un tamaño del rectángulo a graficar el cual variará según la imagen selecciónada, las variables x1, y1, x2, y2 serán las variables de las coordenadas para graficar el rectángulo que seleccione los objetos en la imagen original y copia. Los contadores simplemente serán el número de imágenes a obtener.

Figura 27

Inicialización de variables

```
t=58 #tamaño de foto a tomar
x1, y1 = 0, 0
x2, y2 = 0, 0

count_p = 0
count_n = 3043
```

Nota. Código desarrollado en Visual Studio Code.

Al momento de graficar el rectángulo la variable x1 será igual a x que es la coordenada del centro en la línea menos t2 y x2 será la suma de x más t2, el mismo proceso para las variables en y en la Figura 28 se logra graficar el rectángulo con los puntos ubicados en las correspondientes esquinas.

Figura 28

Rectángulo Imágenes Positivas

Nota. Selección de una planta de cultivo para el entrenamiento del clasificador

Las imágenes al ser capturadas no deben tener el rectángulo rojo, razón por la cual las variables son restadas menos 1 en la imagen original. En la copia donde, si se deben observar son sumadas 1 para el borde, reducimos la imagen a 38 pixeles, tamaño manejable recomendado por Cascade Trainer GUI y guardamos la imagen con un nombre definido y formato JPG.

Al seleccionar una imagen positiva en un cuadrado, primero detectamos dónde está el cursor, para ello el `select`, entonces establecemos las coordenadas con `global` y con `divmod` buscamos el centro de la selección dividiendo para dos, recordando que estamos trabajando en el eje x y al momento estamos buscando la posición central del rectángulo. El código para la selección de imágenes positivas y el resultado se muestra observa en la Figura 29.

Figura 29

Evento clic Izquierdo (Selección de Imágenes Positivas).

```
#detecta donde esta el cursor y el tamano de los pixeles de la imagen
def select(event,x,y,flags,param):
#establecemos un contador de fotos x1,x2 variables de coordenadas para capturar la foto
    global x1,y1,x2,y2,objeto,count_p,count_n,t
    # del tamano total dividimos para dos para centrar
    t2,resto=divmod(t,2)

    # clic boton izquierdo
    if event == cv2.EVENT_LBUTTONDOWN:

        #tamaño de foto
        #dibujamos el cuadrado con la variable x
        x1=x-t2
        x2=x+t2
        y1=y-t2
        y2=y+t2

        #dibuja rectangulo restando 1 para que no salga el cuadrado al guardar la imagen
        cv2.rectangle(imagen_seleccion,(x1-1,y1-1),(x2+1,y2+1),(0,0,255),1)
        #recorta imagen de x y dimenciones sin bordes y guarda en la variable
        objeto = imagen_limpia[y1:y2,x1:x2]
        #reduce tamaño a escala 38 que es la misma escala seleccionada en el entrenador
        objeto = imutils.resize(objeto, width=38)
        cv2.imshow('objeto', objeto) #muestra foto en otra ventana lo q se capturo

        #guarda la imagen
        cv2.imwrite(Datos_p+'/objeto_{}.jpg'.format(count_p),objeto) #nombre y formato
        print('Imagen P almacenada: ','objeto_{}.jpg'.format(count_p))
        count_p = count_p +1
```

(a) (b)

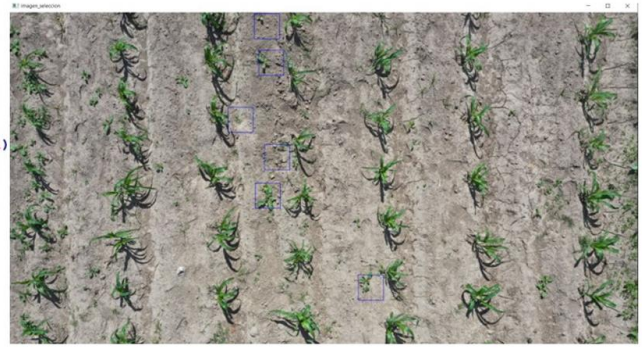
Nota. En la Figura 29 (a) se muestra el código implementado para la selección de las imágenes positivas que contienen las características de las plantas de cultivo. La Figura 30 (b) presenta el resultado, teniendo como ejemplo la selección de las plantas en rectángulos rojos para el entrenamiento del clasificador.

En la Figura 30 el proceso es similar, en este se selecciona las imágenes negativas, que son las características de las malezas o de cualquier objeto no correspondiente a un cultivo a detectar.

Figura 30

Evento clic Derecho (Selección de Imágenes Negativas).

```
if event == cv2.EVENT_RBUTTONDOWN:

    #tamaño de foto
    x1=x
    x2=x+t
    y1=y
    y2=y+t
    #dibuja rectangulo
    cv2.rectangle(imagen_seleccion,(x1-1,y1-1),(x2+1,y2+1),(255,0,0),1)
    #recorta imagen de x y dimenciones
    objeto = imagen_limpia[y1:y2,x1:x2]
    objeto = imutils.resize(objeto, width=38) #reduce tamaño a escala
    cv2.imshow('objeto', objeto) #muestra foto en otra ventana

    #guarda la imagen
    cv2.imwrite(Datos_n+'/objeto_{}.jpg'.format(count_n),objeto)
    print('Imagen N almacenada: ','objeto_{}.jpg'.format(count_n))
    count_n = count_n +1
```



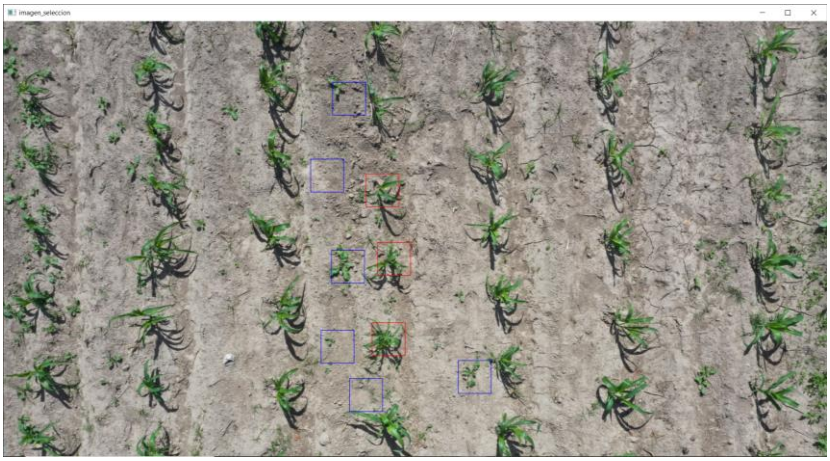
(a) (b)

Nota. En la Figura 30 (a) se encuentra el código implementado y en la Figura 31 (b) está el resultado, teniendo como ejemplo la selección en un rectángulo azul, de varios puntos donde no se mira nada o cuyos verdes no se encuentran en la línea de cultivo para definir características de las malezas.

El resultado final de la selección de imágenes positivas y negativas, concluyendo con un cuadrado rojo para imágenes positivas y cuadrado azul para imágenes negativas presenta la Figura 31.

Figura 31

Obtención de imágenes positivas y Negativas.

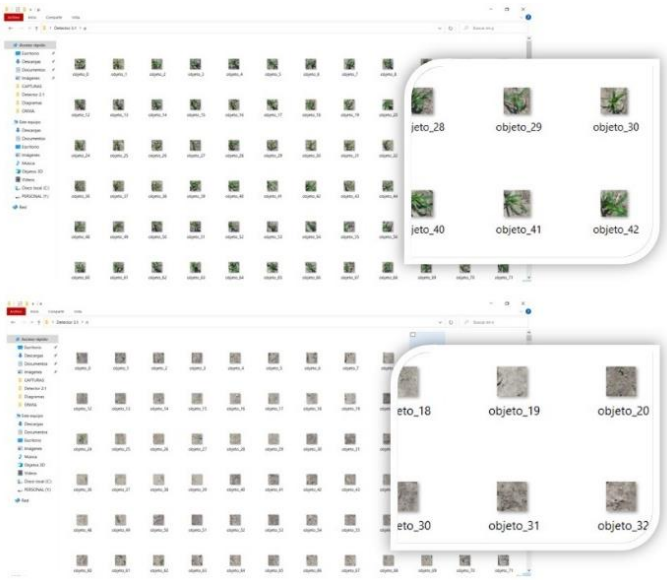


Nota. Selección de Imágenes positivas y negativas.

Al tener ya un banco de imágenes positivas y negativas, continua el proceso donde se ejecuta Cascade Trainer GUI, en la interfaz, seleccionamos la carpeta que se ha creado para el clasificador, esta contiene las imágenes positivas y negativas, se establece un porcentaje de imágenes positivas que se tiene y el número de imágenes negativas obtenidas. La Figura 32 contiene el banco de imágenes, las cuales se encuentran almacenadas en las dos carpetas creadas.

Figura 32

Banco de imágenes positivas y negativas.

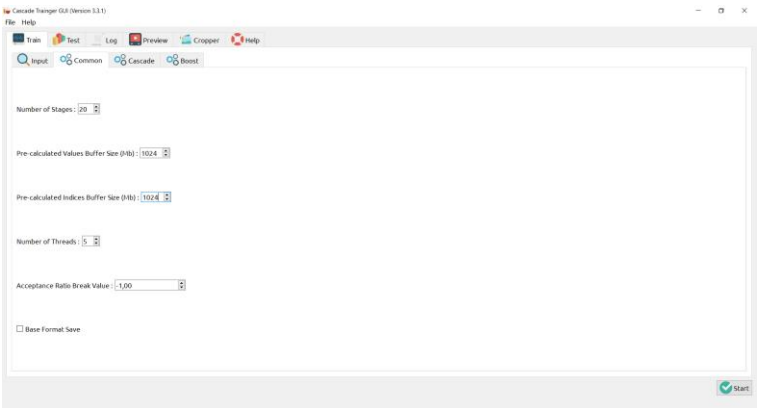


Nota. Imágenes positivas y negativas almacenadas en la carpeta correspondiente.

La Figura 33 muestra la opción de establecer el tamaño del búfer (espacio de memoria, en el que se almacenan datos de manera temporal) previo al cálculo para ayudar con la velocidad del proceso de entrenamiento. Al asignar tanta memoria como se pueda, si se tiene 8 GB de RAM en el computador, se puede establecer de manera segura los dos tamaños de búfer es decir 2048.

Figura 33

Búfer o espacio de memoria a utilizar en el entrenamiento.

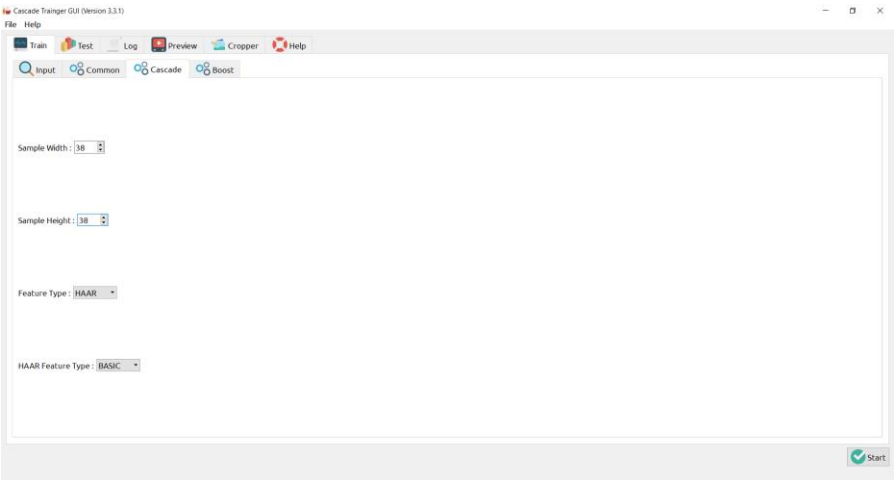


Nota. Captura de Pantalla de la Configuración de Búfer.

Se debe establecer el ancho y la altura de las imágenes, en la Figura 34 configuramos un tamaño ideal a mayor tamaño mayor tiempo de detección, la configuración para el ancho y la altura de la muestra es de 38 x 38 y establecemos el tipo de característica en HAAR o LBP, en este caso se utiliza clasificadores HAAR debido a su mayor precisión, teniendo de inconveniente el tiempo que tarda al entrenar. Finalmente, los parámetros en la pestaña Boost, se recomienda mantener en los valores predeterminados y empezar a entrenar el clasificador.

Figura 34

Tamaño de muestra y tipo de clasificador.

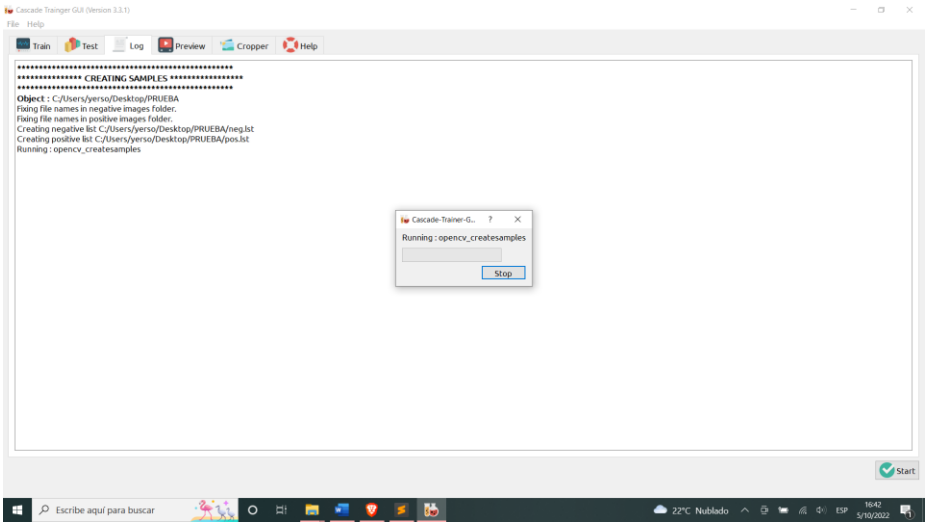


Nota. Captura de Pantalla de la Configuración del tamaño de las imágenes de entrenamiento.

Con todas las configuraciones realizadas y con las suficientes imágenes positivas y negativas obtenidas, se procede al entrenamiento. La Figura 35 muestra el entrenamiento del clasificador el cual tuvo un tiempo de duración aproximado de 15 horas, este debe extraer todas las características de las imágenes negativas y positivas y almacenarlas en una matriz en el Archivo XML.

Figura 35

Proceso de entrenamiento del clasificador.

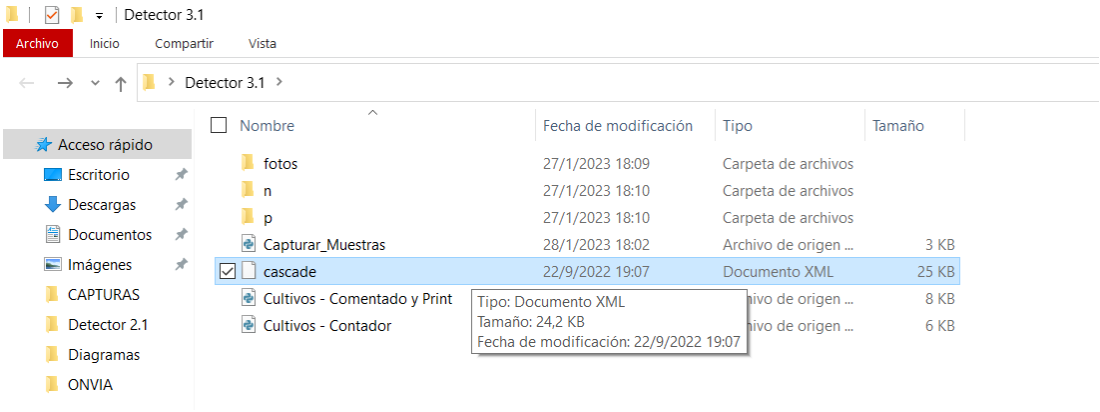


Nota. Captura de Pantalla del Entrenamiento.

Al finalizar el entrenamiento como se presenta en la Figura 36, la carpeta del clasificador tendrá nuevos archivos, la mayoría serán archivos temporales, pero el de interés será el archivo “cascade.xml”, ya que es el clasificador que utilizaremos para la detección.

Figura 36

Archivo “Cascade.xml”.



Nota. Resultado final, archivo XML del proceso de entrenamiento del clasificador.

2.4.4. Implementación de Algoritmo de detección

A continuación, se presenta una descripción de la implementación del algoritmo siguiendo las diferentes etapas previstas. La captura de video se realizó con la clase `VideoCapture` de Opencv, la clase tiene varias opciones de recepción de video, puede ser pregrabado o en tiempo real producido por una cámara IP, en este proyecto se seleccionó la segunda opción, la línea de código se observa en la Figura 37 instanciada en un archivo .py estos archivos corresponden a un archivo de scripts de Python.

Figura 37

Captura de video.

```
cap = cv2.VideoCapture(0,cv2.CAP_DSHOW)
```

Nota. Función de Python que permite la captura de imágenes de video.

ROI (Visualización de video y Área de Interés)

La clase `VideoCapture` busca directamente alguna cámara conectada a la PC para conseguir video en tiempo real, al utilizar la aplicación “Camo” permite conectar una cámara externa al computador. La transmisión comienza al abrir la aplicación en el dispositivo móvil que tiene la cámara, el dispositivo se conecta a la aplicación descargada en el PC y la transmisión inicia, al ejecutar el software en Python la función `VideoCapture` obtiene las imágenes transmitidas por la app. El archivo “cascade.xml” es el resultante del entrenamiento, es el clasificador requerido para realizar la detección del área de interés.

La Figura 38 muestra cómo este archivo se carga mediante la clase `CascadeClassifier`, clase utilizada para detectar objetos mediante un clasificador.

Figura 38

Importación del archivo XML

```
plantaClassif = cv2.CascadeClassifier('cascade.xml')

minT=55
maxT=65
```

Nota. Función de Python que permite la importación del archivo XML al software.

Las variables minT y maxT definen el tamaño máximo y mínimo de lo que vamos a detectar, es decir, si se va a detectar una planta a mayor distancia estos valores serán mínimos y si se va a detectar la misma planta a una menor distancia estos valores aumentan porque el tamaño de la planta aumenta, con esto se controla el ROI a definir sobre la imagen de video transmitida.

Resta de Fondo

Para la resta de fondo se realiza un bucle donde:

- **Read (). -** Hace una lectura de las imágenes en movimiento transmitidas en tiempo real.
- **ctvColor ().** - Transforma en una imagen binaria la imagen capturada por la cámara, se establece un fondo de color negro y grises a los objetos en movimiento.
- **shape. –** Se obtiene la dimensión del video que ingresa: ancho, altura y canal. El canal es el numero de componentes utilizados para representar la imagen, estos pueden ser: de color, espectro visible, imágenes multiespectrales e hiperespectrales. Existen tres tipo de canales principales: RGB (rojo, azul y verde), CMYK (cian, magenta, amarillo y negro), HSV (un canal dedicado al brillo).

La Figura 39 presenta el código implementado donde se realiza un bucle infinito el cual se repite para cada frame, en el actual proyecto se está trabajando a 30 fps, además se convierte las imágenes a escala de Grises.

Figura 39

Imagen a Gris.

```
#BUCLE DEL PROGRAMA
while True:
    #SE REALIZA LA LECTURA DE LA CAMARA
    ret,frame = cap.read()
    #SE PASA A ESCALA DE GRISES LA IMAGEN CAPTURADA POR LA CAMARA
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

Nota. Código implementado en Visual Studio Code.

La figura 40 contiene una captura de las imágenes de video transmitidas en tiempo real a escala de grises o también llamado binarización de la imagen, este proceso ayuda a preservar las características esenciales de la imagen.

Figura 40

Visualización de la resta de fondo

Nota. Imagen binaria de la ejecución del software.

Segmentación

Con la resta de fondo ya realizada, utilizamos la función `detectMultiScale`, la cual ayuda a detectar los objetos de acuerdo con el clasificador que estamos utilizando, delimita en un rectángulo el objeto encontrado, pero para ello se debe definir los siguientes parametros:

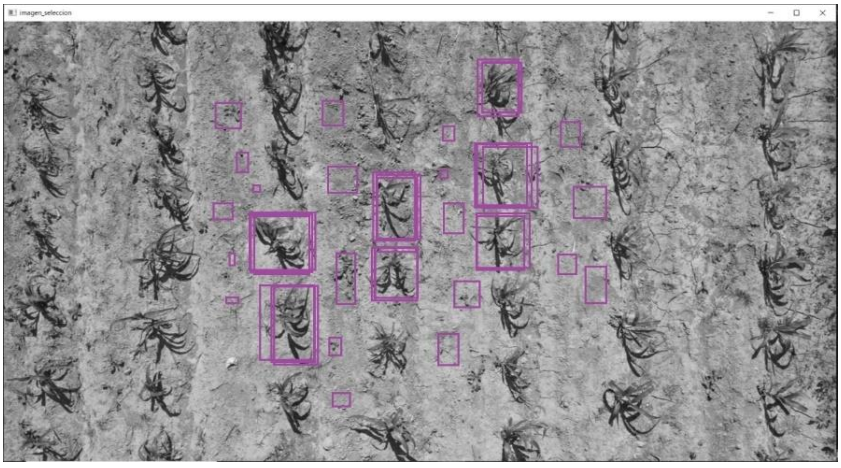
- **gray**, es la imagen de video en tipo real entrante ya convertida a escala de grises, en este caso la variable se llama gray.

- **Scalefactor,** es el factor de escala, indica que tanto va a ser reducida la imagen, trabaja conforme a una pirámide donde los valores van de 1,01 a 2. Para este proyecto ingresamos escalas menores acordes a lo que se quiere detectar, ya que las plantas son pequeños píxeles en una imagen que están cercanos unos con otros. Las líneas de cultivo tienen mayor escala, no están cercanas y no existen en gran número por lo cual la detección no necesita ser tan precisa. Básicamente lo que se busca es ayudar a que las características del clasificador en tamaño coincidan con los objetos a detectar en la imagen.

- **minNeigbors,** el clasificador en cascada Haar funciona con un enfoque de ventana deslizante, quiere decir, desliza una ventana a través de la imagen para cambiar su tamaño y busca nuevamente hasta que no pueda cambiar su tamaño aún más, con cada interacción del clasificador se almacenan más salidas (más rectángulos), cuando esta ventana se desliza en una imagen redimensionada y deslizada nuevamente; en realidad detecta muchos falsos positivos. La Figura 41 muestra un ejemplo con los vecinos cercanos igual a 0.

Figura 41

Funcionamiento de minNeighbors con el deslizamiento en cascada.

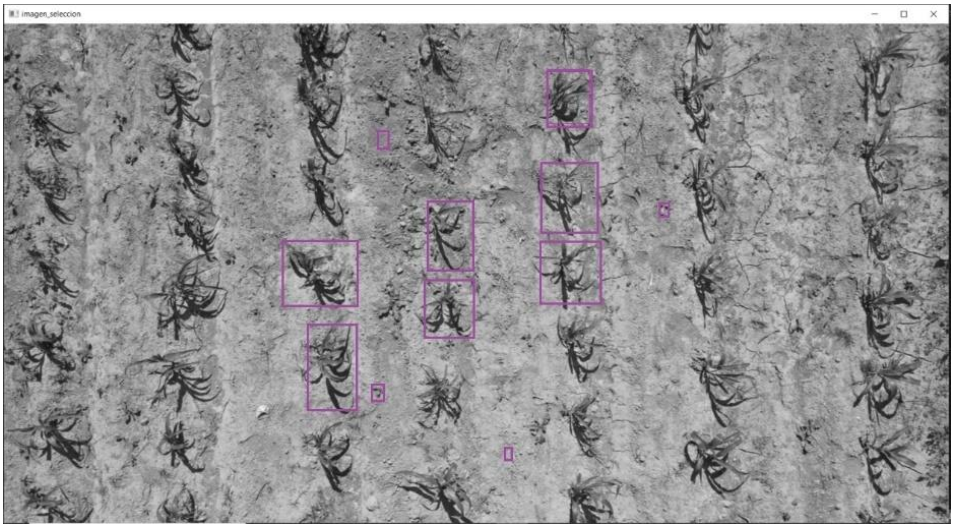


Nota. Ejemplo con minNeighbors = 0

En una explicación más clara, existe demasiada detección de plantas debido al cambio de tamaño de la ventana deslizante y muchos falsos positivos, para eliminar los falsos positivos y sacar los rectángulos sobrantes de las detecciones de las plantas, se aplica el enfoque del minNeighbors, este enfoque asume que un rectángulo está cerca de otro, lo cual en la Figura 42 es correcto, y los aleja. Al aumentar el número de minNeighbors permite eliminar falsos positivos, pero hay que tener cuidado, al aumentarlo, también puede perder verdaderos positivos.

Figura 42

Resultado del proceso: vecinos más cercanos.



Nota. Ejemplo con minNeighbors = 25

La codigo detrás de estos parámetros muestra la Figura 43, donde, el detector se ejecutará en un estilo de escala múltiple y de ventana deslizante, dará múltiples respuestas incluso para una región de una sola planta, para ello se establece un umbral de límite inferior, es decir, solo se contará como una planta válida si el número de respuestas para esta es mayor que el valor establecido en `minNeighbors`.

Figura 43

Proceso de segmentación.

```
he, we, ch = frame.shape #SE TOMA LOS PARAMETROS DE LA IMAGEN CAPTURADA ALTURA, ANCHO, CANAL

#PARAMETROS DE DETECCIÓN PARA IDENTIFICAR CADA PLANTA USANDO EL ARCHIVO DE ENTRENAMIENTO
plantas = plantaClassif.detectMultiScale(gray, scaleFactor = 1.5, minNeighbors = 35, minSize=(minT,minT), maxSize=(maxT,maxT))
maleza=plantas #SE DUPLICA LOS DATOS EN UNA NUEVA VARIABLE

#PARAMETROS DE DETECCIÓN PARA IDENTIFICAR CADA GUACHO USANDO EL ARCHIVO DE ENTRENAMIENTO
lcultivos = plantaClassif.detectMultiScale(gray, scaleFactor = 2, minNeighbors = 150, minSize=(minT,minT)) #, maxSize=(maxT,maxT))
```

Nota. Código implementado en Visual Studio Code.

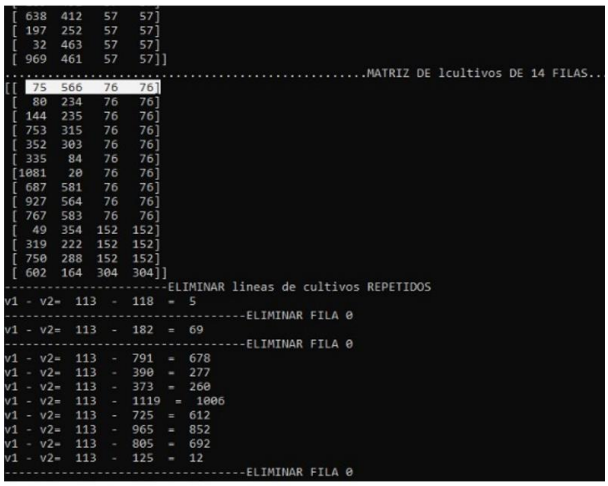
Detección

En esta fase inicializamos 3 matrices para guardar los datos: líneas de cultivo, plantas y malezas. Estas matrices almacenarán las posiciones de los objetos detectados. También sé instancia 3 variables más, las cuales obtendrán el número de líneas de cultivo, malezas y plantas conforme la transmisión de las imágenes de video vaya avanzando.

La Figura 44 muestra una matriz la cual esta compuesta por todas las posibles plantas de cultivo detectadas en base al clasificador, esta matriz contiene 4 columnas, la primera corresponde la posición, la tercera y cuarta al ancho y altura del objeto detectado. Estos son los datos iniciales generados con la función `detectMultiScale` y los parámetros configurados, a continuación, se realizan operaciones matemáticas para el filtrado de los datos.

Figura 44

Matriz almacenada de las posibles líneas de cultivo.



Nota. Código desarrollado en Visual Studio Code.

En la figura 45 se realiza un for externo para i donde el rango de interacción será sobre el número de líneas de cultivo de 0 a el total detectadas menos 2, el numero de líneas de cultivo se asume a cada fila de la matriz inicial de la Figura 44.

El siguiente paso es encontrar el centro o radio de la planta detectada, en la matriz se tiene la altura y el ancho del cuadrado que identifica el área de la planta por lo que realizamos una división de un lado para dos, es decir obtenemos en `cx1` la mitad del ancho de la planta para luego sumar la posición (columna 0 de la matriz) y así almacenar el centro de la planta en `v1`. Hacemos lo mismo con la siguiente fila de la matriz para j, al tener dos centros de unas posibles plantas de cultivo establecemos un umbral que será la distancia del centro de vecino mas cercano que detectara otra planta y comparamos: si la resta de el centro de una planta con el centro de otra planta es menor o igual al umbral establecido esos datos no corresponden a una planta y la fila de la matriz se elimina.

- Con la función `divmod` seleccionamos la columna 1 y dividimos para 2, el resultado sería la mitad del lado, este cálculo lo almacenamos en la variable `cx1`.
- La función `abs` convierte en un valor absoluto cualquier resultado de la resta

Siguiendo una detección del proyecto, asumimos que una fila pertenece a una planta la cual esta encerrada en un cuadrado mide 76 x 76 (Figura 44), buscamos su centro, para ello, dividimos el número del lado en la posición x, sumamos el valor del vecino cercano, en este caso es 75, el valor del centro de la planta en la posición i será 113, realizamos el mismo calculo para j y el resultado es 125 con el área de la matriz 152 x 152 (Figura 44) entonces la resta 125 – 113 resulta de 12, un valor menor al umbral de 150, razón por lo que estos datos no pertenecen a una planta y se eliminan de la matriz.

Figura 45

Detección de líneas de cultivo

```
for i in range(nlcultivos-1):
    cx1,resto=divmod(lcultivos[i,2],2)
    v1=lcultivos[i,0]+cx1
    for j in range(i,nlcultivos-1):
        j+=1
        cx2,resto=divmod(lcultivos[j,2],2)
        v2=lcultivos[j,0]+cx2
        T=abs(v1-v2) #DISTANCIA DE ENTRE CENTROS DE LCULTIVOS
        if T<=150: #RANGO EN EL QUE NO DEBE DETECTAR OTRA L. CULTIVO
            flcultivos.append(i) #GUARDA EN LA LISTA LA FILA QUE ESTA CERCA DE OTRA L. CULTIVO
    i+=1
```

Nota. Código para encontrar el centro de la planta.

La Figura 46, muestra el proceso de escritorio donde encontramos el radio de las plantas poniendo un umbral en el que no se debe detectar otro centro, en este caso, menor o igual a 150, este valor fue tomado basándose en prueba y error del sistema.

Figura 46

Proceso de búsqueda del radio de la planta.

Nota. Análisis de escritorio que se sigue para encontrar el centro de la planta

Con un rango de vecinos cercanos establecido, los falsos negativos se eliminan inmediatamente. La Figura 47 presenta una matriz depurada, ya no con posibles coordenadas de plantas pertenecientes a una línea de cultivo, sino, con coordenadas de una planta perteneciente a cada hilera, lo que permitirá graficar líneas de cultivo.

Figura 47

Matriz sin falsos negativos de líneas de cultivo.

Nota. Matriz observada mediante la ejecución en consola del software.

Al tener una matriz con datos de las hieleras detectadas, estas se pueden repetir o sobreponer por lo que realizamos un filtrado. En la Figura 48 se realiza lo siguiente:

- La función `set` elimina automáticamente los valores repetidos de la lista.
- Con `np.delete` hacemos una actualización de la matriz inicial `lcultivos` donde eliminamos todos los datos de plantas que no consten en la matriz que obtuvimos anteriormente.

Figura 48

Filtrado y eliminación de líneas de cultivo repetidas y sobrepuestas.

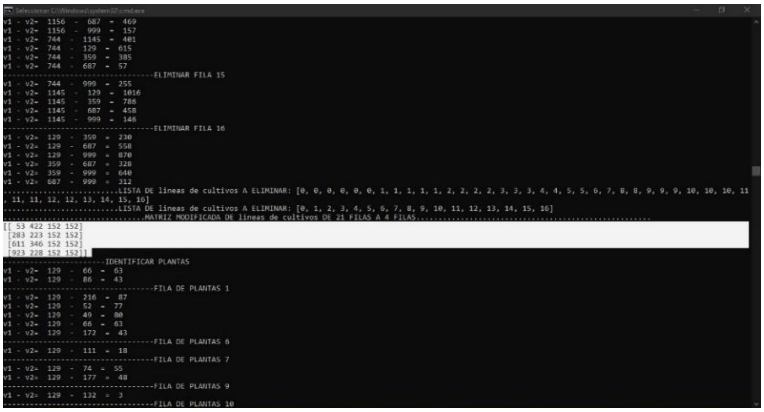
```
flcultivos=list(set(flcultivos)) #ELIMINA NUMEROS REPETIDOS EN LA LISTA

if flcultivos: #VERIFICA SI LA LISTA TIENE ELEMENTOS
    lcultivos=np.delete(lcultivos, flcultivos,axis=0) #ELIMINA LAS FILAS DE LOS lcultivos CERCANOS

nlcultivos_D=len(lcultivos) #NUMERO DE lcultivos RESTANTES
```

Nota. Código siguiente al código de la Figura 47.

En la Figura 49 se muestra una matriz depurada con los datos de cada línea de cultivo detectada en tiempo real, esta matriz es encontrada en cuestión de segundos y al seguir la imagen irá cambiando.

Figura 49

Matriz final con datos de líneas de cultivo detectadas.



Nota. Matriz filtrada y depurada con las posiciones de cada línea de cultivo a graficar.

Para la identificación de la maleza, el punto inicial es la matriz de la Figura 49, es decir, la matriz final con los datos de las líneas de cultivo detectas, a esta se establece un nuevo rango de vecinos cercanos teniendo en cuenta que el valor del umbral será mucho menor al valor del umbral para detección de líneas de cultivo porque las plantas se encuentran más cercanas unas con otras en la misma hilera.

En la Figura 50 se presenta el método completo implementado para la detección de malezas, con la observación que es muy similar al método para detección de líneas de cultivo.

Figura 50

Proceso de detección para malezas.

```
#IDENTIFICAR QUE PLANTA ES MALEZA
for i in range(nlcultivos_D):
    cxg,resto=divmod(lcultivos[i,2],2)
    v1=lcultivos[i,0]+cxg
    for j in range(nplantas):
        cxp,resto=divmod(plantas[j,2],2)
        v2=plantas[j,0]+cxp
        T=abs(v1-v2) #DISTANCIA DE ENTRE CENTROS DE lcultivos CON LAS PLANTAS
        if T<=50: #RANGO EN EL QUE DETECTA SOLO PLANTAS Y LO QUE ESTE FUERA DEL RANGO ES MALEZA
            fplantas.append(j)
        j+=1
    i+=1

fplantas=list(set(fplantas)) #ELIMINA NUMEROS REPETIDOS DE LA LISTA
```

Nota. Código desarrollado en Visual Studio Code.

El método implementado busca el centro de las plantas de la matriz de líneas de cultivo conforme al umbra, los identifica y compara con la matriz inicial que es la matriz obtenida con el clasificador, a la matriz inicial se elimina los datos de las plantas que estén repetidas y sobrepuestas actualizando la matriz inicial con las posiciones de todas las plantas detectadas pertenecientes a una línea de cultivo. La Figura 51 presenta los datos de las plantas detectadas.

Figura 51

Matriz de plantas detectadas en la imagen.

Nota. Matriz filtrada y depurada con todas las plantas detectadas en la imagen.

Para clasificar que es planta y que es maleza se utiliza el umbral establecido para la detección de plantas, donde, si es mayor a este se identifica como maleza, es decir si es menor o igual es planta y si es mayor es maleza, las malezas detectadas son almacenadas en una nueva matriz `fmaleza`. Luego se realiza un filtrado donde se verifica que los datos de la matriz de malezas no se repitan en la matriz de plantas, para luego eliminar los datos de malezas y plantas repetidas y sobrepuestas en cada lista. La Figura 52 implementa el código de filtrado de malezas y a continuación se obtiene un numero de malezas y numero de plantas.

Figura 52

Existencia y filtrado de plantas y malezas

```
for x in range(nplantas): #DENTRO DE LA MATRIZ DE PLANTAS
    if not x in fplantas: #VERIFICAMOS QUE FILAS NO SON PLANTAS Y SON MALEZA
        fmaleza.append(x) #SE GUARDA LAS FILAS DE LA MALEZA

if fplantas and fmaleza: #VERIFICA SI LAS LISTAS TIENEN ELEMENTOS
    plantas=np.delete(plantas, fmaleza,axis=0) #ELIMINA LAS FILAS DE LA MALEZA
    maleza=np.delete(maleza, fplantas,axis=0) #ELIMINA LAS FILAS DE LAS PLANTAS

nmaleza_D=len(maleza) #numero de malezas
nplantas_D=len(plantas) #numero de plantas
```

Nota. Código implementado en Visual Studio Code.

Gráfico y conteo

El proceso final es identificar las líneas de cultivo, malezas y plantas, las líneas de cultivo se las identificará por una línea amarilla, las malezas estarán encerradas en un círculo rojo y las plantas en un círculo verde. La Figura 54 contiene los 3 métodos para graficar los 3 elementos detectados, las malezas y plantas se las encierra en un círculo con la función `circle`, aquí sé instancia los parámetros y el color a graficar, para las líneas de cultivo se las grafica con la función `line`.

Figura 53

Gráfico de líneas de cultivo, malezas y plantas.

```
#DESPUES DE IDENTIFICAR SE GRAFICA UN CIRCULO EN LA PLANTA
for (xp,yp,w,h) in plantas:# CIRCULO AZUL
    cx,resto=divmod(w,2) #SE DIVIDE EL TAMAÑO DE LA IMAGEN DETECTADA PARA 2
    cy,resto=divmod(h,2) #SE DIVIDE EL TAMAÑO DE LA IMAGEN DETECTADA PARA 2

    # cv2.circle(frame, (xp+cx, yp+cy), cx, (255,0,0), 2) #GRAFICA CIRCULO AZUL

#DESPUES DE IDENTIFICAR SE GRAFICA UN CIRCULO EN LA MALEZA
for (xm,ym,w,h) in maleza:# CIRCULO ROJO
    cx,resto=divmod(w,2) #SE DIVIDE EL TAMAÑO DE LA IMAGEN DETECTADA PARA 2
    cy,resto=divmod(h,2) #SE DIVIDE EL TAMAÑO DE LA IMAGEN DETECTADA PARA 2

    cv2.circle(frame, (xm+cx, ym+cy), cx, (0,0,255), 2) #GRAFICA CIRCULO ROJO

#DESPUES DE IDENTIFICAR SE GRAFICA UNA LINEA EN LA LINEA DE CULTIVO
for (xg,yg,w,h) in lcultivos: #LINEA AMARILLA
    cx,resto=divmod(w,2) #SE DIVIDE EL TAMAÑO DE LA IMAGEN DETECTADA PARA 2

    cv2.line(frame, (xg+cx,5),(xg+cx,he-5),(0,255,255),5) #GRAFICA LINEA AMARILLA
```

Nota. Código implementado en Visual Studio Code.

El conteo consiste en la impresión en pantalla de un texto con el número de cada objeto detectado, para ello, se extrae los datos de las 3 matrices que ya se obtuvieron, matriz con las posiciones de una planta perteneciente a una hilera, matriz de plantas y matriz de malezas.

En la Figura 54 se realiza la implementación del código llamando a las 3 matrices para la presentación en pantalla del número de cada objeto detectado.

Figura 54

Conteo de líneas de cultivo, malezas y plantas

```
#TEXTO DE NUMERO DE LINEAS VISIBLES
#CV2.PUTTEXT(imagen en donde saldrá el texto,
#texto a mostrar,posición de inicio del texto,fuente,tamaño,Color de texto,Grosor de fuente)
cv2.putText(frame,'{} lineas de cultivos'.format(nlcultivos_D), (15,25), 2, 1, (0, 255,255), 3)

#TEXTO DE NUMERO DE CIRCULOS VISIBLES PLANTAS Y MALEZA
cv2.putText(frame,'{} plantas'.format(nplantas_D), (15,50), 2, 1, (255, 0, 0), 3)
cv2.putText(frame,'{} maleza'.format(nmaleza_D), (15,75), 2, 1, (0, 0, 255), 3)

cv2.imshow('frame',frame) #MUESTRA EN PANTALLA LAS GRAFICAS REALIZADAS
cv2.imshow('gray',gray)

#TECLA DE SALIDA DEL PROGRAMA
if cv2.waitKey(1) == ord('q'):
    break
```

Nota. Código implementado en Visual Studio Code.

CAPÍTULO 3

Resultados

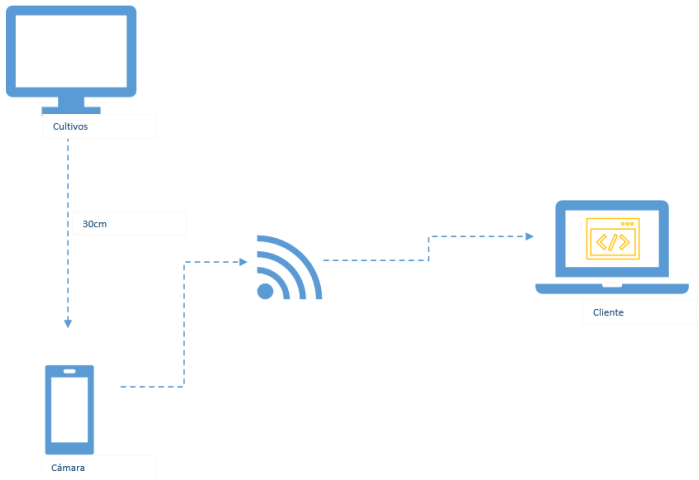
Al evaluar el rendimiento del algoritmo se realiza un seguimiento a un cultivo por cuatro semanas, en este lapso la planta crece y adopta diferentes características, lo que permite establecer medidas cuantitativas de precisión y rendimiento del software conforme a los objetivos del proyecto, el capítulo es elaborado en base a la fase de verificación de software, correspondiente a la metodología en cascada, esta etapa será la final para el proyecto ya que la metodología nos habla de una más que es mantenimiento, pero esta se tendrá en cuenta para trabajos futuros.

3.1. Verificación del Software

Para la verificación del software se utilizó un ambiente controlado, donde el usuario tenga acceso a una red wifi y pueda vincularse con el software que se está ejecutando en una computadora portátil, del banco de fotografías de cultivos se seleccionó las que cumplen unas métricas cuantitativas establecidas y se la proyecto en un monitor, la captura en tiempo real se la realizó con el teléfono móvil al frente del monitor a una distancia de 30 cm, como una cámara fija de transmisión. En la Figura 55 se muestra la estructura del ambiente de pruebas.

Figura 55

Escenario de pruebas.



La Tabla 12 detalla las características de hardware y software de los dispositivos utilizados en el escenario de pruebas además de la cámara cuyas características se encuentran descritas en la anterior Tabla 11.

Tabla 12

Características de los dispositivos a usar.

Dispositivos	Características	
	Hardware	**Software**
Computador	Marca: Dell Procesador: Core I7 Memoria RAM: 8GB	Sistema Operativo: Windows 10
Monitor	Marca: LG Pantalla: 23.8 pulgadas Resolución: Full HD (1920x1080) Tipo de panel: IPS	HDR: Si Frecuencia de actualización: 144Hz

Al ser un software que analiza datos en tiempo real y que se lo dividió en módulos, se realizarán pruebas de integración de cada uno para ver su funcionamiento, los módulos descritos en el proyecto son: (1) Módulo de adquisición y Transmisión, (2) Módulo de recepción y procesamiento, (3) Módulo de Reportes.

3.1.1. Casos de prueba

Una prueba unitaria verifica una parte de código, estas pruebas aseguran que se cumpla con su determinada función antes de ser integrada con los demás, por lo que cada módulo debe aprobar todos los casos de prueba establecidos. Los casos de pruebas ayudan a medir la funcionalidad del software a través de parámetros para obtener los resultados esperados.

La Tabla 13 describe los casos de prueba para la verificación de integración de los módulos del software, en cada módulo se comprueba la transmisión de imágenes mediante los dispositivos y la precisión de la detección del algoritmo. En los casos de prueba donde existen problemas, se describe la manera óptima en que se solucionó.

Tabla 13

Casos de prueba de la integración del software

No.	Módulo	Objetivo para cumplir	Prerrequisitos
CP001	1-2	Prueba de transmisión de imágenes en tiempo real por parte de la cámara del móvil y la recepción en la PC.	Tener configurado y listo el escenario de prueba.
CP002	2-3	Procesamiento del video recibido identificando las líneas de cultivo y malezas.	Tener ejecutado el software con las imágenes de cultivos a evaluar.
CP003	1-2-3	Verificar la detección y el conteo de las líneas de cultivo y malezas detectadas.	Tener ejecutando el software.

CP001 – Transmisión en tiempo real

Paso 1.- Comprobar el funcionamiento de la cámara y la transmisión en tiempo real, el funcionamiento del streaming depende la aplicación “camo”, se empleó la aplicación descargada en los dos dispositivos: en el teléfono, que es el dispositivo de la cámara y en la PC, donde el software recepta las imágenes. La Figura 56 es una captura de pantalla de las imágenes en transmisión de un cultivo, aquí inicia la transmisión de las imágenes mediante una red wifi.

Figura 56

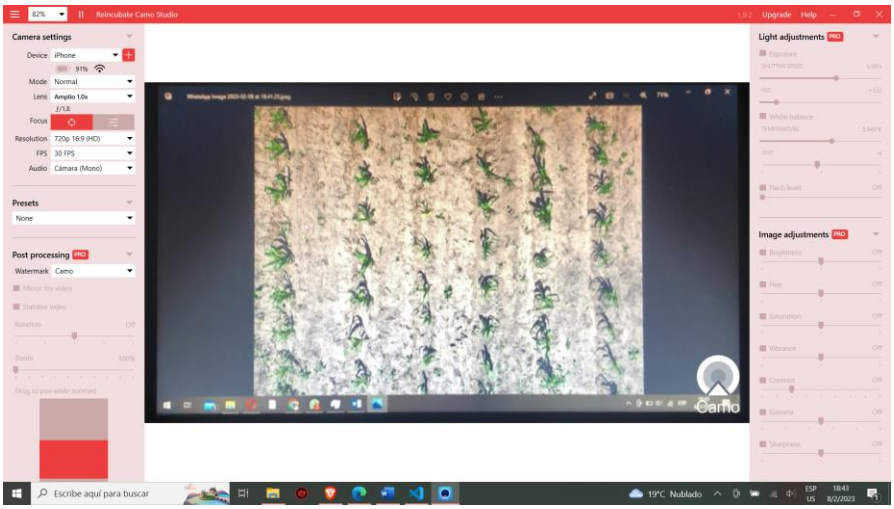
Captura de las Imágenes por la cámara de video.

Nota. Imágenes en tiempo real capturadas por la cámara del iPhone 8

La Figura 57 muestra la captura de pantalla de la transmisión recibida en la aplicación de escritorio “camo” que se estará ejecutando durante todo el proceso de detección

Figura 57

Recepción de las imágenes en la aplicación “Camo”.

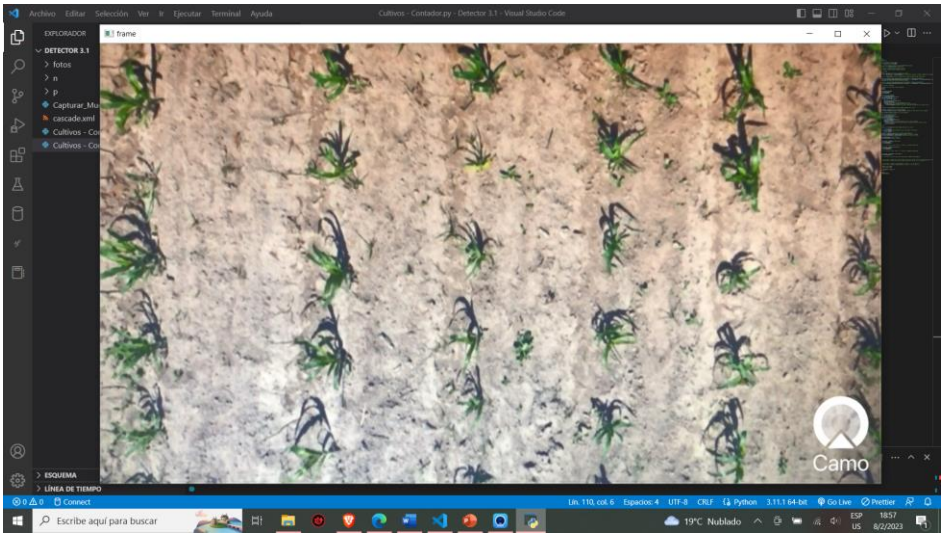


Nota. Captura de la Aplicación ejecutándose en el PC.

Paso 2.- La recepción de la transmisión en el software se da al ejecutarlo, la cámara debe estar especificada en el comando del video capture como se explica en el capítulo 2, la recepción de video en la aplicación es paralelo a la recepción de video en el software.

Figura 58

Captura de las imágenes de video en tiempo real en el software implementado.



Nota. Aplicación ejecutándose Correctamente.

La Tabla 14 resume los resultados de las pruebas de integración del módulo de transmisión y video y el módulo de recepción y procesamiento.

Tabla 14

Resultados caso de Prueba CP001

CP001				
Paso	**Descripción**	**Datos de entrada**	**Datos de salida**	**¿Realiza correctamente?**
1	Comprobar el funcionamiento de la cámara.	Ninguno	Video en tiempo real	Si
2	Verificar la recepción de video en tiempo real	Video en tiempo real	Visualización de Video	Si

Nota. El caso de prueba CP001 obtuvo un 100% de efectividad por que los módulos y los dispositivos se integraron a la perfección.

CP002 – Detección de líneas de cultivo y malezas

Paso 1.- Comprobar la detección de líneas de cultivo mediante el algoritmo desarrollado, se hará una revisión donde el objeto detectado sea lo establecido.

Paso 2.- Comprobar la detección de malezas mediante el algoritmo desarrollado, se hará una revisión donde el objeto detectado sea lo establecido.

Paso 3.- El paso 3 es prueba de detección de plantas de cultivo, se establece como objetivo general la detección de líneas de cultivo y malezas, y más no se requiere un conteo o detección de plantas de cultivos, con el fin de dar un plus al proyecto se decidió realizar una detección de plantas de cultivo e implementar un reporte con un conteo de los 3 elementos a detectar.

Tabla 15

Resultados del caso de prueba CP002.

CP002				
Paso	**Descripción**	**Datos de Entrada**	**Datos de Salida**	**¿Realiza Correctamente?**
1	Detección de líneas de Cultivo	Video en tiempo real de cultivos.	Detección de líneas de cultivo, identificándoles con una línea amarilla sobre estas.	Si
2	Detección de malezas	Video en tiempo real de cultivos.	Detección de malezas, cada maleza estará encerrada en un círculo rojo.	Si
3	Detección de Plantas de cultivo	Video en tiempo real del cultivo.	Detección de plantas de cultivo, encerradas en un círculo verde.	SI

Nota. El caso de prueba CP002 obtuvo un 100% de efectividad debido a que se logró identificar líneas de cultivo, malezas y plantas en tiempo real correctamente, además de presentar un conteo de estas.

En la Figura 59 se observa la detección de líneas de cultivo, malezas y plantas, esta detección se realiza en conjunto y en tiempo real, el conteo se presenta en la parte superior izquierda, tal y como se estableció en el desarrollo del algoritmo.

Figura 59

Resultados prueba 1, detección de líneas de cultivo, malezas y plantas.

Nota. Prueba 1: la detección alcanza un 90% de eficiencia con respecto a lo identificado por el ojo humano.

El porqué del porcentaje de eficiencia se debe al siguiente análisis realizado:

Líneas de cultivo: 7 detectadas y 7 existentes.

Maleza: No se encierra el total de malezas, el reporte es de 29 y el número exacto de malezas no es completamente visible por el tamaño de estas, entonces se establece un margen de error del 5% en base a varias pruebas con la misma imagen.

Plantas de cultivo: A excepción de dos o 3 plantas no son encerradas en un círculo verde, teniendo un total de 43 plantas, por lo que también se establece un 5% de error. Al tener un error del 5% en la detección de malezas y un 5% en la detección de plantas, el resultado de la eficiencia del software en la primera prueba es del 90%.

CP003 – Verificación de detección y conteo

Paso 1.- Se verificará el funcionamiento y eficiencia correcta de la detección y conteo del número de líneas, malezas y plantas pertenecientes al cultivo. Para verificar el funcionamiento del algoritmo se realiza una segunda prueba, para esto, la cámara estará a las distancias de 5, 10 y 15 m del cultivo, distancias establecidas para validar los resultados.

Problema 1: en la implementación del algoritmo se explicó que había que establecer rangos para los vecinos cercanos, tanto en las líneas de cultivo como en las malezas, al tener un cultivo más grande o pequeño y al acercarse o alejarse la cámara, estos rangos cambian por lo que si la detección no es satisfactoria se los debe editar.

La Tabla 16 contiene los parámetros a cambiar que se encuentran distribuidos en diferentes métodos del algoritmo, son las líneas de código que se debe modificar para realizar una mejor detección.

Tabla 16

Parámetros establecidos en el código.

		Umbral de amplitud para líneas de cultivo	Umbral de amplitud para malezas y plantas	Escala de líneas de cultivo a detectar	Escala de malezas y plantas a detectar	Rango de líneas de cultivo vecinas	Rango de malezas y plantas vecinas	Área máxima de detección
SEMANA 1	5m	150	50	2	1.5	150 px	35 px	3575 px
	10m	120	35	2	1.5	135 px	30 px	2000 px
	15m	80	35	1.5	1	120 px	25 px	1444 px
SEMANA 2	5m	130	25	1.5	1.5	130 px	40 px	2475 px
	10m	110	20	1.5	1	125 px	35 px	2200 px
	15m	110	15	1	1	125 px	40 px	1444 px
SEMANA 3	5m	130	15	1.5	1.5	110 px	30 px	1575 px
	10m	120	10	1	1	105 px	30 px	1400 px
	15m	110	15	1	1	100 px	30 px	1444 px
SEMANA 4	5m	150	35	2	1.5	150 px	35 px	3575 px
	10m	120	35	2	1.5	135 px	30 px	2000 px
	15m	80	35	1.5	1	120 px	25 px	1444

A continuación, se explicará cada parámetro establecido en la Tabla 16:

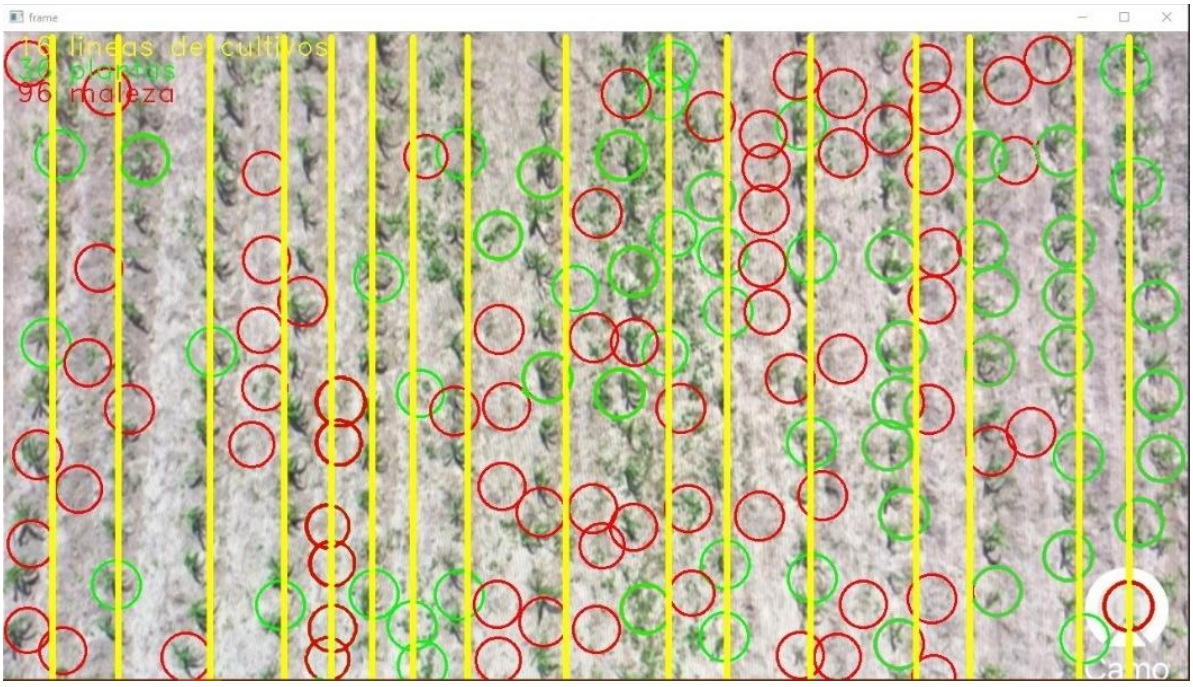
- **Umbral de amplitud para líneas de cultivo:** Es la variable T en el método de detección de líneas de cultivo, establece el umbral máximo para detectar otra línea de cultivo.
- **Umbral de amplitud para malezas y plantas:** Es la variable T en el método de detección de plantas, establece el umbral máximo para detectar otra planta de cultivo por lo que las plantas descartadas se detectan como maleza.
- **Escala de líneas de cultivo a detectar:** función scaleFactor, la escala disminuirá, ya que al alejar la cámara las líneas se acercan, teniendo en cuenta que la escala mínima es 1.

- **Escala de malezas y plantas a detectar:** Función scaleFactor, para las plantas se buscará una escala adecuada, porque a menor escala mayor retraso en la detección y posiblemente detecte un mayor número de falsos negativos.
- **Rango de líneas de cultivo vecinas:** Función minNeighbors, establece el rango de líneas de cultivo vecinas más cercanas a detectar.
- **Rango de malezas y plantas vecinas:** Función minNeighbors, establece las plantas vecinas más cercanas a detectar.
- **Área máxima de detección:** Son las variables minT y maxT, representan el ancho y altura de los objetos que se va a detectar, el valor de estas variables al alejar la cámara disminuirá, debido a que los objetos a detectar son muy pequeños.

La Figura 60 muestra una detección con mayor número de elementos, como ya se predijo el número de falsos negativos aumenta y también aumenta el tiempo de procesamiento.

Figura 60

Resultados prueba 2, detección de líneas de cultivo, malezas y plantas.



Nota. Prueba 2: la detección alcanza un 88% de eficiencia con respecto a lo identificado por el Ground Truth.

Problema 2: Un aspecto muy importante a considerar es al no poder cuadrar paralelamente la cámara al cultivo, su efecto tiene en las hileras porque no estan verticalmente exactas, al alejar más la cámara las hileras se presentan menos verticales y esto hace que se presenten falsos negativos al memento de detectar líneas de cultivo. Los resultados de eficiencia del algoritmo en el CP003 son:

Líneas de cultivo: 16 detectadas y 14 existentes, dos falsos negativos.

Maleza: No se encierra el total de malezas, el reporte es de 96 pero hay plantas de cultivo encerradas como malezas, según lo visto en la imagen por el ojo humano se establece un margen de error del 7%.

Plantas: El reporte de plantas totales de la imagen es de 36 existentes, pero lógicamente en 14 líneas de cultivo existirá un número superior de plantas, por lo que se establece un 12% de error.

En la Tabla 17 al tener un error del 2% en la detección de líneas de cultivo, un 7% en la detección de malezas y un 12% en la detección de plantas, resulta que la eficiencia del software en la segunda prueba es del 86%, un software es eficiente cuando supera el 85% de efectividad entonces se concluye que el caso de prueba CP003 si se realiza correctamente la detección.

Tabla 17

Resultado caso de prueba CP003

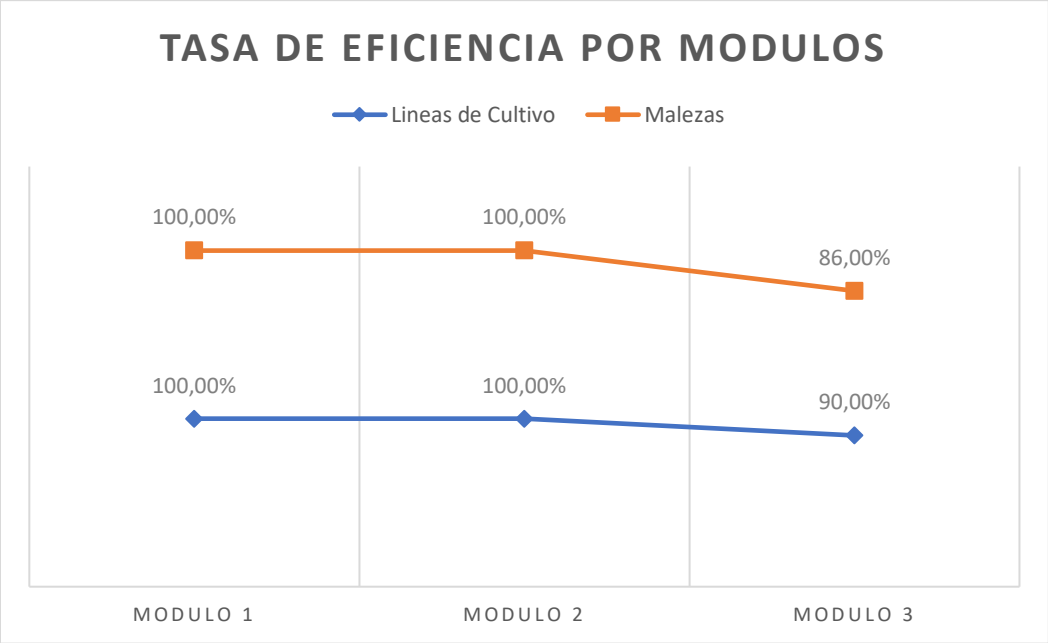
CP003				
Paso	**Descripción**	**Datos de entrada**	**Datos de salida**	**¿Realiza correctamente?**
1	Verificar el conteo de líneas de cultivo, malezas y pantas.	Detección de líneas de cultivo, malezas y plantas.	Conteo de líneas de cultivo, malezas y pantas eficientemente.	Si

Nota. El caso de prueba CP003 obtuvo un 88% de eficiencia.

Según lo analizado con los 3 casos de prueba que componen el funcionamiento y acoplamiento de los módulos de la aplicación, la Figura 61 muestra los resultados de la tasa de eficiencia del software, teniendo una eficiencia del 100% entre el Módulo 1 (Modulo de Adquisición y Transmisión) y el Módulo 2 (Modulo de recepción y procesamiento), mientras que la detección de líneas de cultivo y malezas del Módulo 3 (Modulo de reportes y resultados) presenta una mejora en las dos pruebas realizadas.

Figura 61

Tasa de Eficiencia de cada módulo



3.2. Condiciones de Medición del Software

Estas condiciones son necesarias para evaluar según las métricas establecidas y obtener resultados del software en cuanto a precisión y rendimiento.

Numero de semanas del cultivo.

El seguimiento del cultivo es por cuatro semanas, cada semana se somete al algoritmo a las pruebas necesarias para obtener las métricas requeridas.

- Nombre del cultivo: Zea mays (Cultivo de maíz)
- Tipo de cultivo: Cereales

- Distancia de una línea de cultivo: 10 m
- Distancia entre plantas en una línea de cultivo: 35 cm a 40 cm
- Distancia entre líneas de cultivo: 70cm a 120 cm

La estimación de medidas obtenidas en la Tabla 18 fueron tomadas en base a lo observado en las imágenes de pruebas comparando con medidas de crecimiento establecidas por varias investigaciones científicas.

Tabla 18

Descripción de la planta de cultivo semana 1, 2, 3 y 4

	Semana 1	**Semana 2**	**Semana 3**	**Semana 4**
Ancho de la planta (cm)	2 - 4	7 - 9	10 - 15	17 - 20
Altura de la planta (cm)	2 - 4	10 - 12	17 - 20	20 - 25
Numero de hojas	1 - 2	4 - 6	5 – 8	5 - 8
Tamaño de las hojas (cm)	1 - 2	3 - 5	8 - 10	10 - 12

Nota. El crecimiento del cultivo depende de varios factores: suelo, clima, presencia de minerales, agua, etc.

Altura de captura de las imágenes.

Técnicamente el por qué se evalúa el software sobre la altura de captura de las imágenes, es necesario tener claro algunos términos muy utilizados en el presente proyecto. Un pixel procede de picture element, es el elemento más pequeño de color uniforme que forma una imagen digital, este contiene información de color, brillo, saturación y no tienen un tamaño determinado, el tamaño viene dado por cada dispositivo. Una imagen digital es el resultado de una matriz de píxeles, es decir, es la suma de las filas y columnas de píxeles.

La resolución de una fotografía digital, la resolución es la relación entre el tamaño de pixeles de una imagen con las dimensiones físicas de la misma. Una resolución mayor implica que la imagen debe tener más pixeles, pero se debe tener en cuenta que cada dispositivo tiene una capacidad limitada para mostrar esta resolución normalmente expresada en ppp o dpi que

significa pixeles por pulgada. La captura de las imágenes se ha realizado a tres diferentes alturas 5, 10 y 15 m, cada altura tiene sus propios factores que alteran la imagen, como el tamaño de la planta, el número de hileras y el espaciado entre ellas. El tener imágenes a diferentes alturas aumenta el tamaño físico y esto conlleva a una reducción en la resolución y píxeles, este es un aspecto que puede influir en la detección de líneas de cultivo, plantas y malezas.

3.3. Procesamiento de Imágenes

Con el objetivo de determinar la eficiencia del algoritmo en cuanto a precisión y rendimiento, el algoritmo fue evaluado en 5 transmisiones de imágenes diferentes en las 3 distancias establecidas, teniendo un total de 15 imágenes de video transmitidas por semana. Las 15 imágenes fueron sometidas a un análisis cuantitativo para determinar la eficiencia del software, los resultados se presentan en tablas con los siguientes parámetros:

- **N:** Número asignado a las imágenes transmitidas en tiempo real.
- **LCR:** Líneas de cultivo reales, encontradas por el ojo humano.
- **LCD:** Líneas de cultivo detectadas y presentadas en el contador por el algoritmo.
- **MD:** Malezas detectadas y contadas por el algoritmo.
- **PCD:** Plantas de cultivo detectadas y contadas por el algoritmo.
- **PPA:** Porcentaje de precisión del algoritmo, se lo determina haciendo una regla de tres, obteniendo así la precisión del algoritmo en los 3 apartados: líneas de cultivo, plantas y malezas, luego, se realiza un promedio de los porcentajes y se tiene el porcentaje final de precisión.
- **TIEMPO DE EJECUCION:** Para el analizar el rendimiento del software, se realizó la comparación de compilación en 3 computadoras de distintas características, el algoritmo analizó imágenes a las 3 alturas establecidas y se cronometró los tiempos de ejecución. En la Tabla 19 se especifica las características de cada compilador.

Tabla 19

Características técnicas del equipo de procesamiento.

	Compilador 1 (PC1)	**Compilador 2 (PC2)**	**Compilador 3 (PC3)**
Marca	Dell	Toshiba	Samsung
Procesador	Core i7	Core i7	Core i5
Generación	7ma	11th	9na
RAM	8GB	16GB	4GB
GPU	520MB	8GB	520MB
Sistema Operativo	Windows 10	Linux (Ubuntu)	Windows 10

A continuación, los resultados para cada semana se estructuran de la siguiente manera: La primera imagen corresponde a la imagen original transmitida, la segunda a la captura de la detección de líneas de cultivo, plantas y malezas por parte del algoritmo, y al final la tabla de resultados.

3.3.1. Primera semana

El cultivo en la primera semana no se visualiza salvo algunas excepciones debido al tamaño de las plantas, en la imagen de la Figura 62, se distinguen pequeños rastros de vegetación lo cual hace muy difícil considerar si es maleza o planta de cultivo.

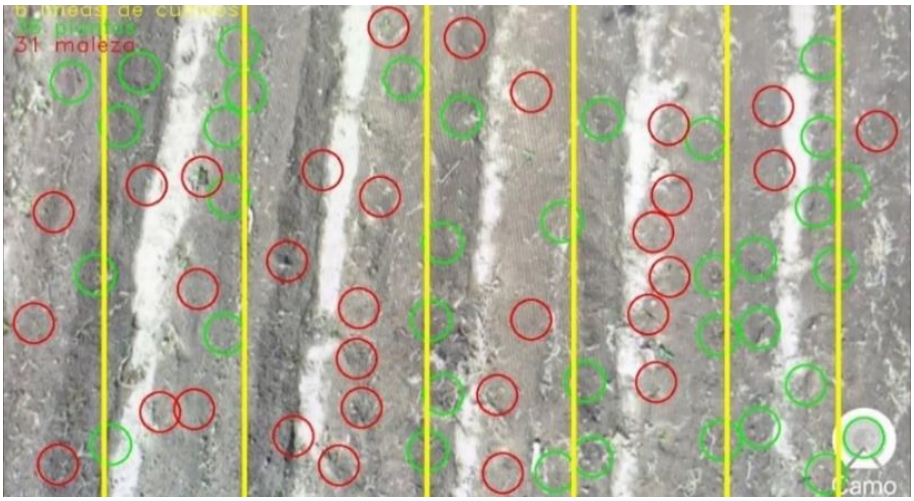
Figura 62

Imagen semana 1 a 5 m de altura.

En la Figura 63 se observa el número de líneas de cultivo detectadas corresponde perfectamente a las líneas de cultivo que se percibe a la vista, aclarando el problema 2 en el caso de prueba CP003, donde las líneas de cultivo no se encuentran paralelas a las líneas graficadas debido a las imágenes transmitidas.

Figura 63

Resultado de detección, semana 1 a 5 m de altura.



Resultados: Existe una diferencia que se puede visualizar entre la maleza y plantas, esta se da por la distancia que se encuentran de las líneas de cultivo.

La Tabla 20 presenta los resultados del algoritmo a la detección de líneas de cultivo malezas y plantas para las imágenes de la semana 1 a 5 metros de altura.

Tabla 20

Resultados semana 1 y 5 m de altura.

SEMANA 1 A 5M DE ALTURA									
Detección							Tiempo de Ejecución (s)		
No.	LCR	LCD	MD	PCD	Ground truth cultivo	Ground truth malezas	PC1	PC2	PC3
1	6	6	31	36	93.30	6.4	12	8	30
2	6	6	38	25	91.20	8.8	13	6	29
3	4	3	44	23	87.46	12.54	11	6	28
4	6	5	39	21	72.56	27.44	12	7	29
5	6	6	40	22	63.87	36.13	12	6	28

Nota. El crecimiento del cultivo depende de varios factores: suelo, clima, presencia de minerales, agua, etc.

Como se explicó en el caso de prueba CP003, al aumentar la distancia de la cámara el software debe ser nuevamente configurado, en la imagen de la Figura 64 el tamaño de los objetos a detectar cambia, así como, la escala y rango de los vecinos cercanos.

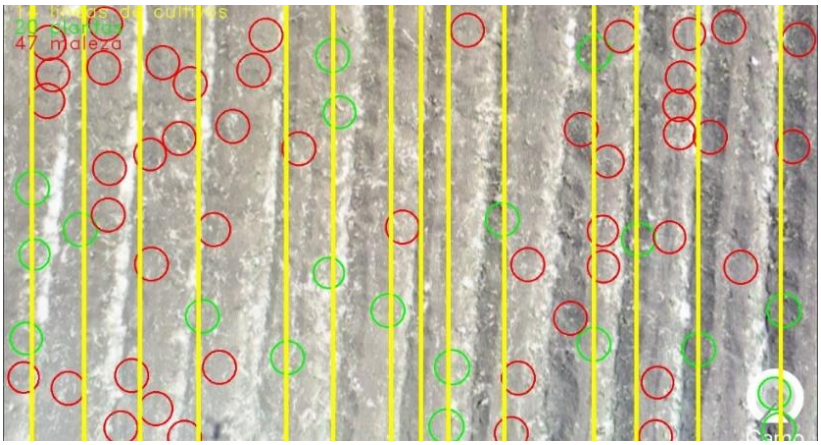
Figura 64

Imagen semana 1 a 10 m de altura.

Las malezas y plantas del cultivo prácticamente no se las observan ni tampoco se las clasifica, lo que se identifica es un mayor número de elementos verdosos, la Figura 65 muestra los resultados, teniendo gráficas para los 3 elementos a detectar.

Figura 65

Resultado de detección, semana 1 a 10 m de altura



Resultados: El procesamiento de las imágenes de video es más lento debido a que aumenta el número de píxeles con elementos a procesar, lo que el software detecta son pequeños puntos para graficar las líneas de cultivo y toma como referencia a estas utilizando los rangos de cercanía para graficar si es maleza o no.

La Tabla 21 detalla los resultados, estos resultados no tienen una mayor comparación con lo captado por el ojo humano por lo ya mencionado que el cultivo no es identificable, sino que se realiza un análisis lógico de dónde está una línea de cultivo y dónde posiblemente estarían las malezas y plantas, con esto se concluye que el algoritmo no tiene un gran margen de error.

Tabla 21

Resultados semana 1 y 10 m de altura.

SEMANA 1 A 10M DE ALTURA									
DETECCION							TIEMPO DE EJECUCION (s)		
No. De Imagen	LCR	LCD	MD	PCD	Ground truth cultivo	Ground truth malezas	PC1	PC2	PC3
1	14	13	47	20	62.56	37.44	15	10	35
2	13	12	53	19	48.32	51.68	16	8	32
3	14	14	59	21	67.82	32.18	14	9	35
4	13	12	62	17	69.49	30.51	15	7	34
5	14	13	58	17	73.41	26.59	16	10	36

El cultivo prácticamente ha desaparecido en la Figura 66, a una altura de 15 metros lo que se observa son líneas verdes.

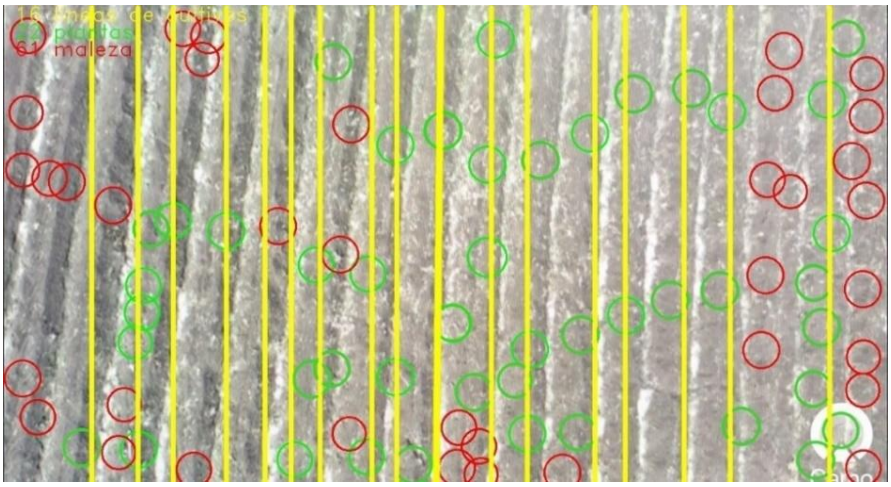
Figura 66

Imagen semana 1 a 15 m de altura.

El algoritmo busca parámetros similares a los entrenados, se encerró el área de una planta en específico y no se entrenó para identificar áreas conformadas por un conjunto de plantas. La Figura 67 presenta una detección en su mayoría de líneas de cultivo y al estar estás tan cercanas el algoritmo identifica mayormente plantas de cultivo dejando a las malezas en los bordes.

Figura 67

Resultado de detección, semana 1 a 15 m de altura.



Resultados: La efectividad en un gran porcentaje se centra en la detección de líneas de cultivo y plantas a la cercanía de estas, la detección de maleza no es precisa por lo que la transmisión de las imágenes es en tiempo real y los bordes se están actualizando constantemente, es decir, la maleza no solo se ubica en los bordes de la imagen. La Tabla 22 muestra los resultados obtenidos mediante el reporte que presenta el software.

Tabla 22

Resultados Semana 1 a 15m de altura.

SEMANA 1 A 15 M DE ALTURA									
DETECCION							TIEMPO DE EJECUCION (s)		
No. De Imagen	LCR	LCD	MD	PCD	Ground truth cultivo	Ground truth malezas	PC1	PC2	PC3
1	19	16	61	22	85.27	14.73	18	13	36
2	18	24	65	19	88.46	11.54	20	11	30
3	28	32	69	20	76.53	23.47	19	14	33
4	27	31	58	21	91.32	8.68	18	13	36
5	25	20	63	17	85.73	14.27	15	13	32

3.3.2. Segunda semana

La principal característica como se observa en la Figura 68 de la segunda semana, se da al distinguir el cultivo en las imágenes, la clasificación sigue siendo difícil.

Figura 68

Imagen semana 2 a 5 m de altura.

La detección de hileras no es tan precisa, en la Figura 69 una línea de cultivo se grafica en la mitad del área de densidad de verde sin ubicarse en las plantas, el tiempo de procesamiento aumenta con respecto al procesamiento de la semana 1 a la misma altura.

Figura 69

Resultado de detección, semana 2 a 5 m de altura.

Resultados: Para el ojo humano todavía existe una confusión entre plantas de cultivo y maleza, para el algoritmo la detección de líneas no es precisa, no ignora la maleza y traza una línea sobre esta, en el resto del cultivo existe algunos falsos positivos, es decir, se encierra plantas con círculos rojos de maleza, tanto para el ojo humano como para el algoritmo la identificación sigue siendo difícil. En la Tabla 23 se detalla resultados de las 5 imágenes evaluadas, la detección mejora debido a que hay zonas del cultivo donde las plantas presentan mayor tamaño a la maleza.

Tabla 23

Resultados Semana 2 a 5m de altura.

SEMANA 2 A 5 M DE ALTURA									
DETECCION							TIEMPO DE EJECUCION (s)		
No. De Imagen	LCR	LCD	MD	PCD	Ground truth cultivo	Ground truth malezas	PC1	PC2	PC3
1	5	5	41	28	34.71	65.29	15	10	34
2	6	5	58	22	21.04	78.96	12	11	32
3	6	5	61	23	18.94	81.06	13	10	29
4	5	5	59	24	20.56	79.44	12	9	34
5	5	5	64	21	18.04	81.96	11	10	30

Al alejar la cámara en un cultivo de dos semanas, como se observa en la Figura 70, el cultivo presenta un tono verde en toda el área, teniendo algunas zonas con mayor concentración y sin ningún parámetro que diferencie si es maleza o no.

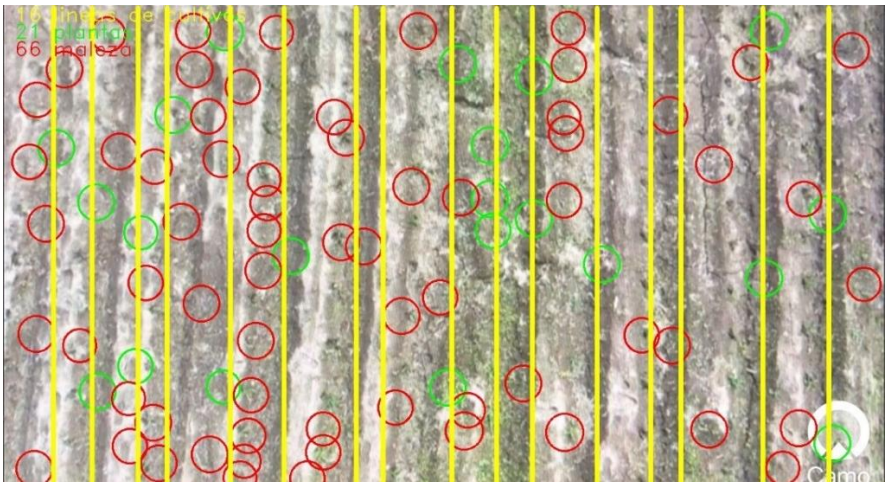
Figura 70

Imagen semana 2 a 10 m de altura.

El tiempo de procesamiento del algoritmo sigue aumentando y como se observa en la Figura 71 se obtiene un análisis que sigue siendo confuso, existe una mayor concordancia en la identificación de las líneas de cultivo, pero no en la identificación de malezas y plantas.

Figura 71

Resultado de detección, semana 2 a 10 m de altura.



Resultados: Al contrario de la semana 1 el sistema detecta mayormente maleza que plantas de cultivo, líneas de cultivo con poco rango de separación y un tanto inexactas en su

ubicación, pero en el número de detecciones no tiene un margen grande de error a las identificadas por el ojo humano.

Tabla 24

Resultados Semana 2 a 10m de altura.

SEMANA 2 A 10 M DE ALTURA									
	DETECCION						TIEMPO DE EJECUCION (s)		
No. De Imagen	LCR	LCD	MD	PCD	Ground truth cultivo	Ground truth malezas	PC1	PC2	PC3
1	13	16	66	21	23.64	76.36	22	16	41
2	14	16	69	29	20.52	79.48	20	18	37
3	13	14	82	24	23.09	76.91	21	14	33
4	14	16	73	28	17.38	82.62	22	12	41
5	13	15	79	27	42.61	57.39	22	12	39

Un factor importante en la obtención de imágenes es la iluminación, en la Figura 72 se tiene una imagen distinta a las demás con mejores condiciones de luz lo que permite tener mayor detalle de los cultivos.

Figura 72

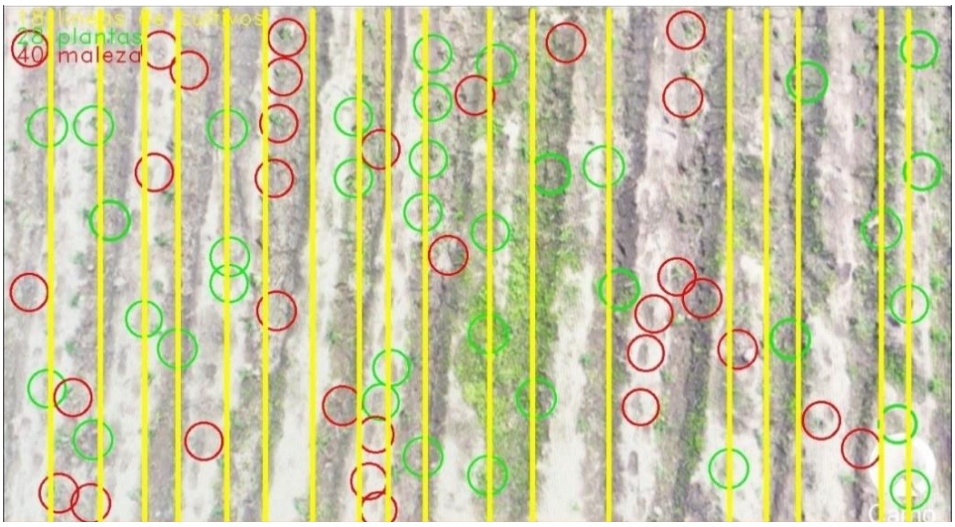
Imagen semana 2 a 15 m de altura.

Al tener mayor detalle en una imagen la detección se vuelve significativamente mejor, concluyendo que a mejor calidad de imagen mejor detección. La Figura 73 muestra una

detección de líneas de cultivo disparejas debido a que las hileras no se encuentran verticalmente, el algoritmo detecta menor número de elementos.

Figura 73

Resultado de detección, semana 2 a 15 m de altura.



Resultados: Mejor detección a mayor calidad de imagen, los elementos a detectar son menores y concuerdan donde: las plantas están en una línea de cultivo y las malezas se concentran más en el área entre hileras. La Tabla 25 detalla los resultados, el tiempo de ejecución disminuye por la calidad de la imagen y el número de malezas y plantas no es exacto a lo observado por el ojo humano.

Tabla 25

Resultados semana 2 a 15 m de altura.

SEMANA 2 A 15 M DE ALTURA									
	DETECCION						TIEMPO DE EJECUCION (s)		
No. De Imagen	LCR	LCD	MD	PCD	Ground Truth Cultivo	Ground Truth Malezas	PC1	PC2	PC3
1	14	18	40	28	64.26	35.74	17	13	35
2	13	16	54	19	47.69	52.31	16	12	32
3	11	10	47	17	73.67	26.33	15	13	30
4	12	11	63	21	71.42	28.58	16	11	34
5	11	13	65	18	67.82	32.18	17	12	34

3.3.3. Tercera semana

Los resultados para la tercera semana mejoraron significativamente, el cultivo es visible tanto para el ojo humano como para el algoritmo, a su vez, ya se identifica la maleza con las plantas de cultivo.

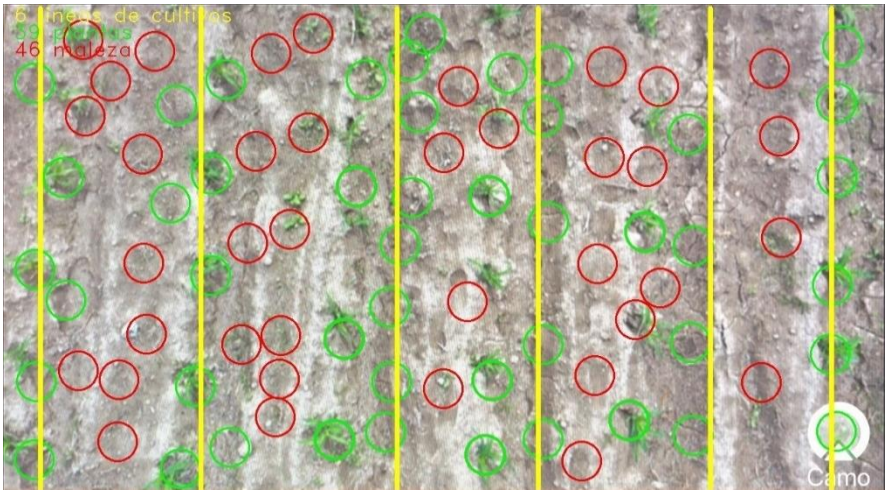
Figura 74

Imagen semana 3 a 5 m de altura.

La Figura 75 muestra una mejor detección a comparación de las semanas anteriores, las plantas de cultivo se encuentran en las hileras y la maleza mayormente entre estas, el tiempo de ejecución disminuye porque los objetos son más visibles.

Figura 75

Resultado de detección, semana 3 a 5 m de altura.



Resultados: Las líneas de cultivo coinciden y perfectamente ubicadas, las plantas del cultivo visualmente si forman una hilera, la detección de malezas presenta falsos positivos, pero no en mayor número.

Tabla 26

Resultados semana 3 a 5 m de altura.

SEMANA 3 A 5 M DE ALTURA									
DETECCION							TIEMPO DE EJECUCION (s)		
No. De Imagen	LCR	LCD	MD	PCD	Ground truth cultivo	Ground truth malezas	PC1	PC2	PC3
1	6	6	46	39	72.08	27.92	11	8	27
2	6	6	58	30	78.91	21.09	11	10	29
3	6	6	63	27	92.86	7.14	13	8	26
4	6	5	68	28	94.79	5.21	12	9	27
5	6	6	67	29	86.62	13.38	10	8	26

En la semana 3 a una altura de 10 metros es una realidad que el cultivo ya se identifica de la maleza, la imagen original: Figura 76, presenta las líneas de cultivo visibles con rasgos claros y distintos entre la maleza y plantas de cultivo.

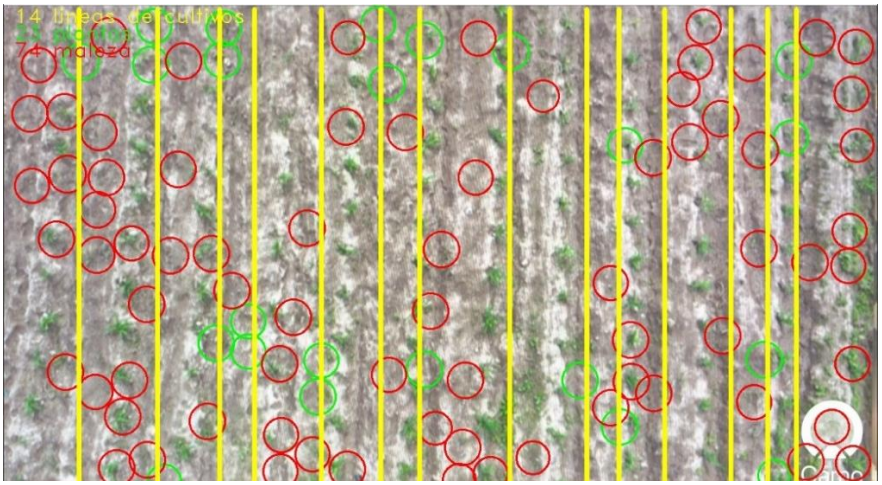
Figura 76

Imagen semana 3 a 10 m de altura.

El proceso de detección efectivo es la identificación de líneas de cultivo, la maleza y las plantas se confunden razón por la cual en muchas imágenes detecta más maleza que plantas, como se puede observar en la Figura 77

Figura 77

Resultado de detección, semana 3 a 10 m de altura.



Resultados: La identificación de maleza tiene un porcentaje alto de efectividad, al comparar con lo visto por el ojo humano, más del 80% de la maleza está detectada, para las líneas de cultivo a mayor distancia de la cámara existen más elementos a detectar y puede haber confusión en la localización.

Tabla 27

Resultados semana 3 a 10 m de altura.

SEMANA 3 A 10 M DE ALTURA									
DETECCION							TIEMPO DE EJECUCION (s)		
No. De Imagen	LCR	LCD	MD	PCD	Ground Truth Cultivo	Ground Truth Malezas	PC1	PC2	PC3
1	13	14	74	23	59.87	40.13	18	13	34
2	12	11	79	25	63.79	36.21	21	14	31
3	13	12	81	22	61.04	38.96	19	13	32
4	13	10	85	21	65.93	34.07	23	10	29
5	13	14	88	24	68.94	31.06	22	11	27

Un patrón que se está dando en las semanas anteriores y se repite en esta es: a la altura de 10 m se identifica mayormente la maleza y a la altura de 15 m se identifica las plantas de cultivo, en la Figura 78 existe un gran número de elementos a detectar y clasificar, razón por la cual el tiempo de ejecución aumenta considerablemente.

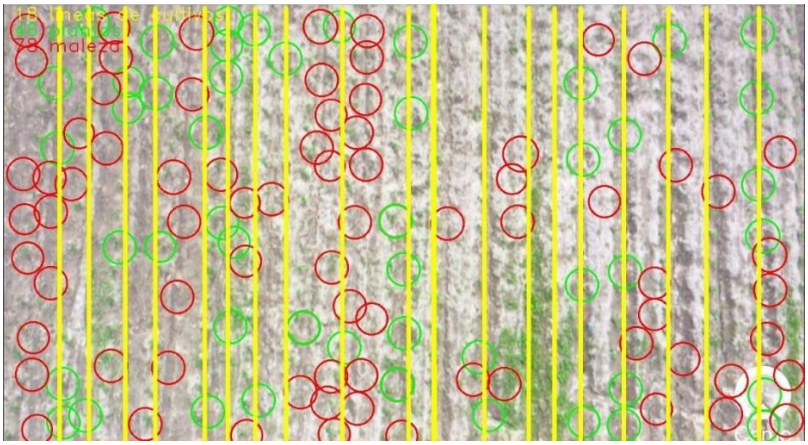
Figura 78

Imagen semana 3 a 15 m de altura.

En la Figura 79 se comprueba que el algoritmo sigue la tendencia en la efectividad de detección de líneas de cultivo, al tener las plantas y líneas de cultivo tan juntas la detección de malezas se complica, pero aun así el algoritmo conserva la lógica, encerrando la maleza entre las hileras.

Figura 79

Resultado de detección, semana 3 a 15 m de altura.



Resultados: Las líneas de cultivo, en algunos casos, no están graficadas precisamente arriba de las plantas, la imagen no está en un plano vertical completamente. Al tener una presencia alta de maleza, mirándose como una gran mancha verde no es detectada ni como planta ni como maleza, a continuación, la Tabla 28 muestra los resultados.

Tabla 28

Resultados semana 3 a 15 m de altura

SEMANA 3 A 15 M DE ALTURA									
	DETECCION						TIEMPO DE EJECUCION (s)		
No. De Imagen	LCR	LCD	MD	PCD	Ground truth cultivo	Ground truth malezas	PC1	PC2	PC3
1	20	18	78	46	83.41	16.59	21	15	32
2	19	18	84	56	58.12	41.88	19	16	30
3	20	17	93	49	58.37	41.63	16	12	30
4	19	18	97	53	69.02	30.98	18	13	27
5	21	19	91	48	61.40	38.60	18	16	31

3.3.4. Cuarta semana

La semana final del seguimiento presentó los mejores resultados en la precisión del algoritmo, las plantas del cultivo alcanzaban una altura que permite distinguir la maleza con las plantas. Los espacios entre hileras son claros y definidos, un punto a favor es que la iluminación de las imágenes era adecuada y sobreponía los detalles del cultivo favorablemente. La Figura 80 muestra cómo se encuentra el cultivo en la semana 4.

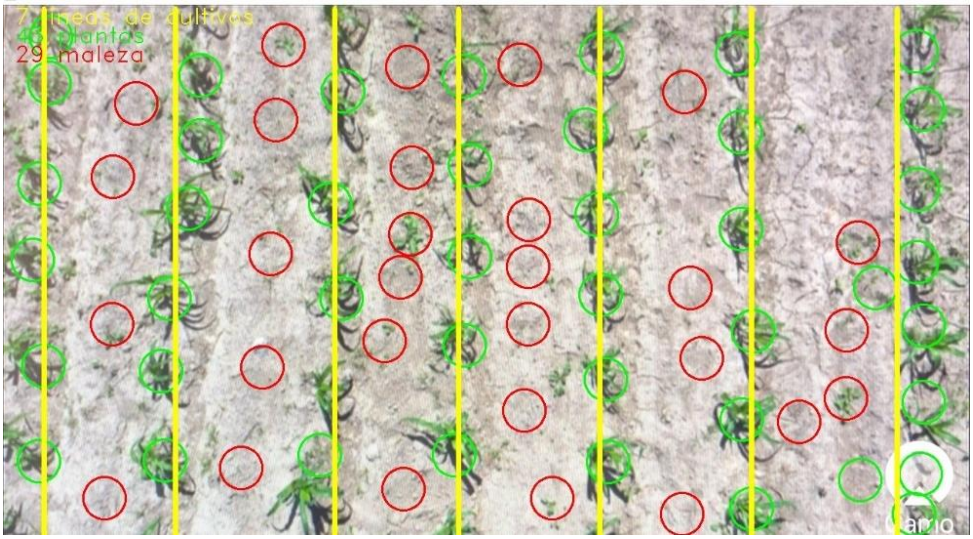
Figura 80

Imagen semana 4 a 5 m de altura

En la Figura 81 se comprueba una efectividad del 98% en todo el proceso, porque las líneas de cultivo se encuentran perfectamente alineadas encima de cada planta identificada y la maleza esta detectada perfectamente entre las hileras sin confundirse ninguna.

Figura 81

Resultado de detección, semana 4 a 5 m de altura.



Resultados: La eficiencia en la precisión y conteo en la detección de líneas de cultivo y malezas es más del 95%, los resultados coinciden con lo visto por el ojo humano y el proceso por parte del algoritmo no dura más de 3 a 4 s en la PC con mejores prestaciones.

Tabla 29

Resultados Semana 4 a 5m de altura.

SEMANA 4 A 5 M DE ALTURA									
DETECCION							TIEMPO DE EJECUCION (s)		
No. De Imagen	LCR	LCD	MD	PCD	Ground truth cultivo	Ground truth malezas	PC1	PC2	PC3
1	6	6	46	39	81.14	18.16	8	4	19
2	7	7	52	30	90.02	9.98	6	3	21
3	7	7	69	27	89.43	10.57	6	3	18
4	7	6	61	28	90.96	9.04	7	4	17
5	7	7	57	31	91.44	8.56	6	4	19

Al alejar la cámara las hileras se mantienen identificables, la maleza si llega a confundirse con los cultivos en ciertas zonas del área, en la Figura 82 se observa como las plantas de cultivo tienen un mayor tamaño que la maleza que se encuentra entre las hileras.

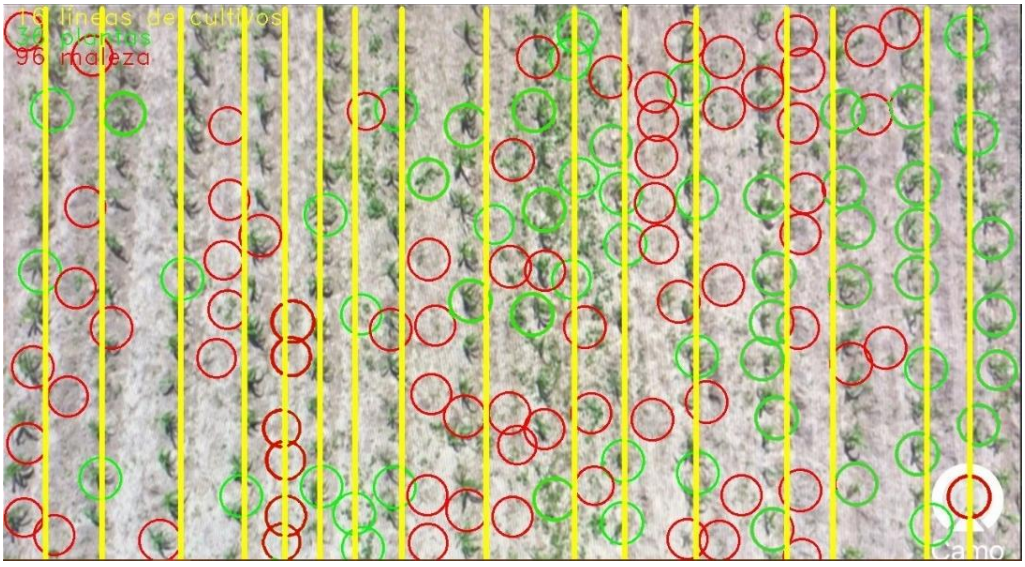
Figura 82

Imagen semana 4 a 10 m de altura

En la detección, el algoritmo cumple con la efectividad con la identificación de líneas de cultivo, pero la maleza se confunde con las plantas en algunas zonas y en otras como se muestra en la Figura 83 la maleza se la identifica como planta, teniendo falsos positivos.

Figura 83

Resultado de detección, Semana 4 a 10m de altura.



Resultados: La precisión y conteo en la detección de líneas de cultivo y malezas es más del 90% por lo visto con el ojo humano, en zonas verdes, el algoritmo tomó la mayoría de las plantas, debido a la cercanía con las hileras.

Tabla 30

Resultados Semana 4 a 10 m de altura.

SEMANA 4 A 10 M DE ALTURA									
DETECCION							TIEMPO DE EJECUCION (s)		
No. De Imagen	LCR	LCD	MD	PCD	Ground truth cultivo	Ground truth malezas	PC1	PC2	PC3
1	15	16	96	36	79.14	20.85	11	8	22
2	15	16	103	31	59.83	41.07	10	7	20
3	12	14	112	34	48.12	51.88	11	6	19
4	14	16	114	33	72.03	27.97	12	9	20
5	14	15	109	38	76.32	23.68	11	8	19

Al evaluar las imágenes adquiridas a una altura de 15 m y tener un cultivo avanzado en dimensiones, se tiene varias zonas verdes en el área formando un solo conjunto, a continuación, en la Figura 84 se muestra la imagen original a evaluar.

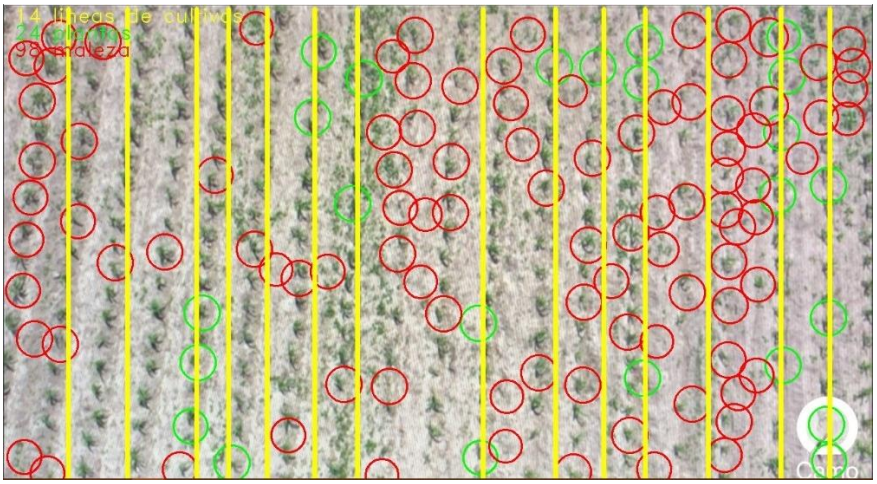
Figura 84

Imagen Original Semana 4 a 15m de altura

A 15 m de altura algunas líneas de cultivo se distorsionan y no son todas verticales, pero la eficiencia de detección se mantiene alta, en cuanto a la maleza debido a la cercanía de las líneas de cultivo no es tan específica, en la figura 85 se presenta un mayor número de elementos encontrados como malezas.

Figura 85

Resultado de detección, Semana 4 a 15 m de altura.



Resultados: Se mantiene la eficiencia en la detección de líneas de cultivo, pero el análisis de la maleza es confuso debido a la cercanía de estas a las hileras. La Tabla 31 presenta los resultados correspondientes a la última semana de seguimiento y la máxima altura de la cámara sobre el cultivo.

Tabla 31

Resultados Semana 4 a 15m de altura.

SEMANA 4 A 15 M DE ALTURA									
DETECCION							TIEMPO DE EJECUCION (s)		
No. De Imagen	LCR	LCD	MD	PCD	Ground Truth Cultivo	Ground Truth Malezas	PC1	PC2	PC3
1	21	14	98	24	76.93	23.07	16	12	28
2	21	19	106	31	74.02	25.98	17	11	26
3	20	18	111	33	65.91	34.09	14	11	27
4	21	18	108	46	75.17	24.83	18	13	29
5	19	17	103	52	69.41	30.59	18	13	31

3.4. Resultados Cuantitativos

Para tener una medida cuantitativa exacta del grado en que un sistema es eficiente, es necesario establecer métricas que midan la calidad y rendimiento, centrándose en la solución que brinda el software, para ello se establecerán los siguientes criterios.

3.4.1. Detección de objetos

Al desarrollar un sistema de inteligencia artificial ya sea en el campo de visión por computador u otros, el algoritmo involucra parámetros como: aprendizaje, recuperación de información, Re-identificación, etc. Para ello, es necesario analizar el rendimiento que estos tienen.

La métrica que se va a utilizar es Mean Average Precision (mAP), es muy usada para evaluar la precisión de detectores de objetos, la precisión mide las predicciones del algoritmo, es decir, busca la capacidad de detección correctamente del software, teniendo una relación entre las predicciones correctas y el total de predicciones a realizar, su definición matemática es la siguiente:

$$Presicion = \frac{TP}{TP + FP}$$

Donde:

TP: Verdaderos positivos, es la predicción correcta verdadera realizada.

FP: Falso positivo, es la predicción incorrecta realizada.

FN: Falsos negativos, Predicción incorrecta falsa.

TN: Verdaderos negativos, Predicción incorrecta verdadera.

To: Número total de elementos detectados que han sido clasificados.

El caso de ejemplo se lo hará para la semana 1 a 5 m de altura, cuyos datos de detección se tiene en las tablas de resultados por semana y a cada altura del apartado 3.3 procesamiento de imágenes.

Siguiendo el apartado mencionado anteriormente, se tiene: para líneas de cultivo LCR (líneas de cultivo detectadas por el ojo humano) y LCD (líneas de cultivo detectadas por el algoritmo), así que, como verdaderos positivos (TP), la suma de las LCD detectadas en las 5 imágenes analizadas, los falsos positivos (FP), se obtienen de la resta de las LCR detectadas en las 5 imágenes menos las LCD.

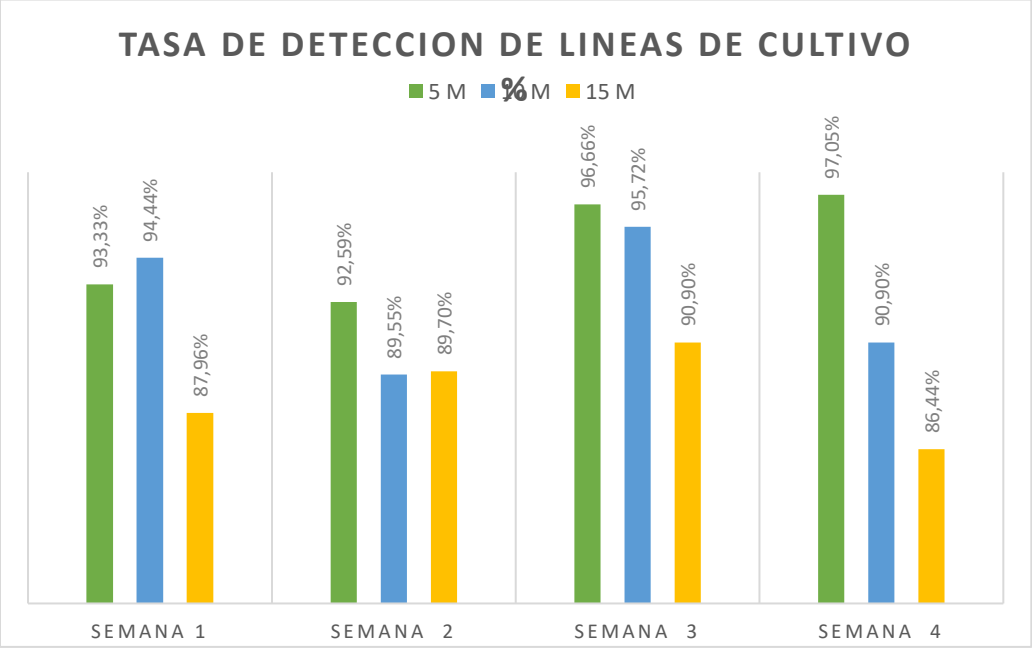
Entonces:

$$Presicion = \frac{28}{28 + 2} = 0{,}9333 * 100 = 93{,}33\%$$

El porcentaje obtenido corresponde al promedio resultante de los valores de las 5 imágenes de la semana 1 a 5 m de altura, en la Figura 86, con un gráfico estadístico se presentan la tasa de detección o precisión del algoritmo para las líneas de cultivo, cada porcentaje fue realizado en función a la métrica Mean Average Precision (mAP) con un conjunto de 4200 imágenes las cuales pertenecen a cada semana y a cada altura.

Figura 86

Tasa de detección de líneas de cultivo



Nota. La tasa de detección de líneas de cultivo es del 92,10%, este porcentaje es el porcentaje total del promedio de los porcentajes obtenidos de cada semana a las diferentes alturas.

Para analizar y poder obtener un porcentaje en la detección de malezas se define una medida de exactitud en una matriz de confusión, una matriz de confusión es una herramienta para visualizar el rendimiento de un clasificador, la exactitud mide la proximidad entre el resultado global del clasificador y la clasificación perfecta, es ideal para saber la proximidad del clasificador a la detección perfecta de la maleza.

La Tabla 32 muestra una matriz de confusión, la primera columna está conformada por las respuestas positivas que el clasificador ha detectado, en este caso serían las malezas detectadas, la primera fila se tiene los verdaderos positivos (TP), representan la maleza detectada y clasificada, la segunda fila contiene los falsos positivos (FP) que son los incorrectamente clasificados. En la segunda columna se tiene las respuestas negativas del clasificador, en la primera fila están los falsos negativos (FN), es la maleza que ha sido incorrectamente clasificada como plantas de cultivo, la segunda fila contiene los verdaderos negativos (TN), estos representan a las plantas de cultivo que han sido correctamente identificadas.

Tabla 32

Matriz de confusión: detección de malezas

		Resultado de la Clasificación	
		MALEZA	NO MALEZA
Instancias Verdaderas	MALEZA	Verdaderos Positivos	Falsos Negativos
	NO MALEZA	Falsos Positivos	Verdaderos Positivos

Para evaluar cuantitativamente se define la siguiente ecuación:

$$Exactitud = \frac{TP + TN}{To}$$

Donde:

Se calcula como la división entre la suma de verdaderos positivos (TP) y verdaderos negativos (TN) con el número total de objetos detectados que han sido clasificados (To) y que corresponden al número total de elementos que están dentro de la matriz de confusión. En el presente proyecto, los verdaderos positivos son las malezas detectadas, más los verdaderos negativos que son las plantas detectadas como malezas divididas al total de elementos detectados entre malezas y plantas de cultivo.

En la semana 1 a 5 m de altura, la sumatoria de malezas detectadas para las 5 imágenes en la Tabla 17 es de 140 los cuales son (TP), el número detectado para plantas de cultivo que han sido identificadas como malezas es (TN) y el número de predicciones (To) sería igual a la suma de las malezas detectadas (TN) con el número de plantas detectadas (PCD), teniendo un total de 294 elementos detectados y clasificados.

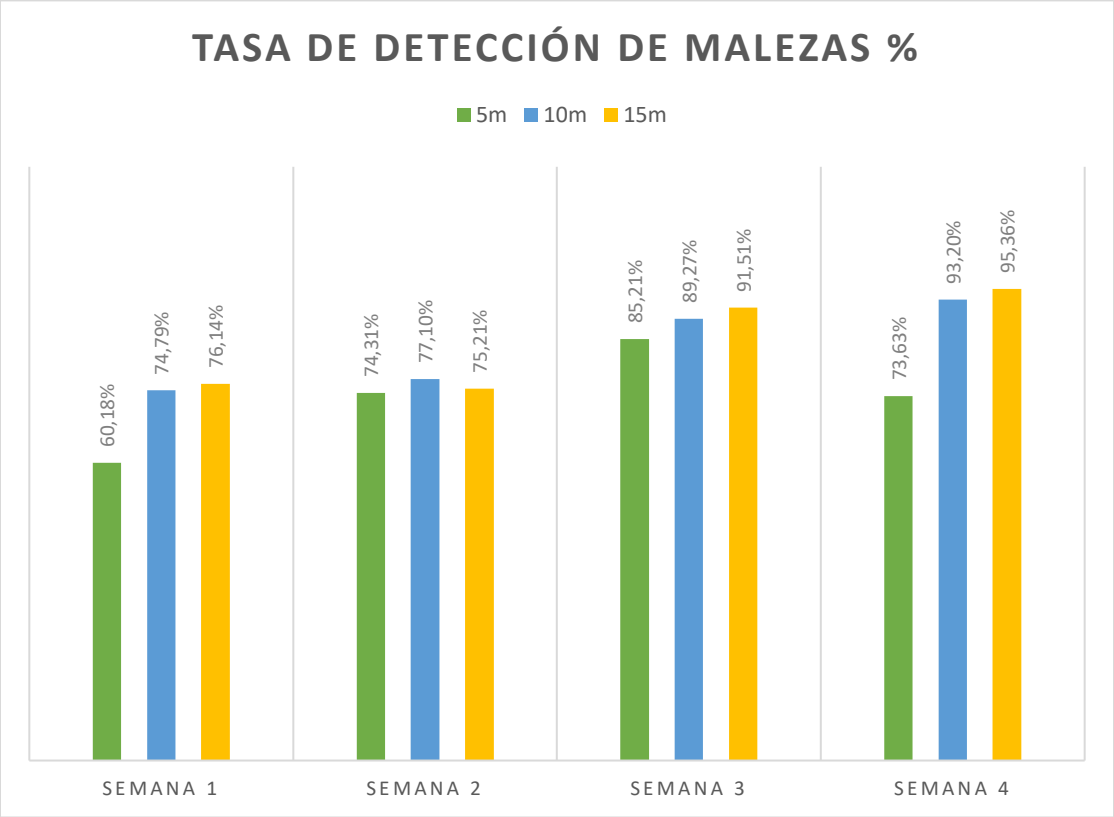
$$Exactitud = \frac{192 + 0}{319}$$

$$Exactitud = \ 0{,}6018 * 100 = 60{,}18\%$$

Los verdaderos negativos (TN) tienen un valor de 0, al tener un cultivo de semana 1 no se logra identificar malezas y plantas de cultivo, por lo que asumimos que no existe malezas detectadas como plantas, igualmente, por esta razón el porcentaje de detección es muy bajo. Hay que recalcar que el cálculo realizado es para el set de imágenes a 5 m de altura de la semana 1, a continuación, en la Figura 87 se presenta el porcentaje promedio de las 60 imágenes evaluadas.

Figura 87

Tasa de detección de malezas



Nota. La Tasa de detección de malezas es del 80,49%, este porcentaje es el porcentaje total del promedio de los porcentajes obtenidos de cada semana a las diferentes alturas.

3.4.2. Tiempo de procesamiento

Para evaluar el procesamiento y rendimiento del algoritmo, se consideró algunos parámetros: La resolución con que se transmite las imágenes en tiempo real es de 720 p, conocida como HD, los FPS (Frames per second o Frames por segundo) es la cantidad de imágenes consecutivas que se muestran en pantalla por cada segundo mientras la cámara enfoca al cultivo, un vídeo es en realidad una secuencia de fotogramas que pasan a gran velocidad, en este proyecto las imágenes son transmitidas a 30 fps. Los tiempos de ejecución evaluados corresponden a las imágenes obtenidas a las 3 alturas de la cámara y 4 semanas de seguimiento del cultivo, el tiempo de procesamiento fue cambiando debido al número de píxeles que existen en un análisis de un cultivo.

Técnicamente, se utilizó los principios cuantitativos de diseño de computadoras, para evaluar el rendimiento del CPU, entendiéndose como pulsos de reloj o ciclos de reloj, además del número de ciclos de reloj para ejecutar un programa, también intervienen el número de instrucciones ejecutadas (IC), si se conoce el número de ciclos de reloj y el recuento de instrucciones podemos calcular el número medio de ciclos de reloj por instrucción (CPI), con la siguiente fórmula:

$$CPI = \frac{Ciclos\ del\ reloj\ del\ CPU\ para\ un\ programa}{IC}$$

El cálculo del CPI es útil en el cálculo del tiempo de CPU, es decir, el rendimiento del CPU, este rendimiento dependerá de tres características:

- Ciclos de reloj o frecuencia
- Ciclos del reloj por instrucción ejecutada CPI
- Número de instrucciones IC.

Para realizar el cálculo del tiempo de CPU se tiene:

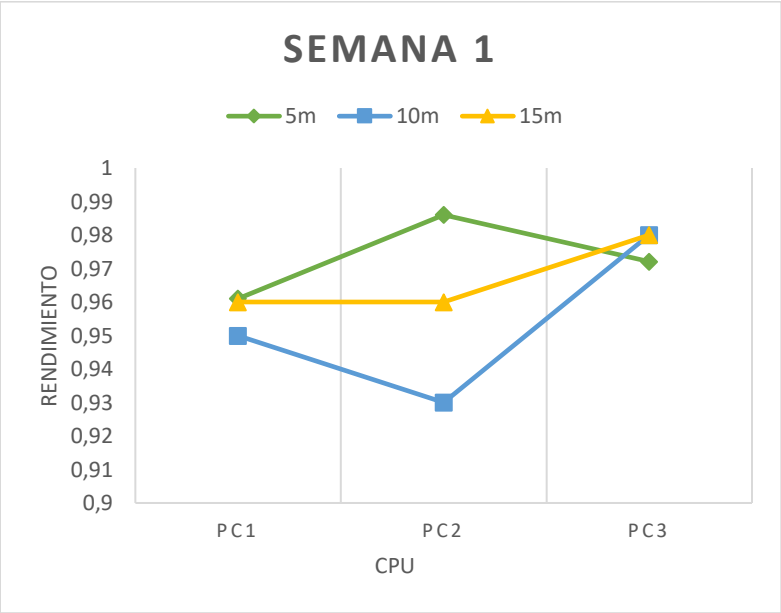
$$Tiempo\ de\ CPU = \frac{IC\ x\ CPI}{Frecuencia\ de\ Reloj}$$

En este proyecto definimos como: Ciclos o frecuencia del reloj al tiempo en que tarda la CPU en realizar la detección de malezas, líneas de cultivo y numero de instrucciones al total de elementos detectados tanto en malezas, líneas de cultivo y plantas, los datos de estas variables se encuentran detallados en las tablas de resultados.

Las Figuras 88, 89, 90 Y 91 presentan datos de rendimiento del CPU, datos que fueron obtenidos utilizando las ecuaciones mencionadas anteriormente, estos valores pertenecen a cada semana y son el promedio de la evaluación de rendimiento de CPU para 5 imágenes transmitidas a cada altura, conforme a las tablas de resultados este análisis se realiza para tres CPU, entendiendo que 1 es el valor máximo de rendimiento del CPU y 0 es el valor donde la CPU no realiza ninguna tarea.

Figura 88

Rendimiento de la CPU, semana 1.



Nota. La PC2 es el computador con mejores características de hardware y software y tiene menos tiempo de procesamiento porque detecta de una forma más rápida la maleza y líneas de cultivo, pero, al evaluar el rendimiento del CPU la PC3 la cual tiene las características más básicas de las 3 es la que tiene un mayor rendimiento.

Figura 89

Rendimiento de la CPU, semana 2.

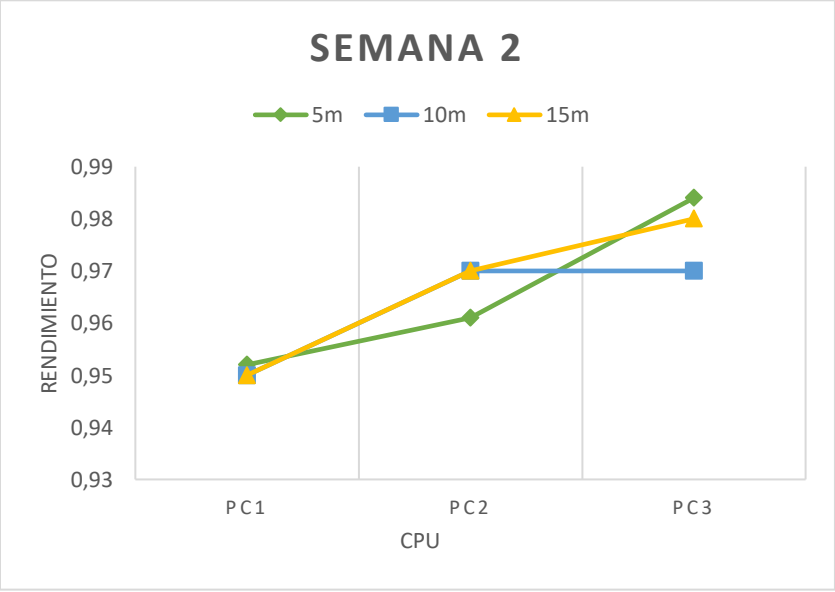
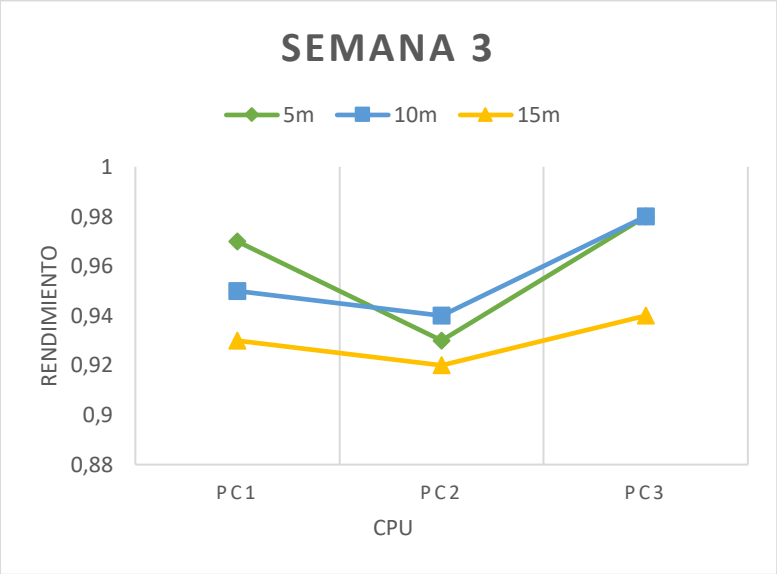


Figura 90

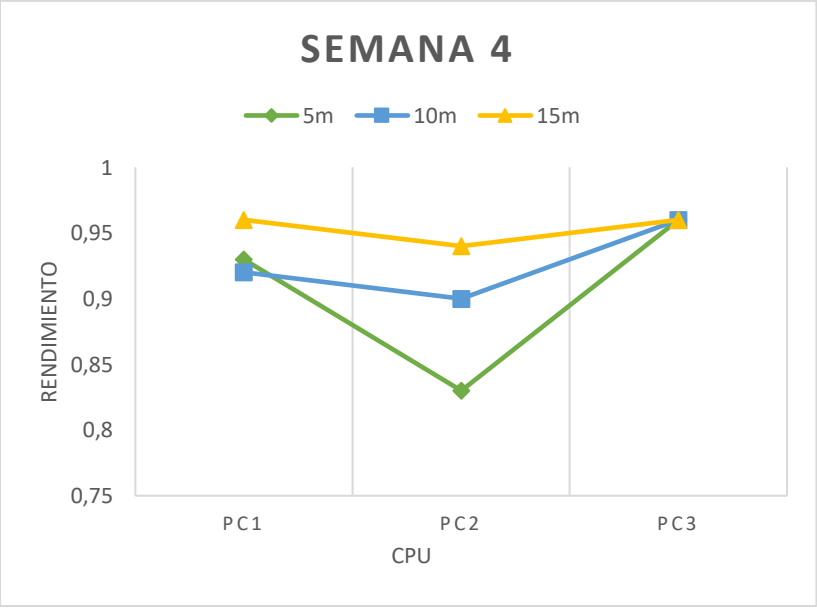
Rendimiento de la CPU, semana 3.



Nota. En la Semana 2 y Semana 3 la tendencia se mantiene, la PC3 tiene mayor rendimiento, pero la PC1 con características comunes a una PC que la mayoría tenemos en nuestros hogares, presenta una mejora.

Figura 91

Rendimiento de la CPU, semana 4.



Nota. En la semana 4 la PC1 y la PC3 se acercan un poco más en el rendimiento, concluyendo: En las 4 semanas no existió una brecha de rendimiento, a pesar de que las características de las PC que se describen sí representan cambios significativos a nivel de hardware y software.

3.5. Análisis de Resultados

Se elaboró un marco teórico con una revisión de la literatura existente sobre el tema, en las diferentes bases de datos bibliográficas, seleccionando artículos científicos publicados desde el 2018 – 2022 con un ranking de impacto en su mayoría Q1, Q2 y Q3, que tratan sobre machine learning y análisis de imágenes, logrando ampliar el conocimiento de los procesos a seguir, métodos a utilizar y qué herramientas seleccionar para desarrollar una aplicación que funcione en tiempo real.

Un equipo fácil de adquirir sin grandes especificaciones dio como resultado el entrenamiento del algoritmo con un tiempo aproximado de 15 horas, donde las características del computador fueron fundamentales, la PC constó de un procesador Intel core i7 con una memoria ram de 8GB y un sistema operativo Windows 10. En la detección, se utilizó una cámara de 12 mpx de un teléfono móvil de marca Apple en la versión 8 con el sistema operativo IOS 15.5.1. El desarrollo tuvo un diseño modular, siguiendo la metodología en cascada facilitando la independencia de funciones y la programación, se implementó un módulo no previsto, esto permitió realizar un conteo de plantas, que sirvió para presentar un mejor análisis del ground truth. Al realizar casos de prueba constituidos por los 3 módulos para evidenciar su funcionamiento, el módulo de adquisición y transmisión con el módulo de recepción y procesamiento de imágenes tienen una tasa de acoplamiento y eficiencia del 100%, cada imagen es transmitida y recibida en tiempo real perfectamente siguiendo el protocolo RTSP, la aplicación “Camo” en su versión libre cumple su función correctamente.

Un conjunto de 4200 imágenes capturadas en tiempo real a 5, 10 y 15 metros de altura, de un cultivo cuyo tamaño fue evaluado en la semana 1, 2, 3 y 4 fueron analizadas, se seleccionó las tres alturas para evaluar la capacidad de detección e identificación. El algoritmo tiene una tasa de detección de precisión de líneas de cultivo del 92,10%, la tasa de detección de malezas presento una exactitud del porcentaje 80,49%. Realizando un promedio se tiene una precisión y exactitud del software en un 86,28% lo que lo hace un software funcional que

cumple con el objetivo propuesto. El sistema de detección en tiempo real tuvo mejores resultados en la semana 3 con un 91% de acierto, tanto en la detección de líneas de cultivo como en la de malezas, el cultivo se encuentra perfectamente identificable para el ojo humano y el software, por lo que la capacidad de cómputo en velocidad, precisión y exactitud es la más alta.

Realizando una comparativa entre el presente sistema en tiempo real con otros sistemas analizados en el capítulo 1, se tiene que las limitaciones del algoritmo realizado se basan en la selección de características porque un algoritmo con un mejor análisis de detección de características como el algoritmo implementado en el proyecto MExG puede alcanzar una precisión del 97,3%. El análisis detallado de las bandas espectrales, así como el uso de cámaras hiperespectrales en UAVs puede facilitar una detección más precisa, ya que en este proyecto se analiza la escala y la cercanía de los objetos limitándose a cerrar un área en una forma específica más no a establecer un contorno de lo identificado. Los métodos tradicionales de aprendizaje automático como los de aprendizaje supervisado, presentan una desventaja a los métodos de aprendizaje profundo como las redes neuronales.

El costo computacional del software se midió mediante el rendimiento que presenta la CPU en tres características: el tiempo, número de instrucciones y tiempo por instrucción. Al evaluar estos parámetros en 3 CPU con características diferentes, no existe una brecha en el rendimiento del algoritmo de una CPU con otra, el rendimiento siempre es mayor al 0,92 siendo 1 el perfecto, la diferencia es de segundos, pero la detección se realizará con éxito.

CONCLUSIONES

El proyecto presenta una aplicación para detección automática de líneas de cultivo y malezas mediante el procesamiento de imágenes agrícolas en tiempo real, utilizando Haar Cascade en el entrenamiento del algoritmo, el cual fue realizado mediante el lenguaje Python y la librería de visión artificial de código abierto OPENCV.

El sistema fue construido con el hardware y software que se tenía a la mano poniendo a prueba los recursos y el acoplamiento de distintos dispositivos con distintos sistemas operativos, ya que el del móvil es distinto al de la PC y los dos dependen de una aplicación libre de uso, siendo así un proyecto factible y escalable ya sea a una implementación permanente en un cultivo o un desarrollo más profundo en la precisión que este tiene.

Comparando los lenguajes de programación más utilizados en machine learning R, Julia, Python y Matlab, Python fue el seleccionado debido a que se partió de la premisa que al construir un modelo de detección de datos es necesario que el lenguaje sea de alto nivel, tipado dinámico, lenguaje interpretado, portable y fiable, Python cumple con todas las características y además, que es un lenguaje fácil de aprender debido a la comunidad de soporte que existe a nivel mundial.

El sistema quiere ayudar a la automatización de tareas agrícolas en tiempo real, reduciendo tiempo, mano de obra y recursos económicos, además de controlar la contaminación ambiental con el uso de pesticidas como herbicidas y fungicidas que son los más utilizados en nuestro medio y generalmente son tan dañinos, pero que no son prohibidos en el país. La detección de malezas mejoró significativamente las semanas tres y cuatro, el cultivo contaba ya con un tamaño considerable para su detección, además, el algoritmo presentó fallas cuando la presencia de la maleza abarcó un área extensa de terreno confundiendo el verde de la maleza con el verde de las plantas de cultivo.

RECOMENDACIONES

La aplicación tiene un funcionamiento óptimo si el hardware no es cambiado, sobre todo la cámara, debido a que las configuraciones son específicas y para realizar un cambio se debe tener en cuenta los parámetros de detección en el software: calidad, tamaño, posición, etc.

Se recomienda verificar los módulos de adquisición y transmisión de video, se debe comprobar la transmisión de la cámara, además de vincular los dispositivos y revisar si el servicio de streaming está funcionando ya sea por HTTP o RTSP, para un mayor desempeño del algoritmo se debe obtener las imágenes con una cámara de alta resolución con procesadores de última generación.

Para mejorar su desempeño, ahorrando tiempo en lo que a implementarlos y entrenarlos se refiere, se recomienda elegir la mejor ubicación de la cámara para optimizar la detección, esta debería estar lo más perpendicular posible al cultivo para eliminar el efecto sombra y obtener hileras más rectas y malezas claras.

Con el objetivo de mejorar el sistema, es recomendable comprobar la detección en cultivos de más de 4 semanas y a otras distancias de la cámara, es decir, imágenes en tiempo real de cultivos susceptibles de generar falsos positivos o falsos negativos, y si existe un empeoramiento de los resultados puede plantearse un reentrenamiento de los tipos de datos.

REFERENCIAS

Ahmad, A., Guyonneau, R., MERCIER, F., & BELIN, E. (2018). *AN image processing method based on features selection for crop and weeds discrimination using RGB images | Request PDF*. https://www.researchgate.net/publication/325846856_AN_image_processing_method_based_on_features_selection_for_crop_and_weeds_discrimination_using_RGB_images

Alfons Crespo, & Alejando Alonso. (2006). Una Panorámica de los Sistemas de Tiempo Real. *RIAI*.

Amin Ahmadi. (2017). *Cascade Trainer GUI - Amin*. https://amin-ahmadi.com/cascade-trainer-gui/

Basso, M., & Pignaton de Freitas, E. (2020). A UAV Guidance System Using Crop Row Detection and Line Follower Algorithms. *Journal of Intelligent and Robotic Systems: Theory and Applications*, *97*(3–4), 605–621. https://doi.org/10.1007/S10846-019-01006-0

Comba, L., Biglia, A., Ricauda Aimonino, D., & Gay, P. (2018). Unsupervised detection of vineyards by 3D point-cloud UAV photogrammetry for precision agriculture. *Computers and Electronics in Agriculture*, *155*, 84–95. https://doi.org/10.1016/J.COMPAG.2018.10.005

Czymmek, V., Harders, L. O., Knoll, F. J., & Hussmann, S. (2019). Vision-based deep learning approach for real-time detection of weeds in organic farming. *I2MTC 2019 - 2019 IEEE International Instrumentation and Measurement Technology Conference, Proceedings*, *2019-May* 1–5. https://doi.org/10.1109/I2MTC.2019.8826921

Danilo García Santillán, I. (2018). *Métodos de visión por computador para detección automática de líneas de cultivo curvas/rectas y malas hierbas en campos de maíz*.

Elizabeth Rodriguez. (2017). *ANALYSIS AND IMPLEMENTATION OF THE VIOLA-JONES FACE DETECTION ALGORITHM*.

García-Santillán, I. D., & Pajares, G. (2018). On-line crop/weed discrimination through the Mahalanobis distance from images in maize fields. *Biosystems Engineering*, *166*, 28–43. https://doi.org/10.1016/J.BIOSYSTEMSENG.2017.11.003

Ib, R., Journal, I., & Systems, I. (2019). *Nº E19*.

Jesus Demetrio Velázquez Camacho. (2012, November 12). *Desarrollo en Cascada (Waterfall) VS Desarrollo Agile (SCRUM) - Northware*. https://www.northware.mx/blog/desarrollo-en-cascada-waterfall-vs-desarrollo-agile-scrum/

Jimenez, A., & Jiménez, F. (2013). *PROCESAMIENTO DIGITAL DE IMÁGENES DE SENSORES REMOTOS PARA APLICACIONES DE AGRICULTURA DE PRECISIÓN*. https://www.researchgate.net/publication/309374952_PROCESAMIENTO_DIGITAL_DE_IMAGENES_DE_SENSORES_REMOTOS_PARA_APLICACIONES_DE_AGRICULTURA_DE_PRECISION

Jimenez, A., Jiménez L., F., & García Ramírez, D. (2015). *Software para el estudio de coberturas vegetales con conceptos de agricultura de precisión*. https://www.researchgate.net/publication/309374703_Software_para_el_estudio_de_coberturas_vegetales_con_conceptos_de_agricultura_de_precision

Kerkech, M., Hafiane, A., & Canals, R. (2018). Deep leaning approach with colorimetric spaces and vegetation indices for vine diseases detection in UAV images. *Computers and Electronics in Agriculture*, *155*(October), 237–243. https://doi.org/10.1016/j.compag.2018.10.006

Li, Y., Qian, M., Liu, P., Cai, Q., Li, X., Guo, J., Yan, H., Yu, F., Yuan, K., Yu, J., Qin, L., Liu, H., Wu, W., Xiao, P., & Zhou, Z. (2019). The recognition of rice images by UAV based on capsule network. *Cluster Computing*, *22*, 9515–9524. https://doi.org/10.1007/S10586-018-2482-7

Mozaffari, M., Saad, W., Bennis, M., Nam, Y. H., & Debbah, M. (2019). A Tutorial on UAVs for

Wireless Networks: Applications, Challenges, and Open Problems. *IEEE Communications Surveys and Tutorials*, *21*(3), 2334–2360. https://doi.org/10.1109/COMST.2019.2902862

Mutchima, P., & Sanguansat, P. (2010). A novel approach for measuring video similarity without threshold and its application in sports video categorization. *Proceedings - 2010 1st International Conference on Pervasive Computing, Signal Processing and Applications, PCSPA 2010*, 868–872. https://doi.org/10.1109/PCSPA.2010.215

Oliveira, H. C., Guizilini, V. C., Nunes, I. P., & Souza, J. R. (2018). Failure Detection in Row Crops from UAV Images Using Morphological Operators. *IEEE Geoscience and Remote Sensing Letters*, *15*(7), 991–995. https://doi.org/10.1109/LGRS.2018.2819944

ONU. (2015). *Infraestructura – Desarrollo Sostenible*. https://www.un.org/sustainabledevelopment/es/infrastructure/

Pusdá-Chulde, M. R., Salazar-Fierro, F. A., Sandoval-Pillajo, L., Herrera-Granda, E. P., García-Santillán, I. D., & De Giusti, A. (2020). Image Analysis Based on Heterogeneous Architectures for Precision Agriculture: A Systematic Literature Review. In *Advances in Intelligent Systems and Computing* (Vol. 1078, pp. 51–70). Springer. https://doi.org/10.1007/978-3-030-33614-1_4

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). *You Only Look Once: Unified, Real-Time Object Detection*. http://pjreddie.com/yolo/

Reinoso, O., García, L. M., Jiménez García, L., & Payá Castelló, Arturo Gil Aparicio, A. P. (2018). *MATLAB: conceptos básicos y descripción gráfica*. https://books.google.es/books?hl=es&lr=&id=ioVxDwAAQBAJ&oi=fnd&pg=PR8&dq=que+es+matlab&ots=G3h4u3lsQo&sig=r5HSxdNiT3Y1uDPhBws0n5fvNRY#v=onepage&q=que es matlab&f=false

Ribeiro, S. R. G. Á., Alberto, S., & Barreda, L. J. (2009). *Seguimiento visual de líneas de cultivo*.

Rocha, A., Associação Ibérica de Sistemas e Tecnologias de Informação, & Institute of Electrical and Electronics Engineers. (2019). *Object detection application and Viola Jones algorithm for the development of a database in Alzheimer's patients*.

Rojas, E. M. (2020). *Machine Learning: análisis de lenguajes de programación y herramientas para desarrollo*.

Schulzrinne, H. (1998). *Network Working Group*.

Seijas, R. G. Á. R., Alberto, & Luis Barreda Sánchez, J. (2009). *Seguimiento visual de líneas de cultivo*.

Shah, J. P., Prajapati, H. B., & Dabhi, V. K. (2016). A survey on detection and classification of rice plant diseases. *2016 IEEE International Conference on Current Trends in Advanced Computing, ICCTAC 2016*. https://doi.org/10.1109/ICCTAC.2016.7567333

Shi, Z., Dang, H., Liu, Z., & Zhou, X. (2020). Detection and identification of stored-grain insects using deep learning: A more effective neural network. *IEEE Access*, *8*, 163703–163714. https://doi.org/10.1109/ACCESS.2020.3021830

Tiwari, P. S., Sahni, R. K., Kumar, S. P., Kumar, V., & Chandel, N. S. (2019). *Precision agriculture application in horticulture*. https://www.researchgate.net/publication/341079105_Precision_agriculture_application_in_horticulture

Tran, H., Dong, C., Naghedolfeizi, M., & Zeng, X. (2021). Using cross-examples in viola-jones algorithm for thermal face detection. *Proceedings of the 2021 ACMSE Conference - ACMSE 2021: The Annual ACM Southeast Conference*, 219–223. https://doi.org/10.1145/3409334.3452083

Viera, G. (2018). Aplicación de procesamiento de imágenes para clasificación de granos de cacao según su color interno. *Universidad de Piura*, 133. https://pirhua.udep.edu.pe/handle/11042/3486

Wu, Z., Chen, Y., Zhao, B., Kang, X., & Ding, Y. (2021). Review of weed detection methods based on computer vision. In *Sensors* (Vol. 21, Issue 11). MDPI AG.

https://doi.org/10.3390/s21113647
Yanez, C., Marcelo, R., Carlos, S., & Xavier, C. (2018). *Primer Congreso Internacional de Ciencia y tecnologia Agropecuaria "Fomentando la Seguridad y Soberanía Alimentaria."* Junio 13 - 14. https://repositorio.iniap.gob.ec/bitstream/41000/5064/1/iniapsc366p169-171.pdf
Yost, M. A., Kitchen, N. R., Sudduth, K. A., Sadler, E. J., Drummond, S. T., & Volkmann, M. R. (2017). Long-term impact of a precision agriculture system on grain crop production. *Precision Agriculture*, *18*(5), 823–842. https://doi.org/10.1007/S11119-016-9490-5