



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA



Universitat Politècnica de València

Departamento de Informática de Sistemas y Computadores

**Diseño de un Sistema para la Vigilancia de
Plantaciones Agrícolas
Basado en Fog Computing y Enlaces LoRa**

TRABAJO FIN DE MÁSTER

Máster en Ingeniería de Computadores y Redes

Autor:

Joel Lenin Quispe Vilca

Directores:

Dr. Pietro Manzoni

Dr. Jose Oliver

Agradecimientos

A Dios, tu amor y tu bondad me permiten sonreír ante todos mis logros que son resultado de tu voluntad que es buena, agradable y perfecta.

A mi familia por su motivación constante, por sus consejos y valores, por su comprensión y aliento en momentos difíciles. Cada cual siendo la mejor expresión del amor de familia que se ve reflejado en la vida de un hijo agradecido con Dios.

A mi novia Ely por ofrecerme su amor, su apoyo y paciencia que son instrumento idóneo mandado por Dios para hacer de mí un mejor Hombre de Bien y cada nuevo día la mejor versión de mí mismo.

Al PRONABEC Perú por apostar por mis virtudes e ímpetu de seguir aprendiendo que harán de mí un mejor profesional y persona con nuevas perspectivas para el desarrollo de mi país.

Para llegar a cumplir nuestras metas y acercarse al éxito se requiere estar en el momento adecuado; lugar adecuado: Universitat Politècnica de Valencia en España; y con las personas adecuadas: cada uno de nuestros profesores, amigos y compatriotas, arquetipos de nuestro perfil profesional.

Resumen

La Agricultura de Precision (AP) se basa en una ciencia conocida como sensado remoto, la cual consiste en obtener información de algún objeto, que sea de interés de investigación sin la necesidad de contacto físico. La Visión por Computador como herramienta de la AP y sus técnicas aplicadas a la agricultura consiste en la interpretación de imágenes digitales del área de cultivo que posteriormente serán analizadas, modificadas y almacenadas. Conceptos que hoy en día se integran en Internet of Things (IoT) con el objetivo de estar conectado y tener un control más eficiente e inteligente.

Este Trabajo de Fin de Máster tiene por objetivo integrar tecnologías, protocolos y conceptos como IoT, LoRa, MQTT, Visión por Computador, etc. Para lo cuál se diseña un prototipo que permite realizar la adquisición, procesamiento y transmisión inalámbrica de información extraída de una plaga denominada Cotonet a un servidor Cloud llamado The Things Network (TTN).

Como estrategia se ha desarrollado sistema de bajo costo y consumo compuesta por 3 etapas. El proceso empieza con la captura de imágenes, cuya muestra es obtenida de los árboles limoneros y de naranjos con una webcam bajo ciertas condiciones de luminosidad. Las imágenes son procesadas utilizando una Raspberry Pi y la librería OpenCV lo que permite un correcto análisis de los elementos y reconocimiento del objeto de interés que es la plaga Cotonet en un tiempo de respuesta tolerable para el escenario. Finalmente las imágenes son manipuladas e interpretadas reduciendo la imagen extraída a un valor o índice de afección porcentual de plaga, para luego transmitirlos por la plataforma TTN utilizando la tecnología LoRa y almacenar dicha información en el servidor TTN. En particular este TFM utilizará las estrategia de IoT llamada Fog Computing y la tecnología de red inalámbrica LoRa.

Abstract

The precision farming is based on a science known as "remote sensing", which consists in obtaining information about an interest object in the investigation without the need for physical contact. The Computer Vision as a tool for the precision farming and its techniques applied to agriculture consists in the interpretation of digital images of the cultivation area that are later analyzed, modified and stored. Concepts that nowadays are integrated in IoT in order to be connected and to have a more efficient and intelligent control.

This work aims to integrate technologies, protocols and concepts such as IoT, LoRa, MQTT, Computer Vision, etc. For which a prototype is designed that allows the acquisition, processing and wireless transmission of information extracted from a plague called Cotonet to a Cloud server called TTN.

As a strategy, a low cost and consumption system composed of 3 stages has been developed. The process begins with the capture of images, whose sample is obtained from the lemon and orange trees with a webcam under certain lighting conditions. The images are processed using a Raspberry Pi and the OpenCV library which allows a correct analysis of the elements and recognition of the object of interest that is the Cotonet pest in a tolerable response time for the scenario. Finally, the images are manipulated and interpreted by reducing the extracted image to a value or index of percentage pest affection, to then transmit them through the "The Thigs Network" platform using the LoRa technology and store said information in the TTN server. In particular, this TFM uses the IoT strategy called "Fog Computing" and wireless networks technologies like LoRa.

Índice general

Índice de figuras	iii
Índice de tablas	v
1 Introducción	1
1.1 Motivación	2
1.2 Objetivos	2
1.2.1 Objetivo General	3
1.2.2 Objetivos Específicos	3
1.3 Metodología	3
1.4 Estructura	4
2 Aspectos Generales	5
2.1 Introducción	5
2.2 Agricultura de Precisión	6
2.3 Internet de las Cosas	8
2.3.1 Arquitectura IoT	9
2.3.2 Fog Computing	11
2.4 Tecnologías de Comunicación	12
2.5 LoRa/LoRaWAN	13
2.5.1 Especificaciones LoRaWAN	15
2.5.2 The Thing Network	18
2.6 Protocolo de Comunicación MQTT	19
2.7 Visión por Computador	20
2.7.1 Elementos de visión por computador	21
2.8 Procesamiento de Imágenes	24
2.8.1 Introducción	24
2.8.2 Modelos de Color	25
2.8.3 Reconocimiento de Patrones	28
2.8.4 Operaciones Morfológicas	30

3	Caso de Estudio y Propuesta del Sistema	34
3.1	Caso de Estudio	34
3.1.1	El Cotonet <i>Planococcus citri</i> (Risso)	35
3.2	Arquitectura Propuesta	36
3.2.1	Hardware Empleado en el Sistema	37
3.2.2	Software Empleado en el Sistema	41
4	Procedimiento Detección y Comunicación	45
4.1	Condiciones de Luminosidad	45
4.2	Procesos de Detección	46
4.2.1	Adquisición de Imágenes	47
4.2.2	Procesado y Segmentación	49
4.2.3	Extracción de Características	55
4.3	Infraestructura de Red TTN	56
4.4	Procesos de Enlace LoRaWAN	57
4.4.1	Aplicación en la consola TTN	58
4.4.2	Gateway	60
4.4.3	Nodo	60
5	Resultados	65
5.1	Conexión a Red	65
5.2	Tratamiento de Imágenes	67
5.3	Transmisión de Mensajes	74
5.4	Prototipo	75
6	Conclusiones y Trabajo Futuro	76
6.1	Conclusiones	76
6.2	Trabajos Futuros	78
	Lista de Acrónimos	80
	Bibliografía	81

Índice de figuras

2.1	Agricultura Inteligente [1]	6
2.2	Agricultura de Precisión	7
2.3	Componentes IoT	9
2.4	Arquitectura de Internet de las Cosas	10
2.5	El Internet de las cosas y el Fog Computing	12
2.6	Tecnologías Low-Power Wide-Area Network	13
2.7	Estructura de red de LoRaWAN	14
2.8	Clases de LoRaWAN	16
2.9	ESQUEMA DE ACTIVACIÓN DE NODOS MODO OTAA	17
2.10	Esquema de Activación de nodos modo Activation By Personalization. (ABP)	18
2.11	Esquema de Funcionamiento MQTT	19
2.12	Cámaras Digitales	22
2.13	Esquema de un sensor CCD	23
2.14	Esquema de un sensor CMOS	24
2.15	Etapas de un Sistema Visión por Computador	25
2.16	Esquema del modelo de color RGB	26
2.17	Esquema del modelo de color HSV	27
2.18	Umbralización	29
2.19	Operaciones de Erosión y Dilatación	32
2.20	Operación Morfológica de Apertura	32
2.21	Operación Morfológica de Cierre	33
3.1	Entorno de Trabajo	35
3.2	Planococcus citri	36
3.3	Arquitectura de red Propuesta	37
3.4	Componentes de una Raspberry Pi 3 [2]	38
3.5	Microcontrolador LoPy de Pycom	39
3.6	Dispositivos Pycom para la Comunicación LoRa	40
3.7	Cámara del Sistema	41
3.8	Logo de MycroPython	43

3.9	Logo de Librería OpenCV	44
4.1	Diagrama de flujo del Sistema	46
4.2	Adquisición de Imágenes	49
4.3	Proceso de Segmentación y Filtrado	50
4.4	Comparación de imágenes con el filtro Dilatación	51
4.5	Comparación de imágenes con el segundo filtro GaussBlur	52
4.6	Segmentación de la imagen mediante el espacio de colores Hue Saturation Vue (HSV)	53
4.7	Segmentación de imágenes con retorno al espacio de colores Red Green Blue (RGB) aplicando el operador AND.	54
4.8	Segmentación de imágenes en el espacio de colores RGB y ex- tracción del objeto de interés.	55
4.9	Representación de la imagen procesada por visión por computador	55
4.10	Arquitectura de Comunicaciones del Sistema sobre la Plataforma TTN	56
4.11	Diagrama de flujo del Bloque de Transmisión del Sistema	58
4.12	Creacion de una Aplicación en TTN	59
4.13	Generación de claves en la Aplicación TTN	59
4.14	Gateway Basado en Raspberry en Operación	60
4.15	Parámetros de configuración en módulo de radio LoRa	61
4.16	Parámetros MQTT para LoPy	61
4.17	Conexión a Red WIFI y Registro del Dispositivo Lopy	62
4.18	Configuración de Parametros Publisher MQTT	63
4.19	Activación de Mosquitto y espera de Mensajes MQTT	64
5.1	Aplicación TTN en modo espera	65
5.2	Conexión OTAA y MQTT	66
5.3	Imagen mal capturadas	68
5.4	Imagen bien capturada y dentro de las condiciones de luminosidad .	68
5.5	Procesamiento de Imágenes del Sistema	71
5.6	Llegada de datos a la Aplicación TTN	75
5.7	PROTOTIPO FINAL	75

Índice de tablas

5.1	Tabla de Datos de Conexión a la Red LoRaWAN.	67
5.2	Tabla de Datos de Adquisición de Imágenes del Sistema.	70
5.3	Tabla de Datos de Segmentación de Imágenes del Sistema.	73
5.4	Tabla de Tiempo de Envío de Mensajes por LoRaWAN.	74

Capítulo 1

Introducción

La agricultura es un sector de alta importancia para la vida cotidiana de las personas ya que su producción influye en lo social y económico, lo cual genera un impacto tanto en el aspecto económico como en el nivel alimenticio de sus consumidores. Esta actividad requiere de métodos tecnológicos eficientes de control para generar productos de alta calidad.

Actualmente debido al constante avance de la ciencia en el campo del procesamiento digital, surgen nuevas aplicaciones que aumentan la eficiencia de los procesos, sin embargo, muchas de estas técnicas siguen evolucionando y algunas resultan obsoletas para el sistema de producción agrícola.

La metodología actual se basa en técnicas artesanales que generan una producción agrícola poco eficiente. El control del campo de cultivo es sumamente limitado ya que estos métodos están orientados a analizar las propiedades de interés de manera superficial, lo cual se refleja en acciones correctivas mal enfocadas al campo. Esta práctica genera un consumo excesivo de pesticidas, fertilizantes y recursos hidrológicos, así como efectos negativos en el cultivo.

Uno de los proyectos en desarrollo es la agricultura de precisión, la cual se refiere al monitoreo y control electrónico aplicado a la recolección de información y su procesamiento como soporte para la toma de decisión y para el uso espacial y temporal de insumos en la producción de cultivos y que actualmente es utilizado en varios países desarrollados. A su vez sumados las tecnologías de comunicación a la agricultura de precisión se pretende aprovechar y potenciar más estos beneficios y denominar una nueva propuesta como Agricultura Inteligente.

Debido a ésta problemática y nuevas soluciones, se propone un sistema inteligente capaz de detectar un índice de propagación de la plaga *Planococcus citri*

en los árboles de frutos cítricos mediante técnicas de Visión Artificial y enviar dicha información mediante la tecnología inalámbrica LoRa en tiempo real a una aplicación en el que tendremos acceso con Internet.

Para lograr el objetivo, se analizarán resultados como el porcentaje de índice de plaga en los árboles frutales y los aspectos que están relacionados a éste, como la luminosidad, calidad de las imágenes y tiempo de respuesta del sistema.

1.1 Motivación

Muchos son los avances y los esfuerzos tecnológicos que se han realizado enfocados al área de la agricultura de precisión y muchas las soluciones de las que hoy disponen empresas y agricultores para sacar el máximo partido de una explotación. Sin embargo ahora se abre un mundo de nuevas posibilidades para el sector. Ha surgido la oportunidad de conectar todas estas soluciones a internet. La aparición de nuevos sensores, actuadores, sistemas embebidos y el abaratamiento del acceso a nuevas tecnologías están favoreciendo que se dote todos estos dispositivos y herramientas con la capacidad de conexión a la red.

La informalidad de la actividad agrícola en muchos países y la falta de recursos tecnológicos obstaculizan la implementación de nuevas técnicas capaces de utilizar de manera más eficiente los recursos e insumos y mejorar la calidad de la producción. De la misma forma, la falta de información acerca de nuevas técnicas de supervisión contribuye a la aplicación de técnicas obsoletas.

Gran parte de líneas de investigación en agricultura de precisión están orientados a visión por computador, y en su mayoría los sistemas con la capacidad sensorial de la visión se complementan con otros mecanismos sensoriales tales como sensores de alcance o proximidad, temperatura, etc, para después completar el proceso de transporte de la información procesada; en donde las redes de comunicación inalámbrica de bajo consumo y largo alcance van ganando adeptos al ser una tecnología madura y confiable.

1.2 Objetivos

Este Trabajo de Fin de Máster tiene como objetivos:

1.2.1 Objetivo General

- Diseñar un prototipo IoT basado en tecnología LoRa y Visión por Computador de bajo coste y consumo y largo alcance capaz de detectar un índice de afección de la plaga Cottonet.

1.2.2 Objetivos Específicos

- Determinar el índice de infestación de la plaga *Planococcus citri* en plantaciones agrícolas técnicas de Visión por Computador.
- Desarrollar un registro de imágenes basado en algoritmos que permitan un nivel de eficiencia y fidelidad acorde con la aplicación del sistema.
- Estudiar y analizar las estrategias IoT llamadas LoRaWAN y tecnología de red inalámbrica LoRa.
- Aplicar las estrategias IoT llamadas Fog Computing.
- Analizar y validar los resultados de las soluciones propuestas.

1.3 Metodología

De acuerdo a los objetivos propuestos se tiende a elegir una planificación que se especifica junto a la estrategia a seguir.

- En la primera etapa se plantea el estudio de las diferentes herramientas de trabajo como son el lenguaje de programación, librerías que hacen referencia al software de trabajo, el minicomputador Raspberry Pi para aplicar las técnicas de tratamiento de imágenes y el microcontrolador Lopy para aplicar la comunicación inalámbrica LoRa.
- La segunda etapa se realizará un análisis del Diseño del Sistema orientado a la agricultura de precisión donde se escogerá un campo de cultivo o muestras de áreas de cultivo y se procesará dicha data.
- La tercera etapa se analiza los parámetros de comunicación que están relacionados a los protocolos LoRaWAN y MQTT y la arquitectura de red TTN para integrar al sistema de visión.
- Posteriormente pasar a una etapa de Evaluación del Sistema de Registro de Imágenes y Transporte de Comunicación y sus diferentes parámetros y protocolos involucrados (IoT, MQTT, Fog Computing, LoRa) que indicaran las ventajas del sistema.

1.4 Estructura

La memoria de la tesis se presentará estructurada en capítulos. Estos capítulos se organizan siguiendo el orden natural de la investigación, cuya distribución es la que se ofrece a continuación:

Primer capítulo. Introducción, donde se estudian y plantean los objetivos de la tesis, así como la motivación e introducción de las propuestas y planteamientos realizados.

Segundo capítulo. Aspectos Generales del Procesamiento de imágenes en agricultura de precisión basado en Internet de las Cosas, donde se describirán, desde un punto de vista teórico, los métodos necesarios para el tratamiento de imágenes agrícolas y estrategias necesarias para la integración del sistema de comunicaciones o comúnmente denominado transporte de datos que forman parte de la investigación realizada en este trabajo. También sirve como introducción a los sucesivos capítulos donde se facilitan información nueva a las investigaciones ya existentes.

Tercer capítulo. Diseño del sistema de registro de imágenes y el transporte, donde se describirán, las consideraciones del lugar de trabajo, el hardware y software propuesto. Así mismo la arquitectura de red propuesta que engloba a los componentes seleccionados.

Cuarto capítulo. Procedimiento de detección de la plaga *Planococcus citri* mediante técnicas de visión por computador donde se detallan los pasos a seguir para segmentar y la transmisión por LoRaWAN, en el que se detallará que procedimientos se realizó para unir la parte de comunicación al bloque analizador de plaga.

Quinto capítulo. Evaluación del sistema, donde se analizará la respuesta del sistema bajo diferentes muestras, así como también el tiempo de requerido del procesamiento y transporte de la información de dicho sistema.

Sexto capítulo. Se presentarán las conclusiones, discusiones y futuras líneas de investigación y la importancia de los sistemas de visión artificial y la tendencia las comunicaciones sobre ésta.

Capítulo 2

Aspectos Generales

En esta sección se presentan algunos conceptos y métodos como antecedentes para la identificación y representación de un objeto de interés para su posterior comunicación de dicha información extraída, basada en investigaciones y pruebas realizadas mediante la visión computacional, añadiendo la parte de comunicación para completar el sistema y así denominarla aplicaciones en agricultura inteligente, tal y como lo plantea Santiago Rodríguez [3] “Para aumentar la eficiencia de las tareas e insumos, es indispensable el aporte de los nuevos elementos desarrollados por la avanzada tecnología de las Telecomunicaciones, incorporadas a la Agricultura de Precisión“. Las Telecomunicaciones nos brindan todos los medios requeridos por los cuales podemos transferir información muy variada. La transmisión de datos e información por medio de las Redes Integradas de Datos se vuelve crucial e indispensable para este tipo de aplicaciones [4].

2.1 Introducción

Analizar datos, incorporar sensores para aumentar el rendimiento por hectárea, optimizar procesos, mejorar la calidad del producto final y ahorrar tiempo, así como predecir fallos en máquinas e, incluso, controlar plagas son solo algunas de las ventajas que ofrece a la agricultura, el Internet de las Cosas o Internet de los objetos.

Aunque es uno de los términos más populares de los últimos años, la definición del Internet de las Cosas (IoT) todavía genera algunas dudas. Se trata de usar la interconexión digital de objetos cotidianos con acceso a Internet, a menudo sin la necesidad de mucha intervención humana, para que se comuniquen entre sí y, por lo tanto, sean más inteligentes e independientes. Nadie duda de su amplia demanda, según una encuesta pasada de la consultora de investigación Gartner [5]

prevé que 8,4 mil millones de artículos estuvieron conectados en 2017, lo que supone un 31 por ciento más que en 2016, y llegará a los 20.4 mil millones en 2020.



Figura 2.1: Agricultura Inteligente [1]

2.2 Agricultura de Precisión

La AP es el análisis de los elementos que componen el campo de cultivo y tiene como objetivo conocer el estado de la vegetación. Su idea básica es definir el estado del cultivo en determinadas áreas y en instantes dados, brindando así información vital para controlar sus propiedades y llevar el cultivo a estados óptimos.

Este concepto se basa en instrumentos de medición y procesamiento capaces de cuantificar las propiedades de interés para desarrollar un sistema capaz de monitorizarlas y analizar su variabilidad en el tiempo. La información obtenida permite aplicar tratamientos específicos de acuerdo con el estado de las zonas de cultivo analizadas y optimizar el proceso de producción [6].



Figura 2.2: Agricultura de Precisión

Jose Miguel Guerrero de Ingeniería Del Software E Inteligencia Artificial Madrid desarrolló un sistema de visión artificial enfocado a la identificación y localización de malas hierbas en campos de maíz, el control del guiado del vehículo y el control del solapamiento de las zonas tratadas cuya contribución principal de este trabajo de investigación radica en el diseño de un sistema experto para AP capaz de detectar las líneas de cultivo y malas hierbas en campos de maíz además de controlar el guiado del tractor y el solapamiento de las zonas tratadas en tiempo real [7].

Jorge Enrique Barba investigó una tesis basada desarrollar un sistema de registro de imágenes de tomas aéreas para agricultura de precisión y recomienda el uso de la biblioteca de visión por computadora OpenCV que ha sido sumamente útil ya que posee funciones para muchas aplicaciones por que mejoró considerablemente el tiempo de procesamiento con respecto a su diseño, el cual llegaba a ser tedioso el uso, e incluso ineficiente, esto debido al cambio del algoritmo por uno más veloz y además al estar programado por un lenguaje de programación de menor nivel, como es el caso del Lenguaje C [8].

Renán Alfredo Rojas también realizó investigación y aplicación de este concepto de AP en su tesis con consiste en Diseñar y desarrollar un sistema de registro de imágenes multi-espectrales enfocado en el análisis de campos de cultivo. El sistema tiene como propósito relacionar geométricamente múltiples imágenes aéreas que abarquen al campo de interés, de tal forma que permita desarrollar una

imagen global. Dicha imagen mostraría la composición del campo de cultivo en su totalidad y brindaría la capacidad [6].

Daniel Chora Garcia de Ecuador realiza una investigación de información de un sistema económicos basado en Raspberry y Arduino que consiste en dar a conocer proyectos que optimizan procesos agrícolas por medio del control de datos ambientales y la gestión de actividades que se involucren en las labores diarias del campo, registro de enfermedades, plagas y malezas; aprovechando los beneficios del hardware y software libre [9].

2.3 Internet de las Cosas

El termino IoT ha sido definido por muchos autores en muchas formas diferentes; definen Internet de las cosas, simplemente como una iteraccion entre lo físico y lo digital. El mundo digital interactúa con el mundo físico usando la plétora de sensores y actuadores. Otra definición define Internet de las Cosas como un paradigma en el que la capacidad de computo y comunicación están incrustadas en cualquier tipo de objeto concebible. Existe múltiples definiciones de IoT, que abarcan pequeños sistemas localizados, restringidos a una ubicación específica hasta un gran sistema global distribuido, compuesto por varios sistemas complejos.

IoT, no es una nueva idea, Mark Weiser, director científico del Xerox Palo Alto Research Center, introdujo el concepto de computación ubicua, a principios de los 90. Weiser abogaba por un futuro en el que la computación formaría parte integral de nuestra vida diaria y resultaría transparente para nosotros. Weiser no acuñó el termino IoT, que se atribuye al Auto-ID Center del Massachusetts Institute of Technology tomado relevancia práctica gracias a la rápida evolución de la electrónica durante la última década.

El concepto “Internet de las Cosas” o “Internet de los Objetos” IoT, tiene una definición bastante amplia y difusa, pero puede resumirse como un paradigma tecnológico que define la dotación de conectividad a internet a cualquier objeto sobre el que se pueda medir parámetros físicos o actuar, así como las aplicaciones y tratamiento de datos inteligentes relativos a los mismos [10]. Esto incluye diferentes sistemas, como:

- Coches conectados a Internet.
- Wearables, incluyendo dispositivos de salud y fitness, relojes, y dispositivos implantados en humanos.

-
- Medidores y objetos inertes en general, que se dotan de inteligencia
 - Sistemas de automatización y control para inmuebles e iluminación
 - Smartphones
 - Redes de sensores inalámbricos que captan información del tiempo, barreras anti-inundaciones, mareas y muchas otras aplicaciones.

2.3.1 Arquitectura IoT

La arquitectura del “Internet de las Cosas” se puede estructurar conforme podemos ver en la figura (2.3), teniendo como áreas principales:

- Objetos conectados
- Tecnologías de red
- Protocolos de comunicación
- Plataforma IoT de tratamiento de datos
- Aplicaciones de usuario



Figura 2.3: Componentes IoT

Muchas de las propuestas de implementación de IoT se basan en los componentes de la figura (2.3). En la mayoría de los casos son aplicaciones de uso

específico que presentan diferencias entre ellas a nivel de objeto conectado e integración con la nube o plataforma IoT. De este modo, se diferencian tres tipos fundamentales de arquitectura según la aplicación:

- Arquitectura de tres niveles con objetos conectados sin protocolo IP
- Arquitectura de dos niveles con objetos conectados con protocolo IP
- Arquitectura de dos niveles con objetos conectados sin protocolo IP

El primer tipo de arquitectura se corresponde con los despliegues con radios de baja potencia que usan repetidores o pasarelas para conectarse a una red IP. El segundo tipo incorpora tecnología con conectividad directa sobre IP, como pueden ser WiFi o 2G. La tercera arquitectura usa nuevos protocolos de red específicos para IoT con su propia red de comunicación, sin usar el protocolo IP.

2.3.1.1 Arquitectura de dos niveles con objetos conectados sin protocolo IP

No existe alguna arquitectura oficial que represente a IoT sin embargo podemos mencionar una de las más mencionadas y asociadas al trabajo propuesto que es la arquitectura de 2 niveles ver Figura (2.4), con tendencia a nuevas tecnologías [11].

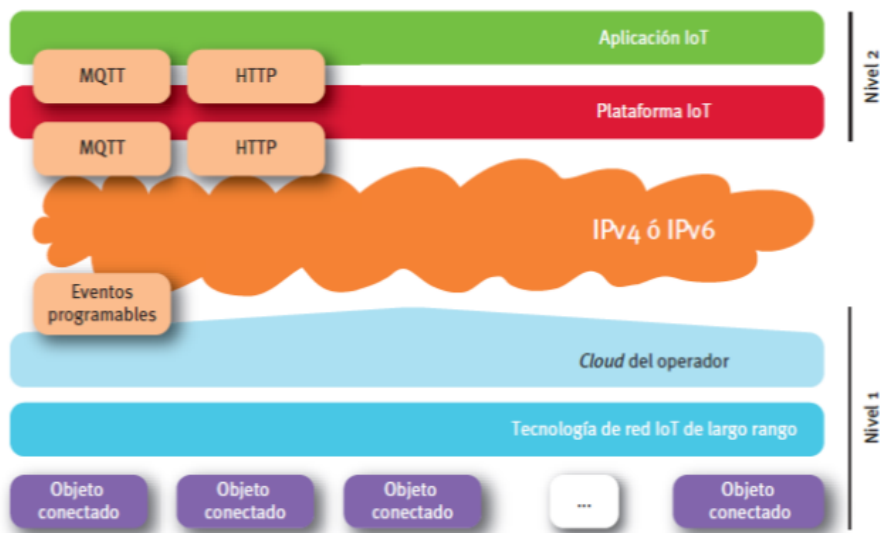


Figura 2.4: Arquitectura de Internet de las Cosas

En este tipo de arquitectura tenemos una estructura de dos niveles como en la figura 5, similar a los despliegues que encontramos en redes celulares.

Con la aparición del mundo del IoT los protocolos existentes han ido evolucionando para adaptarse a este nuevo concepto y, junto al desarrollo de nuevas soluciones técnicas, simplificar los despliegues, ganar cobertura de red y reducir el consumo energético y el coste de los objetos. Estas tecnologías se basan en protocolos propios no dependientes de IP, siendo capaces de ofrecer una conexión directa o casi directa entre el dispositivo y la plataforma.

De la mano de esta arquitectura surgen nuevos protocolos y operadores IoT que ofrecen una conexión directa. Entre ellos podemos encontrar soluciones como las de Sigfox, LoRa o Weighless, las cuales ofrecen objetos conectados de forma directa a la red, capacidad de crear dispositivos de bajo coste y larga duración de batería y un servicio de operador de comunicaciones, evitando que el usuario tenga que usar pasarelas o repetidores.

Además de las arquitecturas comentadas, existen en la literatura otras arquitecturas que recogen aspectos más concretos de IoT tal es el caso arquitecturas basadas en Cloud, Fog y Social IoT. A continuación se menciona una de éstas.

2.3.2 Fog Computing

Aunque el Fog computing tiene como punto de partida el cloud computing, es un modelo que se basa en concentrar datos y aplicaciones en los dispositivos al borde de la red. Los datos son procesados localmente en un dispositivo inteligente en lugar de ser enviados, sin procesar, a través de la red Fog, al igual que cloud, proporciona a los usuarios finales, datos, cálculo, almacenamiento y servicios. [12]

El Fog Computing es un modelo en el cual el procesamiento de los datos y las aplicaciones se concentran en los dispositivos al borde de la red, en lugar de completamente en la nube. De tal manera que los datos puedan ser procesados localmente en un dispositivo inteligente en lugar de ser enviados a la nube para ello.

Las principales características de Fog computing son:

- Los datos y las aplicaciones se sitúan en el borde de la red.
- Almacenamiento y computación distribuidos.
- Alta contribución geográfica de un gran número de nodos.

- Heterogeneidad de dispositivos, datos y aplicaciones.
- Comunicación en tiempo real y movilidad.

La Figura (2.5) presenta la información idealizada y la arquitectura computing que soporta las futuras aplicaciones de IoT, e ilustra el papel de Fog Computing.

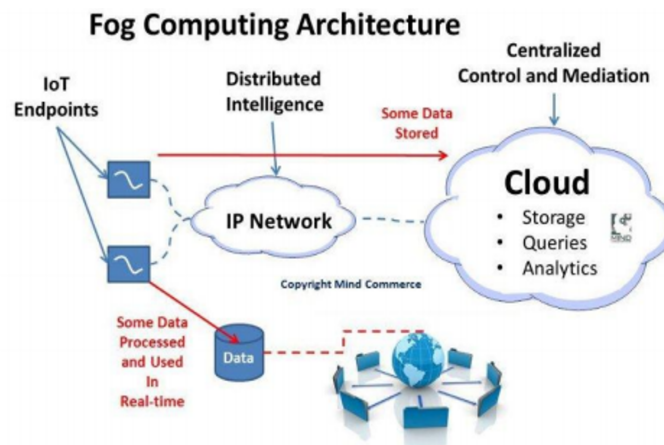


Figura 2.5: El Internet de las cosas y el Fog Computing

2.4 Tecnologías de Comunicación

La conectividad en el mundo IoT no tiene una única forma de implementación, puede permitir la conexión directa con la plataforma de gestión o necesitar el uso de repetidores o pasarelas. El primer caso lo representan, por un lado, las tecnologías tradicionales (WiFi, 2G, 3G o 4G), las cuales ofrecen una gran cobertura a costa de un alto consumo energético. Por otro lado, tenemos las nuevas soluciones de comunicación específicas para el IoT cuyas redes a día hoy, en la mayoría de las soluciones, no están altamente desplegadas, pero presentan importantes ventajas como un reducido consumo de energía, diseños baratos, altas coberturas y baja tasa de datos. Como parte de estas soluciones tenemos las denominadas “Low Power Wide Area Network” (Low Power Wide Area Network (LPWAN)) tales como Sigfox, **LoRa** o Weightless entre otras como se ve en la figura((2.6)), las cuales han ayudado al éxito inicial del IoT ya que han sido las primeras tecnologías usadas. Estas son poco eficaces en despliegues que requieren elevado tráfico de datos, ya que limitan operación de red al cliente final.

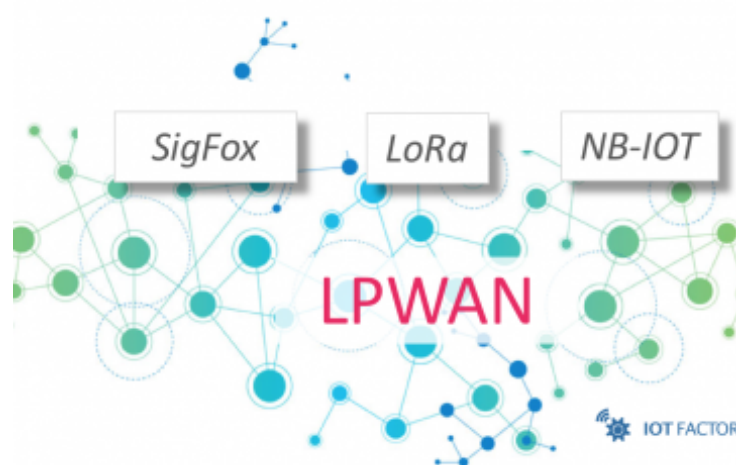


Figura 2.6: Tecnologías Low-Power Wide-Area Network

2.5 LoRa/LoRaWAN

LoRa es una tecnología de modulación de radio desarrollada por Semtech. Junto con Actility e IBM Reserch, Semtech crea la especificación LoRaMAC para construir redes de bajo consumo (Low Power Wide Area Network, LPWAN) enfocadas al IoT [13].

LoRaWAN(Long Range Wide Area Network) es un estándar de protocolo de nivel de enlace que utiliza el esquema de modulación LoRa. Este protocolo está optimizado para dispositivos alimentados por batería que pueden estar en movimiento o en un lugar de difícil acceso. Se compone de dos partes principalmente: gateways y nodos, los primeros son los encargados de recibir y enviar información a los nodos y los segundos, son los dispositivos finales que envían y reciben información hacia el gateway [14].

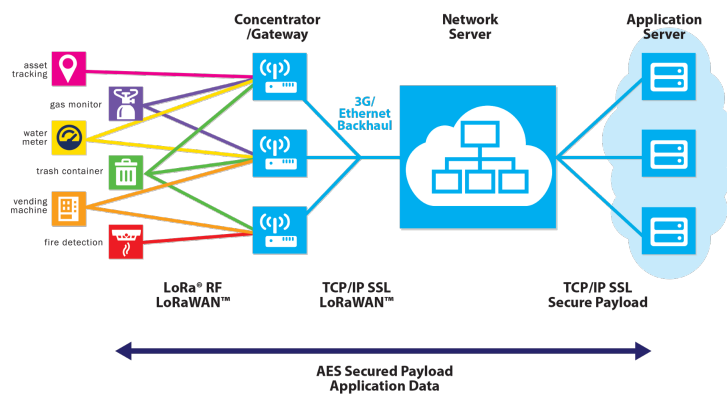


Figura 2.7: Estructura de red de LoRaWAN

En la imagen (2.7) podemos ver claramente cómo se compone una red LoRaWAN clásica, en la que una serie de dispositivos finales se conectan a Gateways y estos envían todo a un servidor, que por medio de una API entrega los datos a una aplicación final para el usuario.

Entre sus principales características de LoRaWAN son las siguientes:

- LoRa(Long Range) tiene un esquema de modulación de espectro expandido desarrollado por la LoRa Alliance, el cual permite un área grande de cobertura con un consumo de potencia reducido, a costa de un bajo bitrate [15]. Esta modulación utiliza las bandas de libre uso de 915MHz en América y 868 MHz en Europa, bandas ISM de uso libre iguales a las utilizadas por Sigfox.
- Presenta una topología de estrella con las gateways o concentradores enlazando los nodos finales con un servidor. La conexión gateway servidor es mediante IP y la conexión gateways nodos mediante LoRa.
- La comunicación es bidireccional.
- La comunicación entre nodos y gateways se da a través de diferentes frecuencias y tasas de datos. En Europa la banda de frecuencias es la ISM 863 - 870 MHz.
- La tasa de datos puede variar entre 0.3 kbps y 50 kbps y se puede ajustar para cada nodo según una fórmula de tasa de datos adaptativa ADR.
- El nodo puede transmitir a cualquier frecuencia disponible a la tasa de datos requerida pero respetando las reglas de máximo ciclo de trabajo, máxima duración de transmisión y los cambios pseudoaleatorios de canal que hace automáticamente para mejorar la robustez frente a interferencias.

2.5.1 Especificaciones LoRaWAN

Estas especificaciones mencionadas a continuación se pueden encontrar mas a detalle en [16].

2.5.1.1 Clases de Dispositivo

En modo LoRaWAN los nodos forzosamente se deben conectar a un gateway que soporta miles de nodos. Para poder unirse a la red y aprovechar las bondades del protocolo, el nodo debe enviar una serie de llaves de identificación y seguridad. Todos los nodos trabajan en un una conexión tipo estrella, los mismos nodos aun estando en movimiento se conectan al gateway más cercano y con mejor calidad de comunicación, muy similar a como funciona una red celular [17].

- **Clase A:** La más soportada en casi todos los dispositivos, este tipo de clase ofrece el mayor ahorro de energía debido a que solo entra en modo escucha (llamado ventana RX) después de enviar un datos hacia el gateway, por eso es ideal para dispositivos que usan una batería de poca vida.
- **Clase B:** Este tipo de dispositivos tiene las ventanas de recepción con base a tiempos predeterminados con el gateway, este tipo de nodos puede usar una batería o una fuente externa dependiendo de los tiempos asignados de escucha. Hasta el momento es poco usada.
- **Clase C:** Este tipo de clase ofrece el menor ahorro de energía debido a que siempre esta en modo escucha y solo cuando es necesario en modo transmitir, la recomendación es usarlo en dispositivos que cuentan con una fuente externa de alimentación.

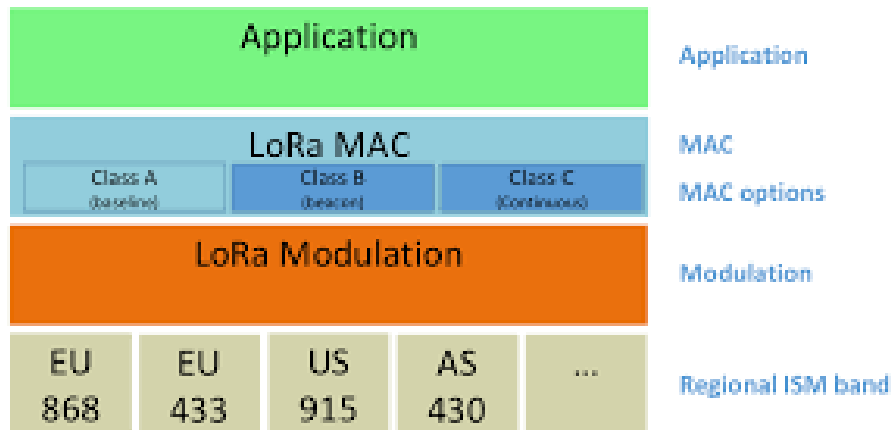


Figura 2.8: Clases de LoRaWAN

2.5.1.2 Conexión ABP y OTAA

Existen dos maneras de conectarse a una red LoRaWAN. Para lograr entrar a la red se requieren una serie de claves y número de identificación por parte del nodo para lograr el correcto funcionamiento de la red y mantener su seguridad.

Modo OTAA El modo Over The Air Activation (OTAA) (Over-The-Air-Activation) es la manera más segura de conectarse a la red. Los parámetros de configuración son:

- * **Identificador del dispositivo (DevEUI)** Es un identificador IEEE EUI64 único para cada nodo.
- * **Identificador de aplicación (AppEUI)** Identificador de aplicación único utilizado para agrupar objetos. Esta dirección de 64 bits se utiliza para clasificar los dispositivos por aplicación. Debe estar registrado en el dispositivo antes de la activación.
- * **Identificador de aplicación (AppKey)** Es una clave secreta AES de 128bits compartida entre el dispositivo periférico y la red. Se utiliza para determinar las claves de sesión.

LoRaWAN Over-The-Air Activation (OTAA)

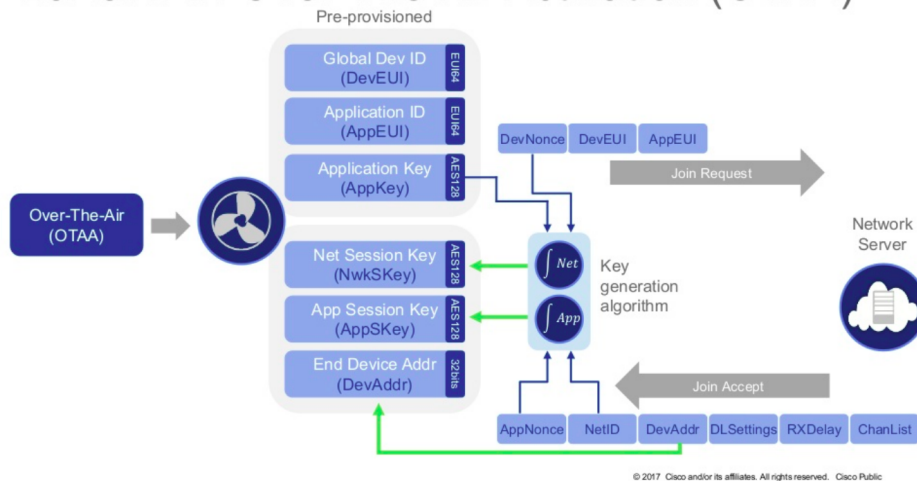


Figura 2.9: ESQUEMA DE ACTIVACIÓN DE NODOS MODO OTAA

La unión de activación aérea OTAA se basa en la identificación del nodo final y la aplicación de destino utilizando EUI (deveui y appeui) más una clave de cifrado que también se utiliza durante la negociación (Appkey). En la respuesta join accept, el servidor proporcionará una dirección de dispositivo (Devaddr) y tanto el nodo final como el servidor derivarán las claves de sesión (Nwkskey y Appskkey) para que nunca pasen por el aire. El resultado final de la unión OTAA es que el nodo final está configurado con los mismos tres parámetros esenciales que se utilizan para ABP (Devaddr, Nwkskey y Appskkey).

Modo ABP El modo ABP (Activation By Personalization) significa que las claves de cifrado se configuran manualmente en el dispositivo y pueden comenzar a enviar tramas a la puerta de enlace sin necesidad de un procedimiento de intercambio de información para intercambiar las claves (como la realizada durante un procedimiento de unión OTAA). Los parámetros de conexión son:

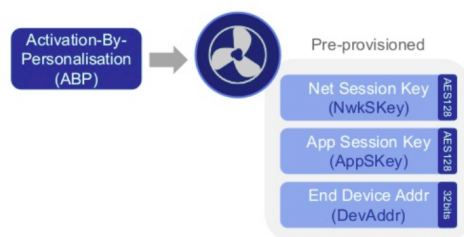
- * **Clave de sesión de red (NwkSKey)** Clave de cifrado entre el dispositivo y el operador utilizado para las transmisiones y para validar la integridad de los mensajes.
- * **Clave de sesión de aplicación (AppSKey)** Igual que NwkSKey, única para cada nodo. Se usa para encriptar los mensajes de la aplicación.
- * **Dirección del dispositivo (DevAddr)** Dirección lógica (equivalente a una dirección IP), consta de 32 bits. En los bits más significativos está la parte

del identificador de red (NwkID) usado para discriminar entre diferentes redes y en el resto la dirección de red (NwkAddr) del nodo, asignado por la red.

LoRaWAN Activation-By-Personalisation (ABP)

ABP pre-provisions keys and device address

Join procedure is bypassed



© 2017 Cisco and/or its affiliates. All rights reserved. Cisco Public

Figura 2.10: Esquema de Activación de nodos modo ABP

2.5.2 The Thing Network

TTN (The Things Network) es una red IoT abierta basada en tecnología de comunicación LoRaWAN sobre LoRa. TTN ha hecho un esfuerzo de desarrollo en la parte de servidor, construyendo todo el back-end de la red que da soporte a los gateways distribuidos por el globo. Un usuario que se conecta a la red puede registrar un Gateway; desde gateways comerciales hasta gateways hechos mediante una raspberry y un módulo de comunicaciones LoRaWAN por ejemplo.

TTN proporciona un back-end de servidores de aplicación que gestionan los gateways, y una plataforma que permite a los usuarios crear sus propias aplicaciones, asociar sus dispositivos y recibir los datos.

Además ofrece muchas formas de recibir la información de la red, desde mediante servicios HTTP, servicios REST, o incluso un broker MQTT al que se puede suscribir un cliente para recibir los datos.

TTN nos brinda una forma simple de desplegar un servicio IoT sin requerir una gran infraestructura, y de forma libre. Solo se requiere tener cobertura de un gateway perteneciente a la red, propio o de un tercero y algún dispositivo que envíe información a la aplicación.

2.6 Protocolo de Comunicación MQTT

MQ Telemetry Transport (MQTT) es un protocolo de mensajería abierto, simple y ligero, basado en publicaciones y suscripciones a mensajes en formato JSON, entre un cliente y un broker. El protocolo trabaja sobre TCP/IP, pero existen otras variantes como MQTT-SN que funcionan sobre redes no TCP/IP. Este protocolo en particular ha sido diseñado para localizaciones remotas con ancho de banda limitado.

Las entidades involucradas en la comunicación son las siguientes:

- **El cliente** puede ser desde un microcontrolador hasta un servidor, es decir, sensores, dispositivos móviles, servidores, etc. Esta entidad se encarga de enviar datos hacia el broker (generar datos) mediante mensajes de tipo publish, o a solicitar que el broker le envíe datos (consumir datos) mediante mensajes de tipo subscribe.
- **El Broker** es el núcleo del protocolo, recibe todos los mensajes de los clientes, los filtra y se los reenvía a los clientes interesados en un determinado topic. Se encarga de la autenticación y autorización de clientes.

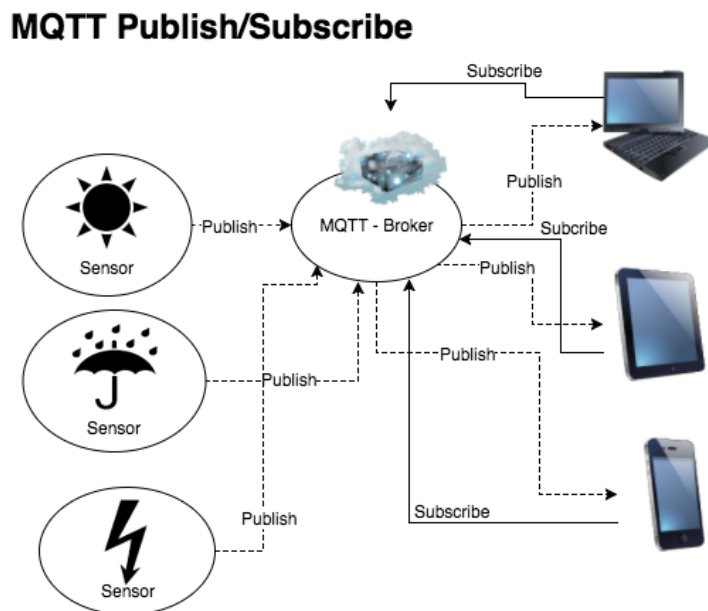


Figura 2.11: Esquema de Funcionamiento MQTT

El broker MQTT separa al publicador del suscriptor, los clientes se conectan siempre con él, los clientes son publicadores y/o suscriptores. Un cliente envía un mensaje (Publish) y uno o más clientes reciben el mensaje, si previamente se han suscrito a ese topic. Cuando un cliente se suscribe a un topic, cada vez que alguien publica algo en ese topic el broker reenvía esos mensajes a todos los suscriptores. Esto se puede observar en la Figura (2.11).

MQTT se basa en los topics que son como etiquetas en los mensajes para decidir qué cliente recibe qué mensaje. Los topics son cadenas de texto jerárquicas separadas por /.

2.7 Visión por Computador

La visión artificial o visión por computador consiste básicamente en la deducción automática de la estructura y propiedades de un mundo tridimensional, posiblemente dinámico, a partir de una o varias imágenes bidimensionales de ese mundo. En esta área de conocimiento se añan conceptos de la física del color, óptica, electrónica, geometría, algorítmica, sistemas de computación, etc [18].

La visión por computador ha surgido como una alternativa confiable para la evaluación de calidad de diferentes productos agrícolas, pues permite el análisis rápido, efectivo, de bajo costo, no destructivo y objetivo de las características externas e internas del producto en la imagen digital. Sin embargo se presentan algunas dificultades al separar objetos del fondo u objetos superpuestos, es por ello que se debe escoger adecuadamente las técnicas de procesamiento de imágenes a utilizar.

Los procesos de la visión artificial puede agruparse en 3 niveles, según su complicación e implementación [7].

- Procesos de bajo nivel: se refieren a operaciones primitivas como es la captura o adquisición de imágenes y el pre-procesamiento de imágenes para reducir el ruido, mejorar el contraste y nitidez de la imagen. Una característica importante de estos procesos es que sus entradas y salidas o resultados son imágenes.
- Procesos de nivel medio: se refieren a operaciones como la segmentación y la descripción de los objetos individuales presentes en una escena para reducirlos a una forma adecuada para su tratamiento informático, reconocimiento y clasificación de estos. Un rasgo importante de estos procesos es

que sus entradas son generalmente imágenes, pero sus resultados son características extraídas de las imágenes como bordes, contornos o la identidad de los objetos presentes en las imágenes.

- Procesos de nivel superior: se refieren a operaciones que reconocen un conjunto de objetos en la imagen y realizan funciones cognitivas que normalmente se asocian con la visión, conocido como el proceso de interpretación.

Aunque estas etapas son aparentemente secuenciales, esto no es necesario, y pueden aparecer interacciones entre los diferentes niveles incluyendo la retroalimentación de los niveles altos a los inferiores.

2.7.1 Elementos de visión por computador

2.7.1.1 Iluminación

Los sistemas de visión por imágenes utilizados en tareas relacionadas con la agricultura y en particular en AP deben trabajar en condiciones meteorológicas adversas y en muchos casos variables en cortos espacios de tiempo. La identificación de estos sistemas juegan un papel importante varios factores entre los que destaca la intensidad luminosa, particularmente la procedente del sol como fuente de iluminación natural.

Uno de los problemas a los que se debe enfrentar la visión por computador en un entorno natural es la iluminación, la cual afecta considerablemente a la calidad de la imagen, ya que una iluminación excesiva provoca una saturación física en el sensor CCD, haciendo que la imagen pierda calidad. Esto se debe a que la mayoría de los píxeles pertenecientes a la zona de interés o bien no son detectados, o bien se identifican como otro elemento asociado a la zona de interés, dando lugar a falsos positivos. Cuando esto ocurre el sistema de visión está trabajando de forma inapropiada y necesita un nuevo ajuste del iris integrado en el sistema óptico o un ajuste en el tiempo de exposición si la cámara lo permite.

Una estrategia propuesta en [7] consiste en determinar si una imagen dada es válida para la identificación de plantas verdes (cultivo y malas hierbas) en base a sus altos niveles de intensidad producidos por exceso de iluminación. La estrategia propuesta se enmarca dentro de las técnicas de aprendizaje adaptadas a tal propósito, donde el objetivo consiste en tomar una decisión sobre la calidad de una imagen basándose en los niveles de intensidad de ésta. Si la calidad se considera suficiente, se puede aplicar un proceso de detección del verde para la identificación de plantas. Por el contrario, si la calidad de la imagen se considera

insuficiente, la imagen deberá ser descartada o se deberá emitir una alarma indicando que el proceso de adquisición no está funcionando correctamente, siempre y cuando no sea posible realizar una corrección de los factores que provocan el fenómeno de iluminación no deseado.

2.7.1.2 Cámaras

El elemento principal para la adquisición de imágenes es la cámara, siendo objeto de este estudio las cámaras digitales. La parte principal de una cámara digital es su sensor. Este sensor no es más que fotodiodos, formados por miles o millones de componentes de distintos materiales sensibles a la luz, los cuales al estar expuestos a la radiación y la luminosidad procedentes de los objetos en la escena, son capaces de capturar este tipo de información que se traduce en lo que se conoce como imagen digital.

Generalmente, las cámaras digitales utilizan dos tipos de sensores: sensor CCD (Charge Couple Device, en español “dispositivo de carga acoplada”) y sensor CMOS (Complementary Metal Oxide Semiconductor, en español “semiconductor complementario de óxido metálico”). La calidad de la resolución de una cámara digital no depende únicamente del número y distribución de píxeles que me puede dar la cámara, sino también de otros factores como las características del sensor y características del lente.

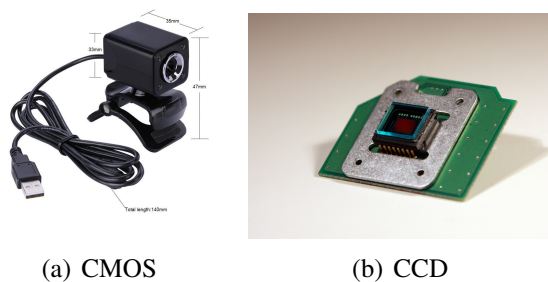


Figura 2.12: Cámaras Digitales

2.7.1.3 Sensor CCD

Este sensor es uno de los más utilizados en la formación de la imagen digital. Proporciona buena calidad de imagen, pero su fabricación es muy compleja y costosa ya que necesitan de otro circuito integrado que es el ACD (Analog Digital Converter, convertidor analógico-digital en español).

El funcionamiento es muy simple, el sensor CCD se encarga de transformar las cargas de las celdas de la matriz en voltajes y de entregar una señal analógica de salida que será posteriormente digitalizada por la electrónica asociada a la cámara, es decir, se hace una lectura de cada uno de los valores correspondientes a las celdas y esta información se traduce a datos mediante un convertidor analógico-digital. La lectura de estas cargas se realiza mediante desplazamientos sucesivos y de forma secuencial. La figura (2.13) muestra un esquema genérico de un sensor CCD.

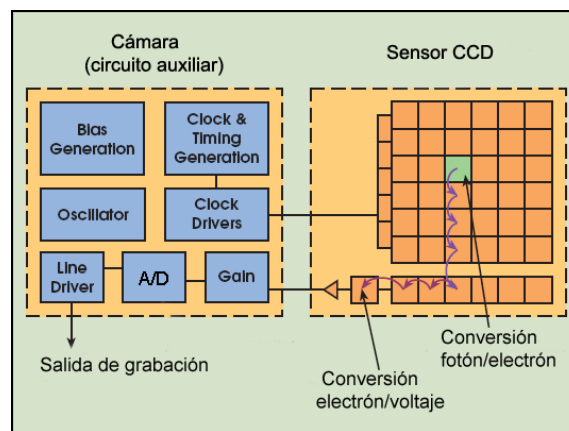


Figura 2.13: Esquema de un sensor CCD

2.7.1.4 Sensor CMOS

En un sensor CMOS (también llamado APS, Active Pixel Sensor), cada celda es independiente del resto. La principal característica es que la digitalización de los píxeles se realiza de forma interna mediante una serie de transistores acoplados a cada celda, por lo que todo el trabajo se lleva a cabo dentro del sensor y no se necesita un chip externo, lo cual se traduce en una reducción de costes a la vez que permite el diseño de equipos más pequeños. La figura (2.14) muestra un esquema de la disposición de los distintos elementos que componen un sensor de tipo CMOS.

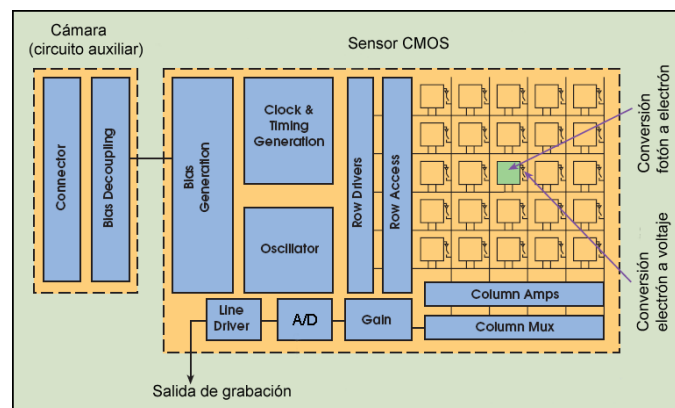


Figura 2.14: Esquema de un sensor CMOS

2.8 Procesamiento de Imágenes

2.8.1 Introducción

Es el proceso en el que es posible tomar fotografías o fotogramas de vídeo y someterlos a un tratamiento con el fin de extraer determinadas características o parámetros, o bien para elaborar nuevas imágenes procesadas como material de salida. Es importante recalcar que el desarrollo de los métodos de procesamiento digital de imágenes tiene su origen en dos áreas principales de aplicación: el mejoramiento de la información pictórica para la interpretación humana, y el procesamiento de datos de la imagen para la percepción de máquina autónoma en el que se incluyen etapas de transmisión y/o almacenamiento de estos datos.

El buen desempeño de un sistema de procesamiento de imágenes en un sistema de visión por computador depende en gran parte de los componentes que lo conforman, se pueden destacar y observar en (2.15) principalmente seis, los cuales son:

- **Captación:** es el proceso mediante el cual se obtiene una imagen visual.
- **Pre procesamiento:** incluye técnicas tales como la reducción de ruido y realce de detalles.
- **Segmentación:** es el proceso por el cual se divide una imagen en objetos de interés.
- **Descripción:** es la fase del proceso en la que se obtienen características convenientes para diferenciar un tipo de objeto de otro.

-
- **Reconocimiento:** es el proceso que asocia un significado a un conjunto de objetos reconocidos.

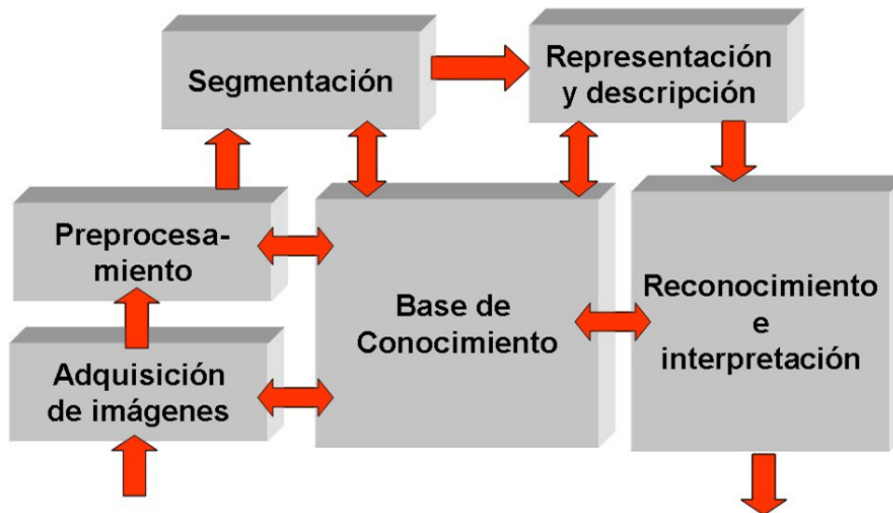


Figura 2.15: Etapas de un Sistema Visión por Computador

2.8.2 Modelos de Color

Un modelo de color es un modelo matemático abstracto que describe la forma en que se representan los colores como matrices de números, utilizando normalmente tres o cuatro valores o componentes de color. Cuando un modelo se asocia a una descripción precisa de cómo se han de interpretar las componentes, el conjunto de colores obtenido se denomina espacio de color. A continuación se describen distintas formas en las que la visión a color puede ser representada [19].

2.8.2.1 RGB

El modelo de color RGB (Red, Green, Blue; Rojo, Verde, Azul) está basado en la síntesis aditiva, mediante la cual los colores primarios rojo (R), verde (G) y azul (B) se combinan de distintas formas para generar un conjunto de colores. Así, combinando uno de estos colores primarios con otro también primario en proporciones iguales se obtienen los colores aditivos secundarios: cian (C), magenta (M) y amarillo (Y). Combinando los tres colores primarios con los máximos valores de representación se obtiene el blanco puro (W), si la combinación se realiza con los mínimos valores posibles el resultado es el negro puro y si la combinación se

realiza también con idénticas proporciones de colores primarios pero con valores de representación intermedios, se obtienen los grises. Este modelo de color es uno de los más utilizados en las cámaras a color. Un canal de color está representado por una matriz donde el valor de cada componente se calcula a partir de una combinación de valores digitales en bits, así en una representación de 8 bits por valor, el máximo valor de representación es $2^8 - 1 = 255$ y el mínimo es 0. En este caso, el rango de valores queda establecido entre 0 y 255, en el que se sitúan los valores intermedios. La (2.16) muestra el esquema general del modelo de color RGB con los colores primarios, secundarios y el blanco.

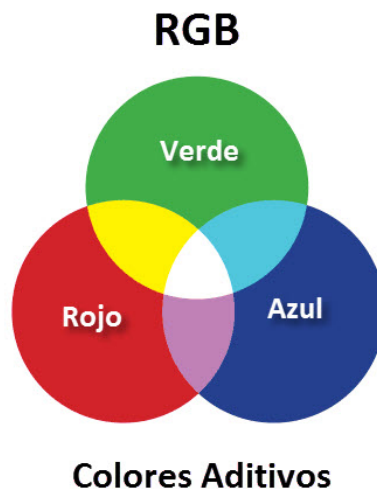


Figura 2.16: Esquema del modelo de color RGB

2.8.2.2 HSV

En el modelo HSV (Hue, Saturation, Value; Matiz, Saturación, Valor) los colores se distinguen unos de otros por su tono o matiz, saturación y valor. El tono hace referencia al color dominante tal y como se percibe, el valor representa la iluminación percibida, y la saturación es la cantidad de luz blanca mezclada con el color dominante, y por lo tanto, diferencia un color intenso de uno pálido. Es por eso que este modelo se utiliza a menudo cuando la iluminación de la escena es determinante y se desea trabajar con la intensidad de la imagen. Existe una relación de transformación entre los modelos RGB y HSV y viceversa. [18].

- **Tono:** Es el color en sí mismo, resultante de la combinación de colores primarios. Todos los matices están representados en un círculo cromático.

Los valores del círculo cromático están en el rango entre 0 y 360.

- **Saturación:** Se representa como la distancia al eje de brillo negro-blanco. Los valores posibles van del 0 al 100. A este parámetro también se le suele llamar "pureza" por la analogía con la pureza de excitación y la pureza colorimétrica de la colorimetría. Cuanto menor sea la saturación de un color, mayor tonalidad grisácea habrá y más decolorado estará.
- **Valor:** Describe la luminosidad, la intensidad luminosa. Es la cantidad de luz emitida por un color. Los valores van de 0 a 100.

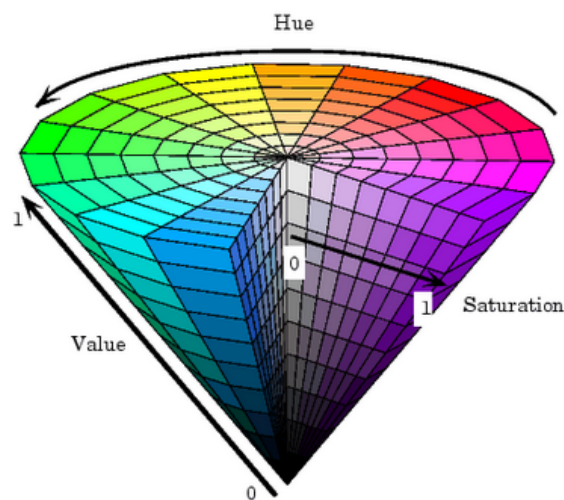


Figura 2.17: Esquema del modelo de color HSV

2.8.2.3 YUV

El modelo YUV representa la percepción humana del color de una forma más cercana que el estándar RGB usado por el hardware de gráficos por ordenador, ya que considera la característica de la visión humana de ser más sensible a los cambios en la intensidad de la luz que a los cambios en el color. El parámetro Y representa la luminancia (información en blanco y negro), mientras que U y V representan la crominancia (información relacionada con el color). Se utiliza porque admite un nivel razonable de error en las componentes de crominancia sin que el humano aprecie el fallo. Este modelo se desarrolló para codificar la transmisión de información de los televisores a color garantizando simultáneamente el funcionamiento de los televisores en blanco y negro, ya que permite enviar información

de color independiente de la información de luminancia. Existe una relación de transformación entre los modelos RGB y YUV y viceversa.

2.8.2.4 Y D_B D_R

Basado en el modelo YUV, este modelo es utilizado en el estándar de televisión en color SECAM (Color secuencial con memoria). Como en el caso anterior consta de una componente que representa la luminancia (Y) y dos que representan la crominancia (D_B y D_R). En este caso se relaciona con el modelo RGB en que los valores se suman para generar una señal simple de luminancia Y, mientras que la señal D_B se obtiene restando Y de la señal azul del modelo RGB, y D_R restando Y de la señal roja, siendo ambas multiplicadas por factores de escala diferentes. Es por ello que este modelo es útil para diferenciar los píxeles de una imagen con dominancia de la componente roja.

2.8.3 Reconocimiento de Patrones

Los patrones se obtienen a través de la extracción de características, un patrón es un conjunto de características, para ello es importante, después de hacer la adquisición de la imagen, tener presente qué procedimiento se quiere realizar, para diferenciar las características tanto de los objetos que se desean detectar como de los objetos indeseados. Debido a que en las imágenes hay variedad de características, se busca una en especial que en primer lugar debe haber un realce de ese contenido en la foto. Para su posterior clasificación en base a las características planteadas [20].

En referencia a los patrones presentados en los cultivos pueden ser espectrales o espaciales, siendo el más común los patrones espaciales ya que incorporan criterios de color, tono, textura, forma, contexto espacial y describir brevemente que los patrones espectrales hacen uso de sensores que muestran la energía capturada en cada longitud de onda del espectro electromagnético [21].

2.8.3.1 Umbralización

Es uno de los más importantes métodos de segmentación. El objetivo es convertir una imagen en escala de grises a una nueva con sólo dos niveles, de manera que los objetos queden separados del fondo [22].



Figura 2.18: Umbralización

2.8.3.2 Segmentación

Subdivide una imagen en sus partes constituyentes u objetos, con el fin de separar las partes de interés del resto de la imagen, por lo tanto el nivel al que se lleva a cabo esta subdivisión depende del problema a resolver. En el proceso de detectar las partes en una imagen se identifican bordes de la imagen, o se segmenta en regiones, líneas o curvas, etc. Otra definición considera a la segmentación como la clasificación de los puntos de la imagen (píxeles), indicando las clases a la que pertenecen los diferentes píxeles. Los atributos básicos de segmentación de una imagen son: la luminancia en imágenes monocromáticas, los componentes de color en imágenes en color, textura, forma, etc [23].

2.8.3.3 Máscara

Es una imagen que contiene únicamente dos colores, blanco y negro (binarizada), adopta la función de selección según el rango que sea asignado a esta, pintando de color blanco los píxeles que están contenidos dentro del rango especificado a la máscara, y descarta todo lo que es de diferente valor, pintándolo de negro, haciendo posible aislar elementos indeseados.

2.8.4 Operaciones Morfológicas

Las operaciones morfológicas son técnicas muy utilizadas en el procesamiento de imágenes. Partiendo de una imagen binaria donde un valor lógico de 1 (color blanco) pertenece al objeto de interés (objeto) y un valor lógico 0 (color negro) pertenece al fondo (otros elementos o ruido), las operaciones morfológicas pueden mejorar los resultados obtenidos mediante el proceso de binarización directa, preservando a la vez características esenciales y eliminando aspectos irrelevantes. Es por esto que las operaciones morfológicas pueden usarse con los siguientes objetivos:

- Pre-procesamiento de imágenes (suprimir ruido, simplificar formas).
- Destacar la estructura de objetos (transformaciones homotópicas, extraer esqueleto, marcado de objetos, envoltura convexa, granulación, rellenado de regiones, ampliación y reducción).
- Mejorar el resultado de la descripción cualitativa de objetos (área, perímetro, diámetro, etc.).

Habitualmente las operaciones morfológicas se usan sobre imágenes binarias, aunque también existen operaciones morfológicas para imágenes en escala de grises.

El conjunto más simple de operaciones morfológicas está formado por erosión, dilatación, apertura y cierre. El conjunto más simple de operaciones morfológicas está formado por erosión, dilatación, apertura y cierre. Estas transformaciones morfológicas utilizan elementos estructurales (M) que se construyen alrededor de un origen local (punto representativo), siendo los más comunes aquellos conjuntos que están 4-conectados (M_4) y 8-conectados (M_8), que tienen la apariencia que se muestra en (2.1):

$$[M_4] = \begin{bmatrix} 0 & 1 & 0 \\ 1 & \cdot 1 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad [M_8] = \begin{bmatrix} 1 & 1 & 1 \\ 1 & \cdot 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad (2.1)$$

Estos elementos estructurales M , tal y como se verá a continuación, se aplican sobre cada píxel de la imagen binaria I dando como resultado una posible modificación del valor del píxel en función de la operación morfológica elegida.

2.8.4.1 Erosión

La erosión, que es una transformación no invertible (la imagen original no se recupera), se lleva a cabo según la ecuación (2.2):

$$I \otimes M = \{p \in E^2 : p + m \in I \mid \forall m \in M\} \quad (2.2)$$

siendo I la imagen binaria, M el elemento estructural utilizado y E^2 el espacio

Euclídeo 2D. Por lo tanto, cada píxel p perteneciente a la imagen I se evalúa con el elemento estructural, el resultado de la erosión viene dado por los píxeles p para los cuales todos los posibles $p + m$ están en la imagen I .

De forma gráfica, se recorre la imagen siguiendo una dirección (normalmente de izquierda a derecha y de arriba abajo), y allí donde aparece un valor 1 lógico en el origen del elemento estructural se procede a realizar la operación morfológica, de tal forma que, si todos los unos del elemento estructural coinciden con unos en la imagen binaria, ese píxel se marca con valor 1, en caso contrario se le asigna un valor lógico 0 [7].

2.8.4.2 Dilatación

Esta operación es la dual de la erosión. La dilatación es el resultado de considerar el conjunto de píxeles de todas las posibles adiciones vectoriales de pares de elementos, uno de cada conjunto I y M . Desde el punto de vista visual, la dilatación añade al objeto aquellos puntos del fondo que lindan con los bordes del mismo, y tal y como ocurre con la erosión, se trata de una transformación no invertible. La dilatación se lleva a cabo según la ecuación (2.3):

$$I \oplus M = \{p \in E^2 : p = mx + m \in I \mid \forall m \in M\} \quad (2.3)$$

La forma gráfica de aplicar esta operación, es la siguiente: se recorre la imagen

siguiendo una dirección (normalmente de izquierda a derecha y de arriba abajo) y donde aparece un 1 lógico en el origen del elemento estructural, éste se evalúa, de tal modo que se ponen a 1 aquellos píxeles de la imagen binaria que coinciden con un uno en dicha posición del elemento estructural [7].

La figura (2.19) muestra el resultado de aplicar las operaciones morfológicas a una imagen binaria, la parte superior es la imagen original mientras que la de

abajo es la erosión y la imagen inferior es producto de la dilatación.



Figura 2.19: Operaciones de Erosión y Dilatación

2.8.4.3 Apertura y Cierre

La combinación de ambas operaciones (erosión y dilatación), da lugar a dos nuevas transformaciones dependiendo del orden de aplicación. Dado que la erosión y la dilatación son transformaciones no invertibles, la imagen resultado es una imagen más simplificada y menos detallada que la imagen original.

La apertura consiste en aplicar una erosión seguida de una dilatación. La apertura de una imagen I por un elemento estructural M se denota como $I \circ M$ y se define según la expresión ecuación (2.4):

$$I \circ M = [I \otimes M] \oplus M \quad (2.4)$$

La figura (2.20) muestra el resultado de aplicar la operación de apertura a una imagen binaria. Al lado izquierdo se encuentra la imagen original y a lado su transformación.



Figura 2.20: Operación Morfológica de Apertura

El cierre, por el contrario, consiste en aplicar una dilatación seguida de una erosión. El cierre de una imagen I por un elemento estructural M se denota como $I \bullet M$ y se define según la expresión (2.5):

$$I \bullet M = [I \oplus M] \otimes M \quad (2.5)$$

La figura (2.21) muestra el resultado de aplicar la operación de cierre a una imagen binaria.

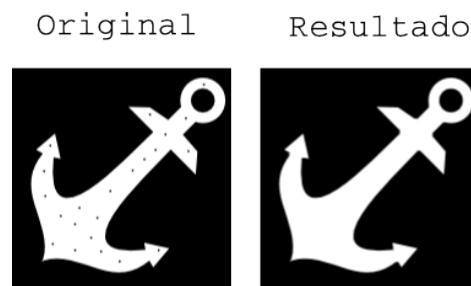


Figura 2.21: Operación Morfológica de Cierre

Capítulo 3

Caso de Estudio y Propuesta del Sistema

Este capítulo trata de describir en qué consiste la solución que se ha llevado a cabo, explicando detalladamente cada una de las partes que la constituyen: zona de trabajo, hardware y software. Partiremos explicando por qué se eligió el entorno de trabajo en el cuál se ha trabajado el proyecto, explicando sus condiciones implicadas. Por otro parte, explicaremos el proceso de selección del hardware del sistema para el reconocimiento de plaga y la comunicación con los diferentes frameworks asociados.

3.1 Caso de Estudio

Para el proyecto propuesto se consideró trabajar en un espacio reducido que contenga las características deseadas y mejor resaltadas con el fin de reconocer con herramientas de bajo coste a nuestro objeto de interés(plaga); se escogió entre muchos escenarios a los árboles frutales cítricos ya que en su mayoría por ser zona tropical y mediterránea, en el cual existe el calor y la humedad a un alto nivel se presentan muchos casos de ataques a éstos árboles de una plaga especial, que es de un color blanquizo y amarillento de nombre común cochinilla y nombre científico *Planococcus Citri* de la familia de *Pseudococcidae*. Esta plaga además de tener mucha presencia en las plantas cítricas posee un color blanco que difiere claramente con respecto a los demás elementos del árbol.

La figura (3.1) representa el lugar donde se trabajó la parte experimental y experimentó el sistema, está ubicado en el Jardín del Turia que es una zona abierta dentro de la comunidad Valenciana.



Figura 3.1: Entorno de Trabajo

3.1.1 El Cotonet *Planococcus citri*(Risso)

Es una especie extendida desde hace muchos años por todas las zonas tropicales y subtropicales del mundo. Las cochinillas de esta familia se alimentan sobre todo del floema de las plantas, con lo que producen abundante melaza. Las hormigas también transportan a las larvas y las defienden de los depredadores, ya que se alimentan de la melaza que esta especie produce [24].

Su reconocimiento se debe a su color claro, amarillento característico, y las hembras son amarillentas y está cubierta de abundante secreción cérica harinosa de color blanco como se ve en la figura (3.2).



(a) Plaga adulta



(b) Limonero con plaga



(c) Panorámico de árbol experimental



(d) Segregación de melaza

Figura 3.2: Planococcus citri

3.2 Arquitectura Propuesta

La infraestructura de red propuesta esta basado en una serie de componentes como un sensor de imágenes, un computador quién realice el procesamiento de imágenes, un microcontrolador para la comunicación y conexión LoRaWAN, una puerta de enlace para enlazar las redes Long Range (LoRa) y IoT y a su vez logrará conectarnos a un servidor TTN de almacenamiento en la nube basado en dispositivos IoT con el que el usuario final podrá revisar el registro de datos que nos envíe el prototipo propuesto. En la imagen (3.3) apreciamos la distribución de los componentes de la red y las tecnologías utilizadas en determinadas partes de la arquitectura.



Figura 3.3: Arquitectura de red Propuesta

3.2.1 Hardware Empleado en el Sistema

En este apartado se analizarán en detalle los diferentes dispositivos y herramientas que han sido utilizados, proporcionado de la arquitectura propuesta para integrar los diferentes elementos. Así mismo se tratarán

3.2.1.1 Raspberry Pi

Raspberry Pi es un ordenador de placa reducida de bajo coste, desarrollado en Reino Unido por Raspberry Pi Foundation con el objetivo de estimular el aprendizaje de la informática en los colegios, diseñada para ser la más barata posible y poder llegar al mayor número de usuarios inmiscuidos en la ciencias de la informática [25].

Raspberry Pi surge con la idea de posibilitar pequeños proyectos en hardware y el aprendizaje de la programación, ya que los ordenadores que tenemos están orientados a tareas informáticas o de ocio, pero cuentan con conexiones para este aspecto.

En cuanto al hardware, podemos encontrar 7 modelos actualmente de Raspberry, Model A, Model B, Model A+, Model B+, Pi 2 Model B, Pi Zero y Pi

3 Model B. Existe un octavo modelo de placa aunque es una versión especialmente pensada para uso empresarial e industrial [26]. Para nuestro escenario se usó modelo Raspberry Pi 3 Model B que cuenta con unas dimensiones de placa de 8.5cm por 5.3cm y presenta unas características muy interesantes. Tiene un chipset Broadcom BCM2387 que contiene un procesador ARM Cortex-A53, con cuatro núcleos, con varias frecuencias de funcionamiento con posibilidad de subirla (overclocking) hasta 1,2 GHz y es de 64 bits. La memoria RAM se mantiene en 1GB y un procesador gráfico VideoCore IV. Los desarrolladores aseguran que este nuevo procesador es diez veces más potente que el de una Raspberry original Pi Zero y un 50 por ciento más rápida que la segunda Raspberry Pi 2 Model B.

En las siguientes figuras (3.4(a)), (3.4(b)) y (3.4) podemos apreciar la distribución de las partes de una Raspberry Pi 3.

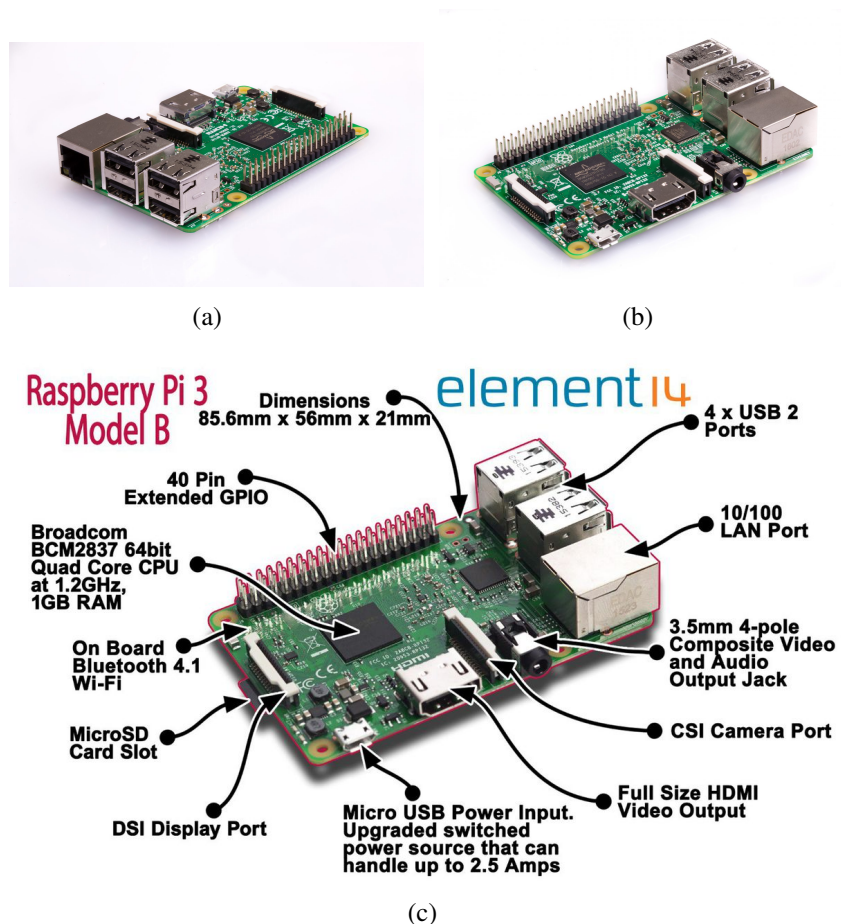


Figura 3.4: Componentes de una Raspberry Pi 3 [2]

También se muestra un repaso de las principales características de la Raspberry Pi 3 Model B, el primer modelo de la Raspberry Pi de tercera generación. Reemplazó el Raspberry Pi 2 Model B en febrero de 2016 [27].

3.2.1.2 LoPy Pycom

LoPy es un microcontrolador del fabricante Pycom, compatible con MicroPython de triple portador para comunicaciones (LoRa, Wifi, Bluetooth) - plataforma de Innovation in the Open (IO) de calidad empresarial perfecta para sus Cosas conectadas. Con el último conjunto de chips Espressif, LoPy ofrece una combinación perfecta de potencia, amabilidad y flexibilidad. Crea y conecta cosas en todas partes.

Entre sus principales características de este dispositivo LoPy de tamaño 55 mm x 20 mm x 3.5 mm que se aprecia en la figura (3.5) es que tiene un Microcontrolador Dual Core Espressif ESP32 chipset, una memoria RAM: 512 KB, Flash externa 4 MB, 802.1b / g / n 16mbps, Bluetooth de baja energía y clásico, Especificación LoRa: Semtech LoRa transceptor SX1272, el consumo de energía con WiFi es 12mA en modo activo y 5uA en modo de espera, con LoRa: 15mA en modo activo y 1-uA en modo de espera.

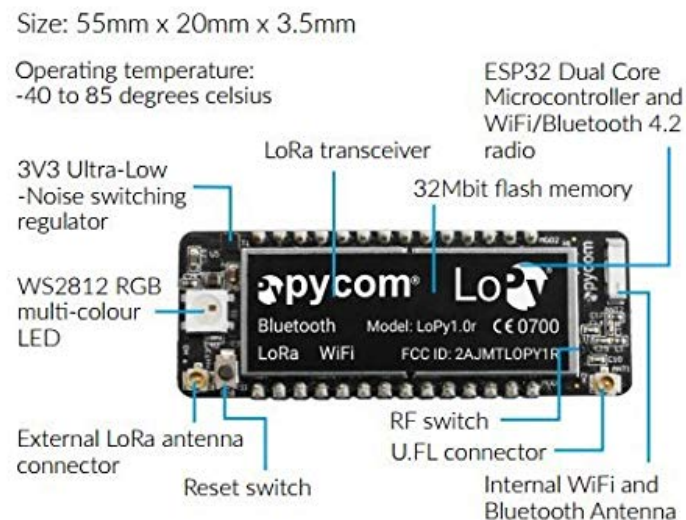


Figura 3.5: Microcontrolador LoPy de Pycom

Se puede encontrar algunas especificaciones del fabricante en [28].

Éste microcontrolador necesita de una placa de expansión para acondicionar y operar de manera fácil el dispositivo, a su vez para la comunicación necesita de tres materiales que son el pigtail, conector U.FL en un extremo y conector SMA del otro extremo para terminar conectando la antena LoRa como vemos en la figura (3.6) .

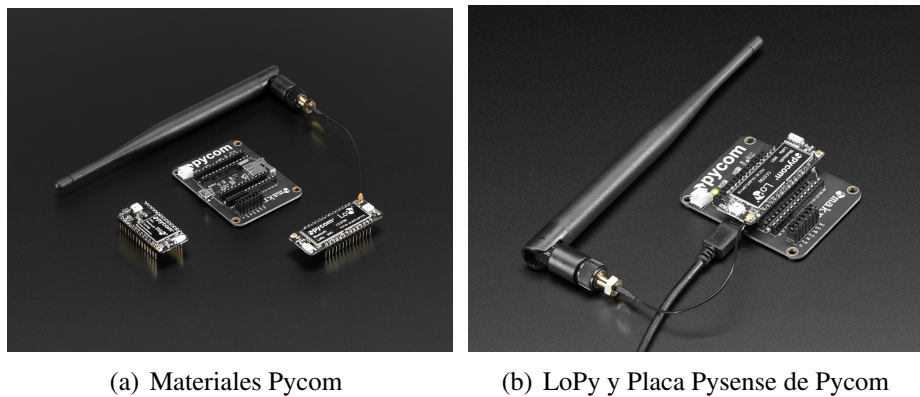


Figura 3.6: Dispositivos Pycom para la Comunicación LoRa

3.2.1.3 Logitech C920

En un principio se utilizó el módulo de cámara Raspberry Pi oficial para analizar las características de nuestro objeto y las diferentes funcionalidades y librerías de operación que ofrece este dispositivo por ser recomendada de la misma compañía Raspberry. Sin embargo no se logró detectar o esclarecer nuestro objeto con los algoritmos empleados así que tuvimos que migrar a otra cámara web de menor presupuesto que ofrezca mejor calidad en imágenes y opere en entornos de luminosidad variables.

Se utilizó la Logitech C920 por las capturas que realizadas y algunos análisis en la etapa de pruebas, el no ser de carácter específico no limita hacer uso de esta. A continuación se presenta las especificaciones de la webcam [29]:

- Alta resolución e imágenes nítidas a través de la cámara de hasta 15 megapíxeles.
- webcam Full HD Logitech C920 cuenta con una lente de cristal Full HD con cinco elementos de cristal para capturar imágenes muy nítidas, a la vez que su enfoque automático de gama alta proporciona una alta definición en todo momento.

- Fácil cableado de la cámara con una Raspberry Pi a través de un cable plano.
 - Compatible con la Raspberry Pi 3 y todas las versiones de Raspberry Pi.
- Detalles técnicos se observan en la figura (3.7).



(a) Logitech C920

Especificaciones

Especificaciones Generales	
Marca	Logitech
Modelo	C920
Color	Negro
Especificaciones de la Cámara	
Sensor de imagen	Lente Carl Zeiss
Sistema de Enfoque	Enfoque automático de 20 pasos
Resolución de Video	Full HD 1080p (hasta 1920 x 1080 píxeles) y hasta 30 cuadros por segundo (H.264)
Conexiones	USB 2.0 (compatible con USB 3.0)
Micrófono Integrado	2 Micrófonos estéreo integrados con reducción de ruido automático
Resolución de Fotografías	SI (hasta 15 Mpx con mejora de software)
Características Especiales	Tecnología Fluid crystal Corrección automática de iluminación escasa

(b) Especificaciones

Figura 3.7: Cámara del Sistema

3.2.2 Software Empleado en el Sistema

En la actualidad hay una gamma alta de lenguajes de programación, desde muy antiguos hasta muy actuales, estos lenguajes presentan determinadas prestaciones, y su uso viene dado por el tipo de aplicación software que se pretenda desarrollar. Además la facilidad de aprendizaje por parte del desarrollador.

Sin embargo enfocándonos en el objeto de este trabajo, en el que hacemos uso de tecnología de hardware libre como el Raspberry Pi, se ha revisado varias recomendaciones en la web y en [9] del uso de Python, OpenCV, Raspbian, Ubuntu, QT por su fácil manejo y prestaciones, en el que describiremos un breve resumen y aplicación de estos.

3.2.2.1 Mycropython

MicroPython es un pequeño pero eficiente interprete del Lenguaje de Programación Python 3 que incluye un subconjunto mínimo de librerías y que además está optimizado para que pueda correr en microcontroladores. Con MicroPython tienes la posibilidad de escribir códigos más simples, en lugar de usar Lenguajes de Programación de más bajo nivel como C o C++, que es el que utiliza Lopy de Pycom por ejemplo.

MicroPython es un compilador completo de Python. El usuario tiene un prompt interactivo (REPL) para ejecutar comandos de forma inmediata, junto con la capacidad de ejecutar e importar scripts desde el sistema de archivos incorporado [30].

Hay algunas características que MicroPython tiene y es lo que lo hace único y diferente de otros sistemas embebidos:

- Tiene una REPL Interactiva (Read-Eval-Print Loop Por sus siglas en Ingles). Que es un pequeño programa que Lee e interpreta los comandos del usuario, los evalua y después imprime el resultado. Esto permite conectar alguna tarjeta (microcontrolador que soporte Python) y esta tiene que ejecutar el código sin necesidad de compilar ni cargar el programa.
- Muchas librerías. Así como el Lenguaje de Programación Python cuenta con un sin fin de librerías para la ejecución de tareas, MicroPython también viene bien cargado con bastantes paquetes para ahorrar trabajo. Es posible ejecutar análisis de datos JSON desde un servicio web, búsqueda de texto en expresiones regulares o hasta levantar un Socket dentro de una red tan solo con las funciones ya precargadas.
- Extensibilidad. Para los usuarios avanzados de MicroPython, pueden extender de Python a funciones de más bajo nivel como C o C++, pudiendo mezclar códigos que requieran de ejecución más rápida a bajo nivel con MicroPython que es de más alto nivel.



Figura 3.8: Logo de MycroPython

3.2.2.2 OpenCV

Open Source Computer Vision Library (OpenCV) es una librería de Vision por Computador de código abierto, del inglés (Open Source Computer Vision Library), es una biblioteca con licencia BSD (Berkeley Software Dstribution) que incluye bastantes algoritmos de Visión Artificial. Desarrollado inicialmente por Intel, la primera versión apareció en 1999 y ha sido en Junio de 2015 cuando la última versión ha sido lanzada (3.0.0). Durante todos estos años, la biblioteca ha sido actualizada constantemente con numerosos algoritmos añadidos y corrección de errores [31].

OpenCV tiene tiene interfaces C/C++, Python y Java, y soporta Windows, Mac Os, GNU/Linux, IOS y Android. La biblioteca está escrita en un lenguaje C/C++ optimizado y orientado a la eficiencia computacional con un enfoque especial en aplicaciones en tiempo real [32].

Uno de los paquetes de la estructura modular que posee OpenCV con respecto al campo de procesamiento de imágenes y obeitivo de este proyecto es: Image Proccessing(Procesamiento de Imagen). En este módulo se incluyen filtros de imagen lineales y no lineales, transformaciones geométricas de imagen, conversiones del espacio de colores, histogramas, etc. Estas funciones son utilizadas en distintas áreas de la visión por computador, como es la inspección de productos, identificación de personas u objetos en movimiento, reconocimiento de rostros humanos en una imagen, imágenes médicas, seguridad, interfaces de

usuario, reconstrucción 3D, robótica, etc.



Figura 3.9: Logo de Librería OpenCV

3.2.2.3 Librerías MQTT

Son librerías en Python que proporciona funciones del protocolo Message Queuing Telemetry Transport (MQTT) con el objetivo de crear proyectos basados en IoT. Estas bibliotecas son compatibles con dispositivos de Pycom y Raspberry Pi y simplificarán procesos en el desarrollo del software para la parte de comunicaciones del sistema.

Entre las librerías mas usadas para estos proyectos de IoT estan Paho-MQTT [33], MQTT-Pycom [34] y un servidor/broker o sandboxe libre en internet, el Mosquitto [35].

Capítulo 4

Procedimiento Detección y Comunicación

En este apartado nos concentraremos en explicar los procedimientos para la detección, tratamiento de imágenes y comunicación en el escenario propuesto. Se describirá la arquitectura de comunicación, la configuración, ubicación y la interacción de los dispositivos asociados a la transmisión por la red LoRaWAN.

4.1 Condiciones de Luminosidad

Teniendo como referencia la problemática de la iluminación en la sección 2.7.1.1, en el que detalla aspectos de iluminación variable y afección a las imágenes capturadas. Las condiciones de este escenario se realizaron al intemperie donde la iluminación es muy variante, así mismo el clima, viento, temperatura y animales son factores claves que se observaron para acondicionar y limitar al sistema de registro y tratamiento de imágenes.

La parte de recolección de imágenes se presentó en la estación de verano, se trabajó y analizó varias muestras en la Raspberry Pi; imágenes que fueron capturadas durante el día, tarde y noche en algunas ocasiones; llegando a la conclusión que la poca y/o excesiva iluminación provoca una saturación física en el sensor de la cámara haciendo que se pierda la calidad de la imagen y en consecuencia los algoritmos utilizados no sean eficientes a la hora de encontrar el objeto de interés o mezclar el objeto con falsos positivos y como parte de la solución a este caso respecto a las condiciones de iluminación se necesitaba a favor un día despejado, sin viento, soleado con una posición del sol al anochecer, las horas más representativas fueron de 19:00 a 21:00 horas para la captura de imágenes.

4.2 Procesos de Detección

Este sistema está compuesto por dos bloques de procedimientos en los que se concentran todo el procesamiento de imágenes aplicando técnicas de visión por computador y Fog Computing con la Raspberry Pi y otro que es la comunicación IoT aplicando tecnologías Long Range Wide Area Network (LoRaWAN) en la Lopy y MQTT sobre WIFI y redes Internet Protocol (IP) en ambos dispositivos hardware Lopy y Raspberry Pi, que se encargarán de enviar la información y se describirán con mas detalle en el siguiente capítulo.

En resumen se muestra un diagrama de flujo que permite expresar de manera más sencilla lo que el sistema propuesto contiene y la forma en la que funciona el código que se diseñó para llevar a cabo el sensado de imágenes, tratamiento de imágenes y transporte de información a la nube. A continuación, se muestra en la figura (4.1) en el que se explicaran los diferentes bloques del diagrama.

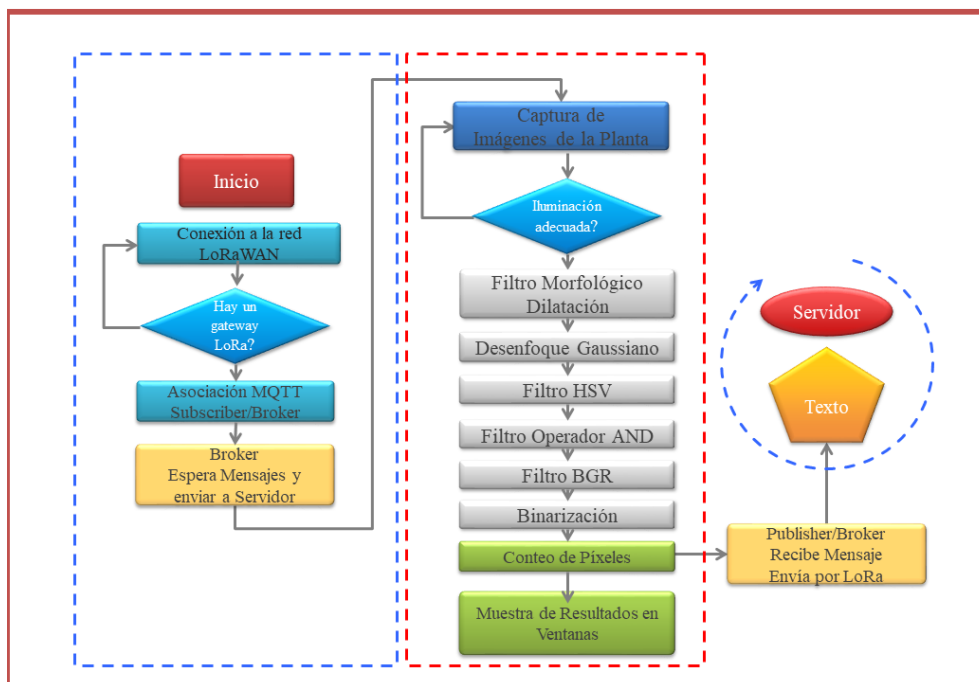


Figura 4.1: Diagrama de flujo del Sistema

4.2.1 Adquisición de Imágenes

El primer proceso de un sistema de visión artificial es la adquisición de las imágenes. Este proceso se puede realizar por diferentes medios, los principales son por medio de una cámara web o importando una imagen ya establecida. En esta etapa se obtuvo la información a utilizar que fue elemento fundamental sobre el que se va a llevar a cabo el procesamiento y guía hacia las técnicas de visión. Es buena idea capturar imágenes de prueba y procesarlas con algún software disponible antes de capturar un conjunto completo de datos. Para este trabajo se tomaron un promedio de 40 imágenes de prueba en diferentes partes del árbol, con diferentes posiciones y distancias para analizar los mejores lugares que se presentaron en dichas imágenes y establecer que condiciones nos favorecen, los cuales fueron tomas frontales y distancias cortas de 0.3 a 0.5 mts.

Podremos ahorrarnos muchos procesos que conlleva a mal uso de los recursos del hardware y software con la cámara utilizada; para este trabajo se seleccionó la Logitech C920 descrita en la sección 3.2.1.3 quién sensará las imágenes óptimas y buen contraste; siempre es mejor tratar de reducir los posibles problemas de procesamiento de imágenes por adelantado, en lugar de tratar de procesar imágenes malas e inconsistentes.

La mayoría de las cámaras vienen con una configuración por defecto siendo un caso de suma importancia configurar muchos parámetros de la cámara a la hora de adquirir imágenes con el objetivo de establecer algunos parámetros en la umbralización que le compete a los filtros de procesamiento de imágenes dentro del algoritmo trabajado. Además a parte de la resolución y la velocidad de captura, existen otros parámetros configurables a la hora de realizar capturas con la cámara, entre los parámetros que se modificaron acorde al sistema estos son algunos y de valores aleatorios para cada tipo de cámara, de los cuales presentamos a continuación:

- Sharpness. Establece la definición de la imagen (de 0 a 255), 128 por defecto.
- Contrast. Establece el contraste de la imagen (de 0 a 255), 128 por defecto.
- Brightness. Establece el brillo de la imagen (de 0 a 255), 128 por defecto.
- Saturation. Establece la saturación de la imagen (de 0 a 255), 128 por defecto.
- Exposure. Define el modo de exposición (auto, night, backlight, etc).

-
- Gain. Ganancia de la imagen ayuda a mejorar la escasa intensidad de luz que hay en el medio (de 0 a 255), 0 por defecto.

Para poder manipular estas herramientas mencionadas anteriormente, debemos trabajar en el sistema con el modulo **OS.System** de Python e instalar el controlador de dispositivos de video4linux [36] en la Raspberry Pi en el que se podrá apreciar en el código anexado.

Como segunda parte de la adquisición con los parámetros adecuados en la cámara digital, se sigue la estrategia sugerida en la sección 2.7.1.1 que es la de evaluar las imágenes capturadas y toma de decisión, que mediante la Raspberry Pi se determina la validez de una imagen según su luminosidad media Y media donde Y es la componente de Luminancia en un espacio euclídeo [18]. Para obtener dicho valor se realiza la conversión de la imagen capturada RGB al espacio gris o escala de grises de la siguiente forma en la ecuación (4.1).

$$Y = 0,30R + 0,59G + 0,11B \quad (4.1)$$

Podemos definir una imagen en el plano Y como una función bidimensional $f(x_1, x_2)$, donde $x=(x_1, x_2)$ son las coordenadas espaciales, y el valor de f en cualquier x es la luminancia de la imagen en dicho punto. Realizamos el barrido en dicha imagen y hallamos la media ponderada con una ecuación simple cuyo resultado nos dará un valor entre [0,255] por trabajar con datos habituales 8 bits.

Dados los n números $x_1, x_2, x_n, x_1, x_2, x_n$, la media aritmética se define como:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i = \frac{x_1 + x_2 + \dots + x_n}{n} \quad (4.2)$$

En las siguientes figuras de (4.2) se muestra imágenes mal capturadas y dependen mucho de la iluminación q posteriormente serán rechazadas ver (4.2(a)) y (4.2(b)), puesto que se volverá a capturar la imagen mejor ajustada a los valores predeterminados ver (4.2(c)).



(a) Captura con poca iluminación



(b) Captura con mucha iluminación



(c) Buena captura

Figura 4.2: Adquisición de Imágenes

4.2.2 Procesado y Segmentación

Como segundo proceso del procesamiento de imágenes es el filtrado y realce del objeto o región de interés Region of Interest (ROI) de interés, basado en el reconocimiento de color que se reflejan en el tallo del árbol, debido a que a simple percepción del ojo humano puede distinguirse con claridad la diferencia de colores impregnados en el árbol como son el tallo, hojas, frutos y la plaga como ROI, se descartó la parte de tamaño y forma debido a la expansión y despliegue sin correlación en la planta de estos insectos. Se buscó, inicialmente, utilizar algún tipo de morfología matemática para resaltar dichas características. Posteriormente, se realizó una suavización de la imagen para disminuir el ruido y aumentar la precisión del algoritmo. Además, se aplicó un filtro HSV, que es muy robusto en tareas de separación de color, y se hizo una umbralización, seguido de un operador matemático para eliminar restos de píxeles considerados como ruido, seguido de una

separación de colores BGR que es así como interpreta OpenCV para distinguir mejor la plaga; por último se binarizó la imagen para trabajar la extracción; con lo que esta parte del algoritmo esta compuesto por 6 etapas como se muestra en la figura (4.3) y se describirán a continuación:

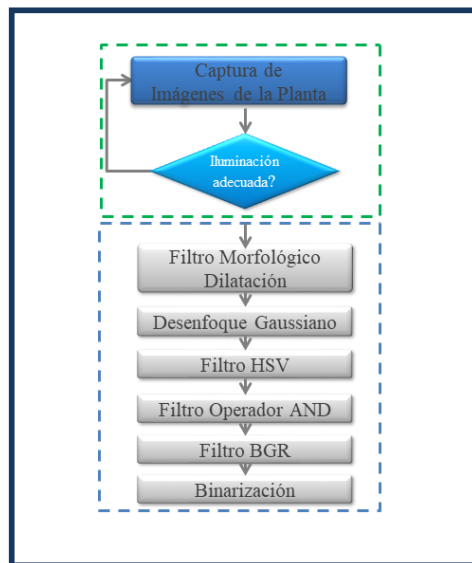


Figura 4.3: Proceso de Segmentación y Filtrado

4.2.2.1 Filtro Morfológico Dilatación

Inicialmente se buscó un algoritmo que permitiera resaltar las características de la plaga que se pretenden reconocer en la imagen. Este objetivo se puede alcanzar a través de un tipo de morfología matemática denominada ‘dilatación’, la cual se describe como un crecimiento de píxeles. Esto permite el aumento de un píxel alrededor de la circunferencia de cada región y de esta forma poder incrementar las dimensiones. Este mismo método fue utilizado en [9]. La función utilizada está predeterminada en la librería OpenCV y la dilatación está dada por las ecuaciones (4.3) y (4.4). Lo que significa que el conjunto de todos los posibles vectores es la suma de pares de elementos, uno perteneciente a A y el otro a B

$$A \oplus B = \{ a + b : a \in A \text{ y } b \in B \} \quad (4.3)$$

$$X \in A \oplus B \Leftrightarrow x = a + b, a \in A \text{ y } b \in B \quad (4.4)$$

donde, A y B son estructuras geométricas de espacio continuo o discreto.

En las imágenes presentadas en (4.4) vemos la transformación tras aplicar el filtro dilatación y resaltar la plaga.

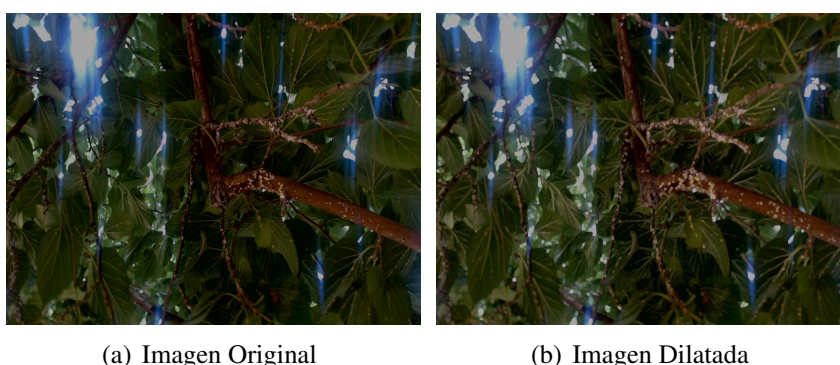


Figura 4.4: Comparación de imágenes con el filtro Dilatación

4.2.2.2 Filtro Gaussiano

Después de aplicar la dilatación se observó que en la imagen aún aparecía blancos muy intensos que son los de la luz solar considerado como ruido en la imagen y algún otro ruido puede ser generado por el filtro morfológico. Esto hace necesario suavizar la imagen a través del proceso de filtrado. En este caso se utilizó el Desenfoque Gaussiano o "Gaussian Blur". El tipo más común de filtros es el lineal como en la ecuación (4.5), en el que el valor de un píxel de salida (es decir $g(i, j)$) se determina como una suma ponderada de valores de píxeles de entrada (es decir $f(i + k, j + l)$):

$$g(i, j) = \sum_{k,l} f(i + k, j + l)h(k, l) \quad (4.5)$$

$h(k, l)$ se llama kernel, que no es más que los coeficientes del filtro, ver mas detalles en [37].

En la imagen (4.5(b)) de la figura (4.5) se muestra la aplicación del Gaussian Blur a la imagen resultante después de aplicar la dilatación.



(a) Imagen Original

(b) Imagen Suavizada

Figura 4.5: Comparación de imágenes con el segundo filtro GaussBlur

4.2.2.3 Espacio de canales HSV

Con la imagen suavizada de la figura X se realiza el filtrado, con el fin de obtener únicamente las el tallo y la plaga eliminando hojas, luz y otros objetos no deseados de la planta, para posteriormente, calcular el porcentaje infestado de plaga incrustado al tallo que se encuentra afectado. Se utiliza un filtro HSV, debido que es muy robusto y más flexible que los filtros RGB y muy utilizado en visión artificial [38] y AP, además filtra los colores sin tantos problemas de luz o brillo, como presenta el filtro RGB [39]. Para el filtro se utilizan valores predeterminados entre [0 y 255] por ser datos de 8 bits, estos valores serán establecidos por el ajuste de valores de las imágenes analizadas con un panel selector de valores HSV como apreciamos en la imagen (4.6(a)) de la figura (4.6). Los valores utilizados de saturación se tomaron entre 45 y 140, la luminosidad entre 50 y 227 y el tono entre 6 y 20. como se muestra en la imagen resultante de aplicar el filtro HSV (4.6(c)) de la de la figura (4.6).

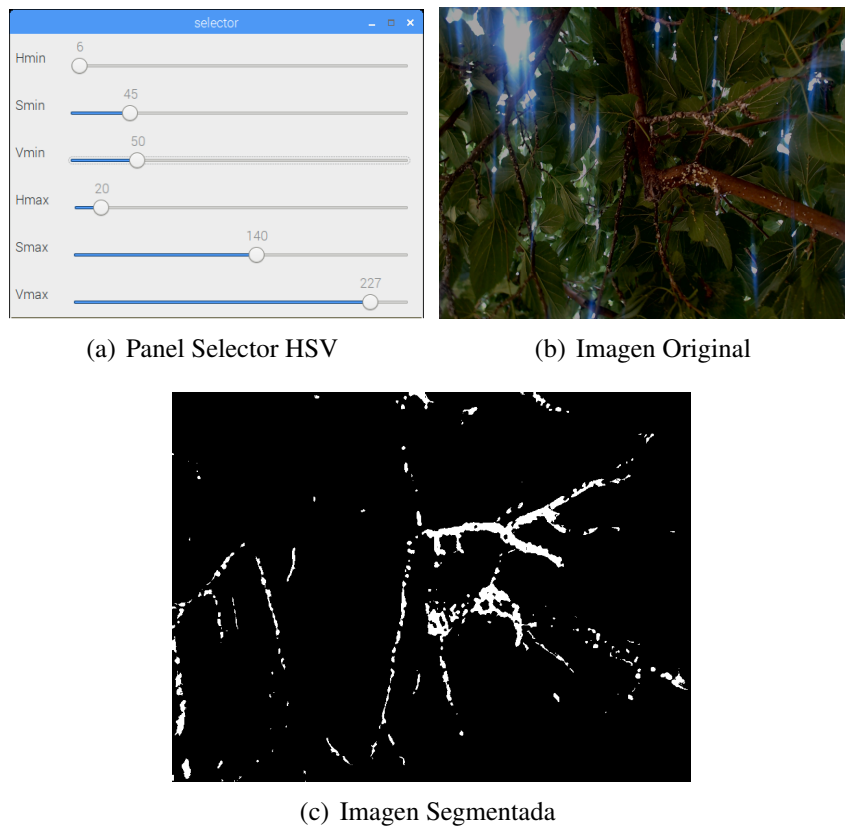


Figura 4.6: Segmentación de la imagen mediante el espacio de colores HSV

4.2.2.4 Filtro Operador AND

Luego de proceder el filtrado en el espacio de canales HSV necesitamos segmentar el tallo y la plaga, entonces el método que se propone es el filtrado en los canales RGB, sin embargo como ya se normalizó los el rango de valores HSV necesitamos aplicar una función de retorno a la imagen original RGB sustrayendo todo los elementos sin interés, eliminados previamente con los filtros anteriores, este trabajo se puede lograr con una función matemática bitwise-and u operador matemático conocido como AND bit a bit. El AND bit a bit, o bitwise, toma dos números enteros y realiza la operación AND lógica en cada par correspondiente de bits. El resultado en cada posición es 1 si el bit correspondiente de los dos operandos es 1, y 0 de lo contrario. En la siguiente figura apreciamos la segmentación de tallo incrustada la plaga.



(a) Imagen Original

(b) Imagen Segmentada Plaga y Tallo

Figura 4.7: Segmentación de imágenes con retorno al espacio de colores RGB aplicando el operador AND.

4.2.2.5 Espacio de canales BGR y Binarización

Finalmente se aplicó la segmentación de colores en el espacio BGR puesto que en OpenCV se antecede primero el canal Blue y no comúnmente utilizado RGB. Para obtener el rango de valores a filtrar se trabajó con el visor de imágenes Paint, y éste indique la posición de los píxeles que queremos encontrar, y con dicha posición ejecutar comandos de consultas de valores BGR en el framework Python dentro de la Raspberry Pi. Encontrado la serie de valores no deseados, para esta etapa se tomaron 15 datos, se notará que éstos valores son aproximados el uno con el otro, con lo que se selecciona un rango de valores coherentes, en el que se asignan los mínimos y máximos a los valores del filtro en el código, quienes representan los valores del espacio RGB del tallo, quedando el objeto de interés o plaga en blanco y negro o binarizado y procesarla como un arreglo bidimensional. En la imagen (4.8(b)) de la figura (4.8) vemos la separación de la plaga y eliminación del rango de colores marrón y finalmente binarizar la imagen resultante.

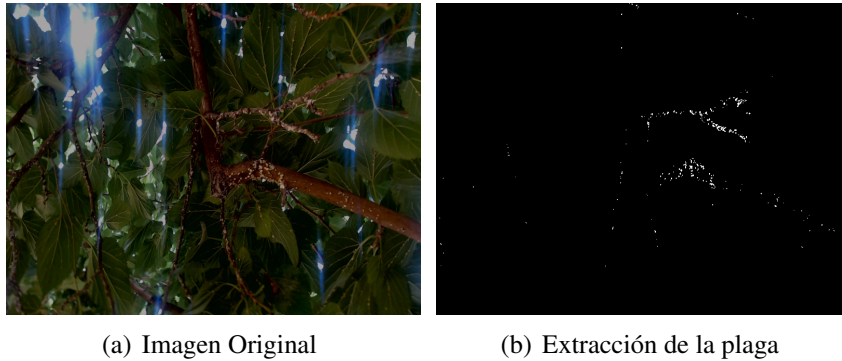
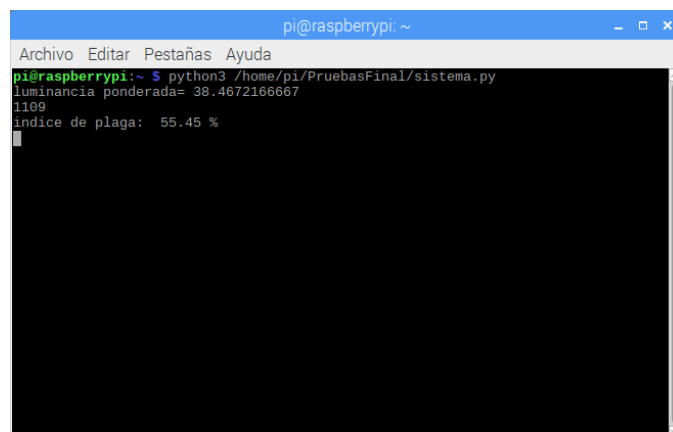


Figura 4.8: Segmentación de imágenes en el espacio de colores RGB y extracción del objeto de interés.

4.2.3 Extracción de Características

A continuación, se procede a pasar la imagen a un arreglo bidimensional (matriz), con valores de 1 (blanco) y 0 (negro), con el fin de realizar una búsqueda de las zonas blancas o píxeles blancos que son la representación de la plaga y calcular el factor de afección de plaga en cultivo. Éste factor se consigue haciendo un barrido en la imagen o arreglo de datos de la imagen tratada en el paso anterior y realizar un conteo de los blancos o unos y con una regla aritmética simple indicar que un aproximado de 2000 píxeles blancos es un 100 por ciento de plaga infestada en la planta y 0 por ciento los 0 píxeles encontrados.

En la imagen (4.9) se muestra un impresión por consola de los datos que representará a las imágenes procesadas:



```
pi@raspberrypi: ~  
Archivo Editar Pestañas Ayuda  
pi@raspberrypi:~ $ python3 /home/pi/PruebasFinal/sistema.py  
luminancia ponderada= 38.4672166667  
1109  
indice de plaga: 55.45 %
```

Figura 4.9: Representación de la imagen procesada por visión por computador

4.3 Infraestructura de Red TTN

La plataforma IoT en el que el sistema trabajó fue TTN, como conceptualizamos en el subcapítulo (2.5.2) del capítulo (2), y dada las características y bondades que se ofrece al usuario, como toda las opciones de la tecnología LoRaWAN, bibliotecas para nodos, software para gateways, etc, se ha elegido esta infraestructura como la más adecuada para este trabajo.

La siguiente figura (4.10) muestra una pequeña arquitectura de comunicaciones de este trabajo sobre la plataforma TTN en el que se describirá el funcionamiento a continuación:

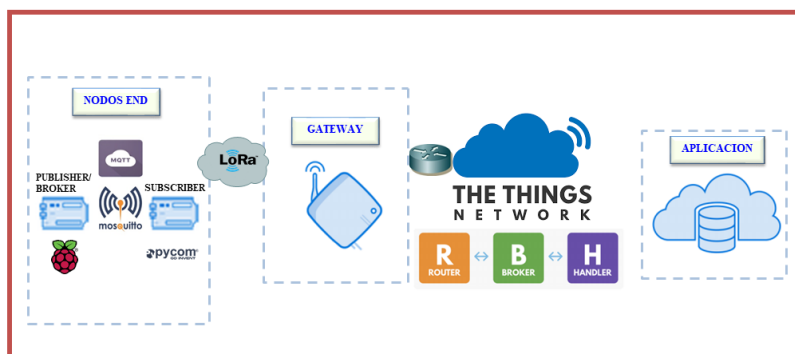


Figura 4.10: Arquitectura de Comunicaciones del Sistema sobre la Plataforma TTN

En resumen, los dispositivos Raspberry Pi y Lopy deben estar en red mediante la tecnología WIFI para habilitar la capa MQTT y ambos serán el nodo final; en el protocolo LoRaWAN el nodo manda mensajes a las estaciones base a través del protocolo de radio LoRa. Una o varias gateways reciben el mensaje del nodo y se encargan de retransmitirlo a la aplicación mediante un servidor que se describe a continuación. Las gateways están conectadas a un router que es el encargado de ajustar su estado y programar las transmisiones. Cada Router se conecta a uno o más broker que identifican el nodo emisor y mandan el mensaje a la aplicación correcta. También en sentido inverso, eligen al gateway a la que enviar los mensajes

dowlink desde la aplicación. Un Handler maneja los datos de las aplicaciones y es también en el Handler donde los datos se encriptan y desencriptan. También existe el Network Server que es el encargado de las funciones específicas de LoRaWAN. Para ver más detalles de la red TTN revisar en [40].

- **Nodo:** Se une a la red por OTAA [41]. Envía datos a la aplicación creada.
- **Gateway:** Recibe los mensajes LoRa, los convierte en paquetes de información y se los manda al router.
- **Router:** Microservicio que recibe los mensajes de las gateways y busca un broker a quién enviárselos.
- **Broker:** Microservicio que identifica el nodo, desdobra el tráfico y envía el paquete al handler donde está registrada la aplicación.
- **Handler:** Microservicio que encripta y desencripta los payload y publica los mensajes a los brokers para que los usen las aplicaciones.
- **Aplicación:** El software que usa el usuario para recopilar los e interpretar los datos.

4.4 Procesos de Enlace LoRaWAN

El procedimiento o secuencias de trabajo del bloque de las comunicaciones esta relacionado con el procesamiento de imágenes descrito en el apartado anterior y resumida en la figura (4.1) del capítulo (4). En el siguiente diagrama de flujos (4.11) mostramos el bloque de la comunicación del sistema.

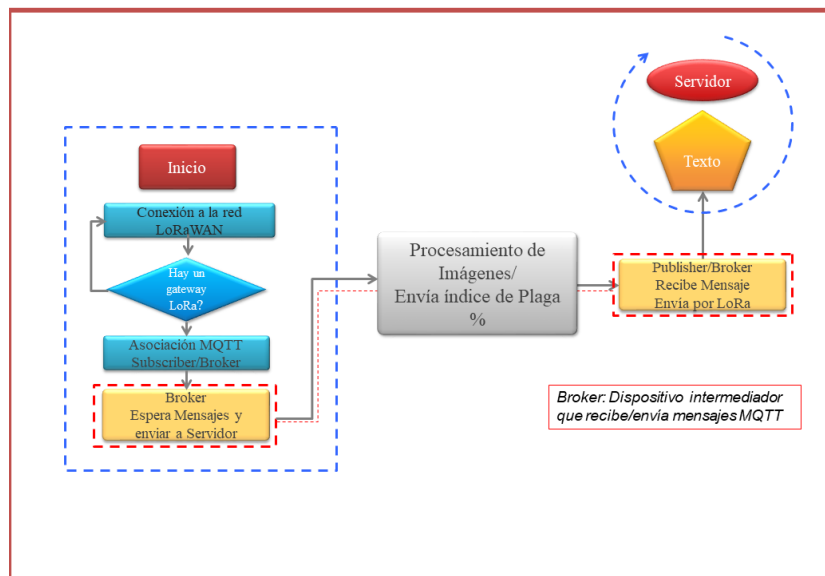


Figura 4.11: Diagrama de flujo del Bloque de Transmisión del Sistema

En resumen el código inicia con la búsqueda y conexión a una red LoRaWAN de parte del nodo final hasta encontrar algún gateway LoRaWAN quién lo registre, seguidamente se proceden a enlazar una comunicación Machine to Machine (M2M) del subcriptor al broker sobre WIFI, luego el broker pasará a un estado de espera de mensajes de parte del publisher, luego pasará al análisis de imágenes y extracción de la información relevante que representa todo el proceso para finalmente enviar dicha información al broker que estaba en espera y éste transmitirlo por LoRaWAN a un servidor en la nube y el usuario visualizar la data en la aplicación. Ahora nos concentraremos en explicar la creación del nodo, gateway y aplicación a continuación:

4.4.1 Aplicación en la consola TTN

Una vez definido los componentes de la red, empezamos creando una Aplicación en la consola TTN para registrar el nodo del sistema, para ello es necesario registrarse con un usuario y seleccionar aplicación como primera etapa se muestra en la figura (4.12):

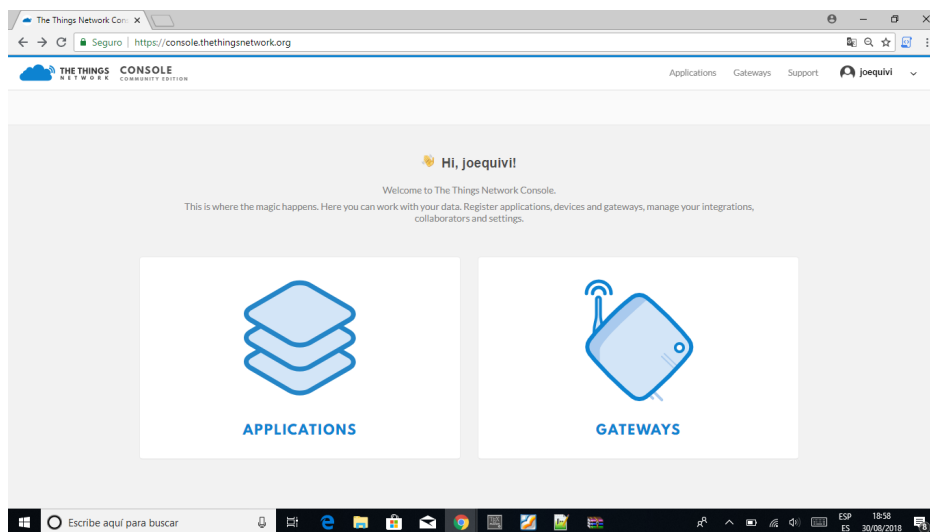


Figura 4.12: Creacion de una Aplicación en TTN

Ésta aplicación genera las claves AppEUI, DevEUI y AppKey que deben copiarse y añadirse al código que se insertará en la Lopy, así mismo las claves son necesarias para activar al nodo usando activación OTAA y el nodo intente unirse con éxito para enviar sus datos como se ve en la figura (4.13):

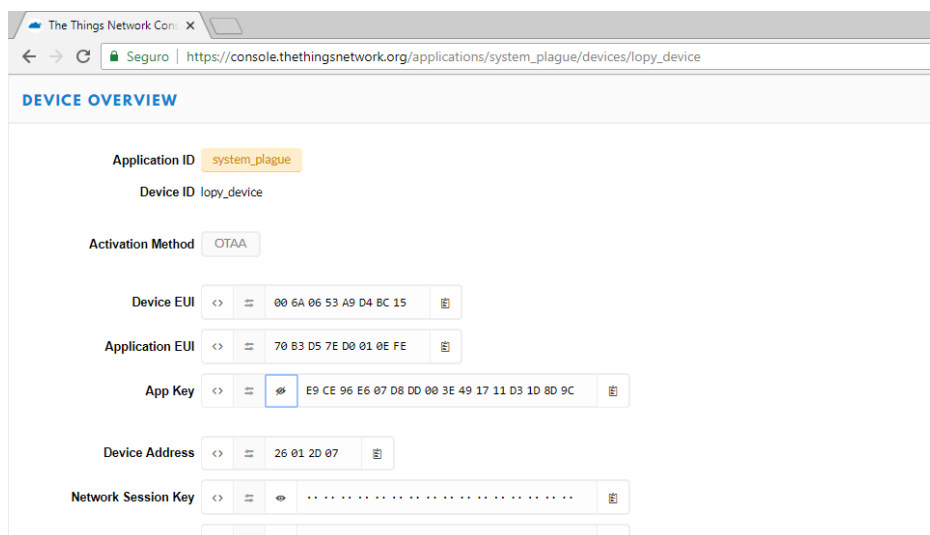


Figura 4.13: Generación de claves en la Aplicación TTN

4.4.2 Gateway

El dispositivo gateway sirve como pasarela para comunicar dos tecnologías por un lado el sistema propuesto basado en LoRa y por otro lado la Aplicación creada sobre la plataforma TTN. Para este trabajo se usó el gateway existente del Grupo de Investigación Grupo de Computadores y Redes (GRC) basado en una Raspberry Pi y agregada a la red TTN. En las imágenes se muestra el dispositivo en servicio.



Figura 4.14: Gateway Basado en Raspberry en Operación

4.4.3 Nodo

Los dispositivos que integran el nodo final son la Raspberry Pi que trabaja como analizador de imágenes y actuar como Broker/Publisher MQTT, y por otro lado está el módulo Lopy de Pycom que trabaja como Cliente MQTT y emisor de mensajes LoRaWAN por el módulo de radio LoRa.

4.4.3.1 Lopy

Para trabajar sobre el módulo de radio LoRa del microcontrolador Lopy se le debe cargar varias librerías como MQTTclient, LoRa y Pycom, ficheros que fueron descargados de los repositorios de GitHub [42] y tutoriales de Pycom [43]. Los

parámetros a configurar son el modo LoRaWAN que básicamente accede a la radio directamente y los paquetes se envían utilizando la modulación LoRa en la frecuencia Europa 868 MHz, añadir las claves generadas por la aplicación para establecer la conexión OTAA, ver figura (4.15).

```
# Initialize LoRa in LORAWAN mode.
lora = LoRa(mode=LoRa.LORAWAN)

# create an OTA authentication params
dev_eui = binascii.unhexlify('006A0653A9D4BC15'.replace(' ', ''))
app_eui = binascii.unhexlify('70B3D57ED0010EFE'.replace(' ', ''))
app_key = binascii.unhexlify('E9CE96E607D8DD003E491711D31D8D9C'.replace(' ', ''))

# join a network using OTAA
lora.join(activation=LoRa.OTAA, auth=(dev_eui, app_eui, app_key), timeout=0)
```

Figura 4.15: Parámetros de configuración en módulo de radio LoRa

Una vez configurado los parámetros LoRa, seguidamente procedemos a añadir los parámetros Cliente MQTT a la Lopy para subscribirlo, y éstos son la dirección IP del Broker MQTT que por defecto la Lopy asignó la primera IP a la Raspberry "192.168.4.2", luego se crea un identificador MQTT y Topic MQTT, para luego la LoPy pasar a modo espera de mensajes MQTT.

```
# parametros mqtt
broker_addr = "192.168.4.2"

#connect to broker mqtt
client = MQTTClient("dev_id", broker_addr, 1883)#crear un id para enlace mqtt
msg= client.set_callback(on_message)

if not client.connect():
    print ("Connected to broker: "+broker_addr)

# subscribe to broker
client.subscribe('lopy/raspy')# crear un topic para recibir mensajes por mqtt
print ("lopy subscribe to broker")
print("Waiting messages...")

while 1:
    msg2=client.wait_msg()
    print("indice de plaga recibido:",msg2)
    s.send(b'IP%'+ msg2)
```

Figura 4.16: Parámetros MQTT para LoPy

4.4.3.2 Raspberry Pi

En el capítulo (4) se describió la principal función de la Raspberry Pi del sistema, que es la de Analizador de imágenes. Para la parte de comunicaciones se optó en aprovechar el módulo WIFI que posee la Raspberry Pi 3 para conectarnos al red WIFI que irradia la Lopy por defecto y a su vez registrar la Lopy mediante un terminal interactivo Read Evaluate Print Loop (REPL), que le permite escribir y probar su código directamente en la Lopy, y descrito en la sección (3.2.2.3) del capítulo (3), ver la figura (4.17).

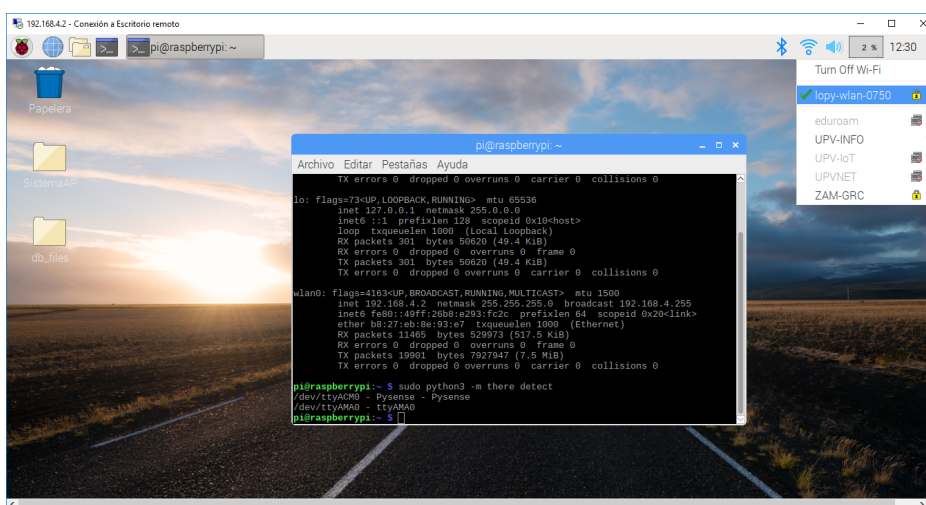


Figura 4.17: Conexión a Red WIFI y Registro del Dispositivo Lopy

Se usó las librerías Paho.Publisher para configurar la Raspberry Pi como Publisher y luego del procesamiento de imágenes poder publicar los mensajes al suscriptor y éste transmitirlos por LoRa. En la siguiente figura (4.18) se observa el uso de algunas funciones y parámetros configurados:

```

import paho.mqtt.client as mqtt
import paho.mqtt.publish as publish

Broker = "192.168.4.2"

#sub_topic = "lopy/raspy"    # receive messages on this topic
pub_topic = "lopy/raspy"    # send messages to this topic

#Iniciamos la camara video
captura = cv2.VideoCapture(0)

# when receiving a mqtt message do this;

def on_message(client, userdata, msg):
    message = str(msg.payload)
    print(msg.topic+" "+message)
    display_sensehat(message)

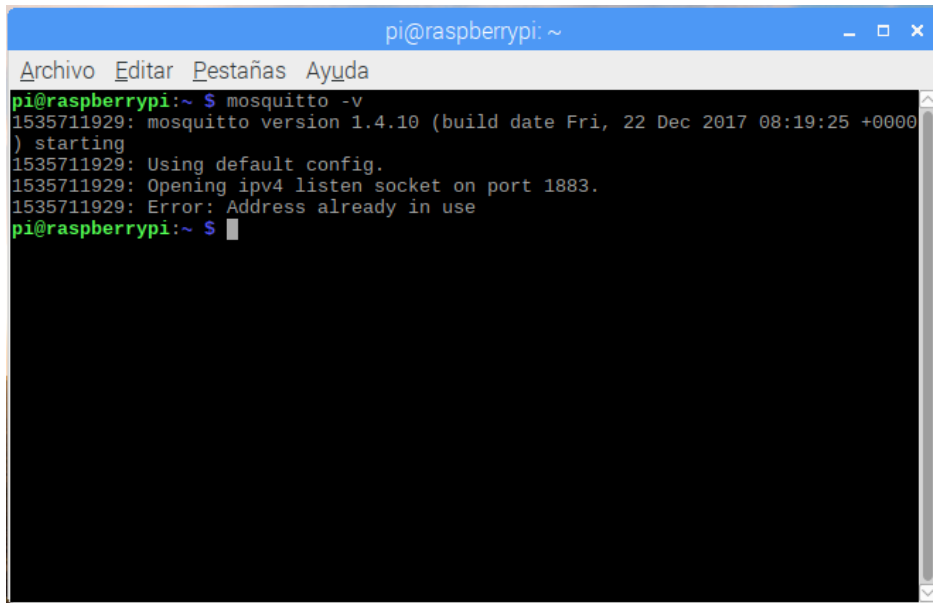
def on_publish(mosq, obj, mid):
    print("mid: " + str(mid))

client = mqtt.Client()
#client.on_connect = on_connect
client.on_message = on_message
client.connect(Broker, 1883, 60)
client.loop_start()
datos = p
client.publish("lopy/raspy",datos)
time.sleep(2)
#panel[:] = [h_min, s_min, v_min]

```

Figura 4.18: Configuración de Parametros Publisher MQTT

Para asignarle la función de Broker MQTT a la Raspberry Pi del sistema se optó por instalar el software Mosquitto y configurarlo en el sistema desde el encendido o reinicio de la Raspberry Pi. En la figura (4.19) se muestra el Mosquitto escuchando mensajes MQTT.



```
pi@raspberrypi: ~  
Archivo Editar Pestañas Ayuda  
pi@raspberrypi:~ $ mosquitto -v  
1535711929: mosquitto version 1.4.10 (build date Fri, 22 Dec 2017 08:19:25 +0000  
) starting  
1535711929: Using default config.  
1535711929: Opening ipv4 listen socket on port 1883.  
1535711929: Error: Address already in use  
pi@raspberrypi:~ $
```

Figura 4.19: Activación de Mosquitto y espera de Mensajes MQTT

Capítulo 5

Resultados

En este apartado describiremos el desarrollo completo y secuencial de la parte experimental del prototipo propuesto, así mismo se analizará de forma cuantitativa el desempeño del conjunto de algoritmos propuestos en el bloque de procesamiento de imágenes y la fiabilidad y tiempo de conexión y llegada de los mensajes a la aplicación web por el medio LoRa.

5.1 Conexión a Red

Los trabajos de comunicación se realizaron en el laboratorio GRC dentro de la Universitat Politècnica de Valencia (UPV). Luego de hacerse usuario TTN y registrar al nodo del sistema, la aplicación entrará en modo desactivado. En la Figura (5.1) vemos que la ultima conexión se realizó hace una hora y esta modo espera.

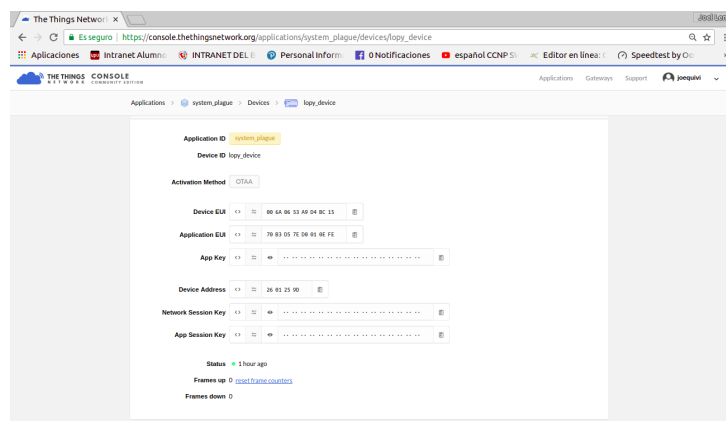
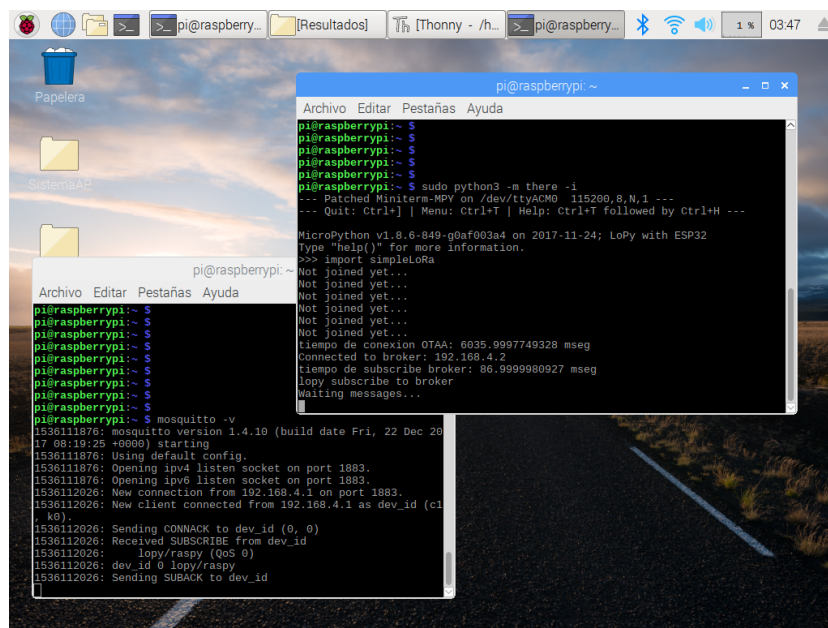


Figura 5.1: Aplicación TTN en modo espera

En modo LoRaWAN inicialmente el sistema debe conectarse a la primera gateway LoRa que esté disponible. Siendo la única gateway LoRaWAN montada en el laboratorio GRC, el nodo debe enviar una serie de llaves de identificación y seguridad para acceder a la aplicación, vale decir que el nodo solicita un join a la red con los datos de configuración y abre la ventana de recepción. El Gateway recibe la solicitud y la envía al servidor. El servidor verifica que el nodo este dado de alta y la llave de encriptación sea correcta. Si es correcta asigna una sesión temporal y la envía por medio del gateway al nodo, si los datos son incorrectos rechaza el join. El nodo recibe la sesión temporal y puede enviar datos a la red. Con estos datos previos analizamos el tiempo de conexión a la red LoRaWAN por el modo OTAA y se registran en una tabla. También se aprovecha en capturar el tiempo de conexión del Subscriber que es la Lopy al broker que es la Raspberry Pi quién ya tiene activado la aplicación Mosquitto desde inicio de sistema con el puerto 1883 o puerto estándar sin encriptar a modo de pruebas para terminar el cierre de conexión a red y esperar los mensajes analizados por la Raspberry Pi.

En la Figura (5.2) se ve la conexión LoRa por medio de OTAA y el enlace M2M del subscriber al broker:



```
pi@raspberrypi:~$ sudo python3 -m there -i
--- Patched Miniterm-MPY on /dev/ttyACM0 115200,8,N,1 ---
--- Quit: Ctrl+] | Menu: Ctrl+T | Help: Ctrl+T followed by Ctrl+H ---

MicroPython v1.8.6-649-g0af003a4 on 2017-11-24; LoPy with ESP32
Type "help()" for more information.
>>> import simpleLoRa
Not joined yet...
Not joined yet...
Not joined yet...
Not joined yet...
Not joined yet...
tiempo de conexión OTAA: 6635.9997749328 mseg
Connected to broker: 192.168.4.2
tiempo de subscribe broker: 86.9999980927 mseg
Lopy subscribe to broker
Waiting messages...

pi@raspberrypi:~$ mosquitto -v
1536111876: mosquitto version 1.4.10 (build date Fri, 22 Dec 20
17 08:19:25 +0900) starting
1536111876: Using default config.
1536111876: Opening ipv4 listen socket on port 1883.
1536111876: Opening ipv6 listen socket on port 1883.
1536112026: New connection from 192.168.4.1 on port 1883.
1536112026: New client connected from 192.168.4.1 as dev_id (cl
k0).
1536112026: Sending CONNACK to dev_id (0, 0)
1536112026: Received SUBSCRIBE from dev_id
1536112026:   lopy/raspy (QoS 0)
1536112026: dev_id 0 lopy/raspy
1536112026: Sending SUBACK to dev_id
```

Figura 5.2: Conexión OTAA y MQTT

En la tabla (5.1) se muestra el tiempo de conexión constante a la red LoRaWAN que en todos los casos tomados demoró 6.034 segundos aproximadamente

para tener los beneficios del protocolo LoRaWAN, a su vez el tiempo promedio de 88 milisegundos que le cuesta conectarse el Lopy a la Raspberry Pi por MQTT, en total sumarían 6.114 segundos la conexión global del nodo.

Pr.Nº	T.OTAA(ms)	T.MQTT(ms)
1	6035.99	85.00
2	6035.99	85.00
3	6035.99	81.00
4	6035.99	81.99
5	6035.99	80.00
6	6035.99	75.99
7	6035.99	81.99
8	6035.99	75.99
9	6035.99	95.00
10	6035.99	84.00

Tabla 5.1: Tabla de Datos de Conexión a la Red LoRaWAN.

5.2 Tratamiento de Imágenes

De un total de 80 imágenes tomadas por el sistema, desde una distancia de 0.5 metros entre la zona de interés y la cámara Logitech C920, se consideraron 40 capturas para el análisis final del sistema, debido a la presencia de varias irregularidades en las imágenes que afectaban su procesamiento. Por ejemplo, en la Figura (5.3) se muestra una imagen mal capturada por la presencia de viento e inestabilidad del prototipo y la imagen no corresponde a la zona de interés pero si cumple con las condiciones luminosidad del sistema.



Figura 5.3: Imagen mal capturadas

En la Figura (5.4) se muestra una prueba de varias imágenes bien capturadas por la cámara Logitech C920 y luego de analizar y aprobar la condición de luminosidad por la Raspberry Pi, serán guardadas, procesadas y finalmente, extraer el valor representativo de las imágenes para posteriormente pasar a la parte de comunicación.



Figura 5.4: Imagen bien capturada y dentro de las condiciones de luminosidad

Al poner a prueba el sistema en fases iniciales se capturaban imágenes sin condiciones de luminosidad y se ajustaban los parámetros de segmentación con las primeras tomas; para sistemas a campo abierto es inaceptable no poseer un rango de valores o condicionales de luminosidad, ya que la información extraída nos daban falsos positivos y en muchas oportunidades sin información, el parámetro de luminosidad en la adquisición juega un rol muy importante y según los valores que se establezca ayudará al algoritmo analizador a no ser muy complejo o hasta

a veces sin éxito; siguiendo la etapa de capturas de pruebas del sistema de registro de imágenes se encontró que éste parámetro es muy sensible y variante respecto a demasiados factores como posición del sol, estación del año, temperatura, clima, roedores, etc., haciendo que la detección de plaga se viera restringida a ser hecha en un tiempo específico y no en cualquier momento en el que se necesite, estos valores de luminosidad promedio de la una imagen se presentaron mejor entre 36 a 45 trabajando con valores de 0 a 255 por se datos de 8 bits, poniendo como condición al algoritmo solo registrar a éstas imágenes que cumplan dicho rango; así mismo las horas mejor presentadas fueron de 19:00 a 21:00 horas del día en verano.

Para el desarrollo de la investigación se requería llevar a cabo diferentes procesos de recolección de información y tratamiento de los mismos, es importante encontrar la forma en que las muestras recolectadas tuvieran una representación gráfica o resultados en tablas que fuesen comprensibles, para la selección de imágenes se tomaron como dato la luminosidad promedio de una imagen bien tomada, el tiempo de adquisición con una función la función `time()` de python a su vez éstos se imprimieron datos en consola de la Raspberry Pi. Para la parte de segmentación y extracción del índice porcentual de plaga infestada en la planta para una zona de interés se usa un pequeño algoritmo de barrido por toda la imagen y aprovechando que está binarizada podemos contar la una cantidad fija de píxeles blancos que existe en ella y el tiempo que conlleva a procesar la imagen desde su captura.

A continuación en la tabla (5.2) se muestran los resultados de 40 pruebas divididas en 2 partes tanto alta calidad y otra de baja calidad de imagen en la etapa de Adquisición se determinan los tiempos capturados para dos tipos de resoluciones de imágenes que son la High que está en 1920X1080 píxeles por imagen y la Low que está con 800X600 píxeles por imagen, a su vez se halla la luminosidad promedio que está en el rango de valores condicionados en la Adquisición de imágenes.

Pr.Nº	TC/R=H	Lumin/R=H	TC/R=L	Lumin/R=L
1	11.54	38.47	3.49	37.97
2	11.81	38.51	10.67	37.78
3	11.89	38.53	3.60	37.75
4	11.60	38.48	10.93	37.71
5	11.56	38.38	7.21	37.72
6	11.74	38.16	7.38	37.72
7	11.22	37.43	3.57	37.80
8	11.33	38.38	11.11	37.69
9	35.15	38.91	10.98	37.65
10	35.40	38.88	7.34	37.70
11	35.44	39.01	3.59	37.87
12	12.25	39.05	7.09	37.74
13	12.22	39.06	3.66	37.81
14	34.37	38.95	15.03	37.64
15	12.27	38.39	3.60	37.81
16	11.90	38.39	11.56	37.08
17	11.53	38.39	15.01	37.66
18	22.78	39.34	3.39	38.00
19	22.35	39.83	7.62	37.95
20	12.87	38.39	15.13	37.64

Tabla 5.2: Tabla de Datos de Adquisición de Imágenes del Sistema.

Si se analiza los datos obtenidos la luminancia está dentro de los márgenes de luminosidad establecida por el sistema y mantiene cierta concordancia y similitud para ambos escenarios, con respecto al tiempo de captura, es cierto que el proceso de captura tarda menos con una imagen de menor calidad, sin embargo se nota que el tiempo es muy inconstante y en muchas ocasiones se tiene que repetir la solicitud de captura aprovechando que sería mejor optar por la captura de imágenes con mayor definición.

Éstas imágenes se sometieron a cambios de canal RGB a HSV, dilatación y suavización, modificaciones de características morfológicas como procesos de segmentación de umbralización por la Raspberry Pi que da como resultado que a partir de una fotografía original, cómo se da el aislamiento de objetos que no son afines a la plaga y a la región de interés o planta tal como se muestra en las siguientes imágenes:

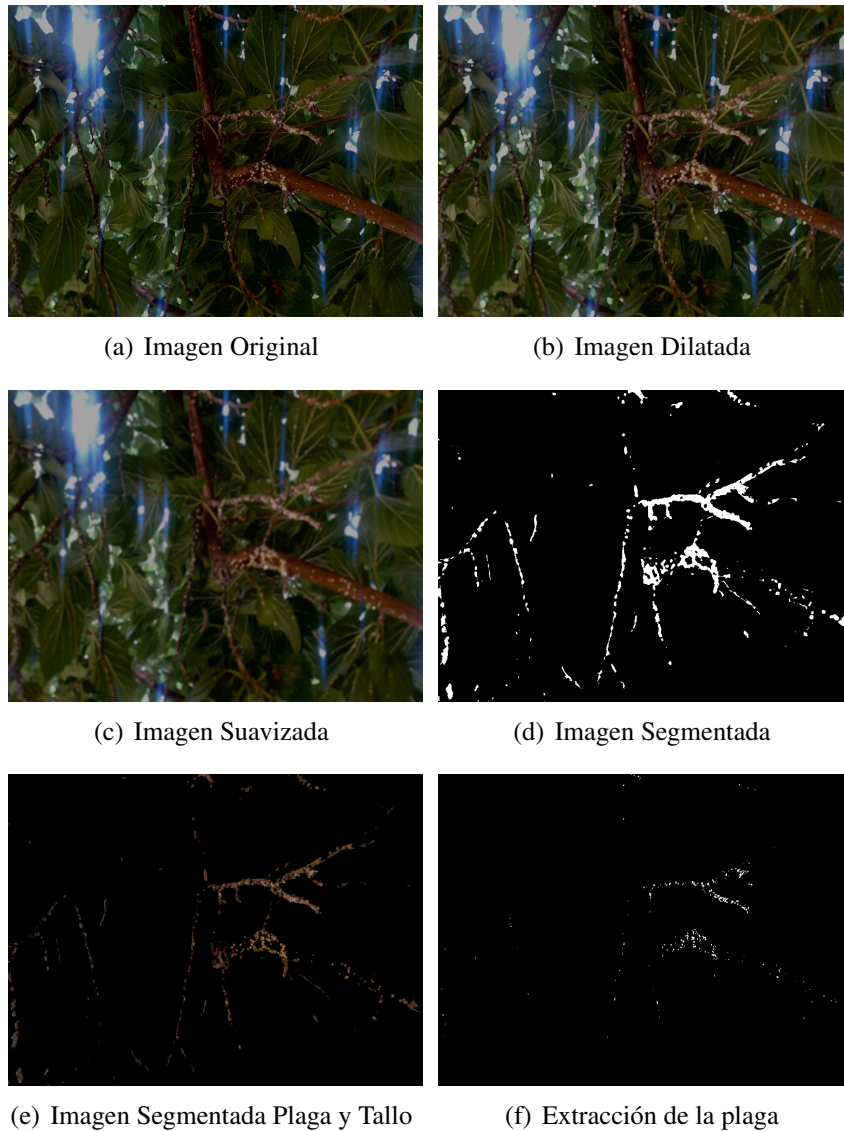


Figura 5.5: Procesamiento de Imágenes del Sistema

Para la primera parte de segmentación se logra obtener exclusivamente todos los píxeles que conforman los objetos pertenecientes al tallo y a la plaga ver Figura (5.5(e)) mediante el filtro y umbralización de colores HSV, siendo éste el método mas adecuado respecto a tratamiento de plantas, la ventaja que se aprovechó de éste filtro fué con mas énfasis a la eliminación de la luz del día que es un color que juega en contra de nuestro objetivo principal ya que en varias pruebas del sistema dejaba o proporcionaba falsos positivos a reflejar en las hojas o tallos claros que

hacían que parezca plaga en la imagen capturada, entonces se optó por encontrar horarios para capturar imágenes con menos brillo pero sin perder claridad en la imagen. A su vez nos ayudo a eliminar los verdes hasta cierto grado dejando solos al tallo y plaga.

Finalmente se optó por adoptar como criterio de selección los valores de los canales RGB tanto de la planta como de la plaga *Planococcus Citri*, aunque existen semejanzas en el color entre el tallo y ciertas hojas es fácil diferenciar por color a la plaga, lo que quiere decir que los valores de los canales presentan diferencias, con esta característica no se presentan restricciones de diferencias de tamaño, volumen, predominancia y forma, aunque si de fase ya que teóricamente esta plaga inicialmente es de color blanco y luego pasa a ser adulta con un color amarillo. Esto genera mayor confiabilidad y conservación de las características. Lo que da paso a la extracción del índice de porcentaje referencial de plaga encontrado y posteriormente a enviar por el sistema de comunicación. Este índice porcentual se debe a las características y limitancias de la tecnología LoRa que se comentaran mas adelante.

En la tabla (5.3) se muestran los resultados de 15 pruebas variando la resolución de la imagen adquirida y dando como resultados el índice porcentual de luminosidad y el tiempo de procesamiento. Se consideró tomar el 100 por ciento a 5100 píxeles, ya que el mejor valor encontrado de varias imágenes analizadas en laboratorio fueron 5084 píxeles en alta calidad y a pesar del tiempo requerido en procesamiento tiene mayor capacidad en reconocer los píxeles de la plaga y las siguientes pruebas se rigen a este valor como tope máximo de plaga encontrada. Respecto al tiempo de procesamiento es casi constante para cada tipo de resolución.

PN°	L1	T1	L2	T2	L3	T3	L4	T4
1	54.84	27.95	29.53	12.54	26.34	10.55	15.46	6.48
2	97.76	28.35	40.70	12.55	35.78	10.74	21.74	6.58
3	99.68	28.45	42.11	12.55	36.93	10.81	21.81	6.63
4	92.43	28.34	41.50	12.63	36.03	10.80	20.48	6.60
5	87.65	28.37	39.52	12.62	33.21	10.73	20.35	6.62
6	92.74	28.32	40.49	12.51	36.38	10.79	20.53	6.63
7	51.79	28.57	26.71	12.44	26.57	10.89	16.14	6.62
8	94.14	28.44	41.39	12.51	36.11	10.75	21.63	6.61
9	55.02	28.23	28.57	12.46	28.98	10.75	16.22	6.58
10	91.40	28.32	41.97	12.52	36.34	10.69	20.11	6.58
11	75.26	28.45	33.15	12.54	33.59	10.74	18.31	6.58
12	86.18	29.28	37.53	12.45	35.09	10.75	20.92	6.60
13	80.52	28.03	38.47	12.46	33.66	10.75	20.02	6.60
14	85.37	28.88	39.54	12.40	33.03	10.75	20.84	6.59
15	95.12	28.24	41.94	12.41	36.60	10.76	21.01	6.58

Tabla 5.3: Tabla de Datos de Segmentación de Imágenes del Sistema.

En la tabla (5.3) T1 representa al tiempo requerido para el procesamiento de la imagen y L1 representa Luminosidad hallada para una imagen de alta calidad con resolución 1920X1080 píxeles por imagen, T2 y L2 para una imagen con resolución de 1280X720, T3 y L3 para una imagen con resolución de 1024X768, T1 y L1 para una imagen con resolución de 800X600. Es claro ver que el tiempo de procesamiento reduce en cuanto bajamos la calidad de la imagen sin embargo perdemos reconocimiento de píxeles de plaga al reducir la resolución, y con los valores encontrados para el resto de imágenes con otra resolución que no sea la alta no sobrepasa el 50 por ciento y se puede considerar no aceptable la información. Con respecto a la luminosidad promedio encontrada apreciamos que con alta resolución de la imagen logra detectar la mayor cantidad de píxeles de la plaga y en ocasiones aparecen valores con 50 por ciento de índice de plaga, eso debido a la falta de estabilidad del sistema en físico o por los vientos haciendo mala posición de las imágenes pero que también cumplan las condiciones de luminosidad al momento de adquirirlas.

5.3 Transmisión de Mensajes

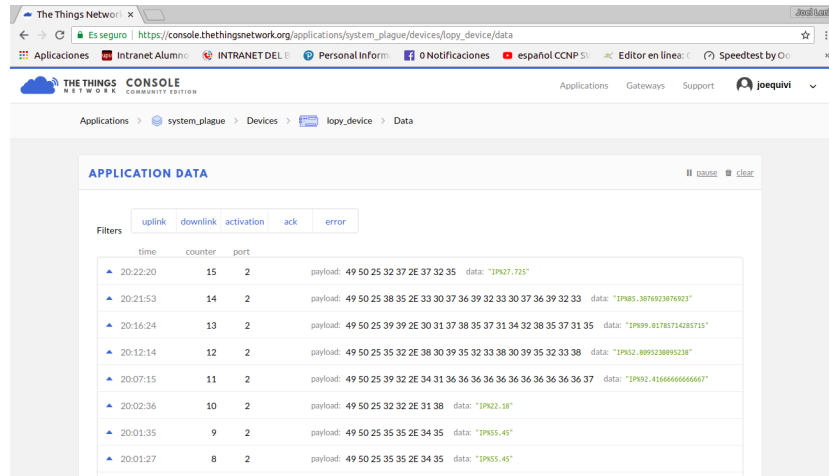
Las ventajas del protocolo LoRaWAN son el bajo consumo y el largo alcance sin embargo para hacer uso de esta infraestructura, tiene sus restricciones, como es el caso del tamaño de los paquetes al enviar, no debe exceder 255 bytes y no se puede enviar mensajes a cada instante, es por esto que el tratamiento de imágenes debe ser resuelta en el nodo y extraer un valor representativo tal es el caso del índice porcentual de plaga en la planta que equivale aproximadamente a 17 bytes. En las pruebas para la transmisión de mensajes por LoRaWAN se enviaron datos cada 5 minutos durante 2 horas de las diferentes imagenes almacenadas en la Raspberry Pi para observar el tiempo de envío de los mensajes. Así los resultados obtenidos fueron un promedio de 14.57 milisegundos. En la tabla ((5.4)) se muestran las pruebas del tiempo de envío de mensajes:

Pr.Nº	T.SendMsg(ms)
1	15.27
2	14.59
3	13.42
4	13.21
5	13.57
6	13.31
7	13.88
8	15.10
9	14.78
10	13.23
11	13.98
12	14.53
13	14.36
14	13.79
15	13.53
16	13.87
17	15.91
18	15.43
19	14.09
20	13.24

Tabla 5.4: Tabla de Tiempo de Envío de Mensajes por LoRaWAN.

En la siguiente Figura (5.6) se muestran los datos recibidos en la aplicación en

donde el usuario tendrá acceso y revisar dicha información en texto plano gracias a la plataforma en el que se puede realizar cambios con los datos de llegada y decodificarlos a caracteres.



The screenshot shows the 'APPLICATION DATA' section of the The Things Network Console. It displays a table of data points for the device 'lopy_device'. The table has columns for 'time', 'counter', 'port', 'payload', and 'data'. The data points are as follows:

time	counter	port	payload	data
20:22:20	15	2	payload: 49 50 25 32 37 2E 37 32 35	data: "1PK27.725"
20:21:53	14	2	payload: 49 50 25 38 35 2E 33 30 37 36 39 32 33 30 37 36 39 32 33	data: "1PK85.3076923076923"
20:16:24	13	2	payload: 49 50 25 39 39 2E 30 31 37 38 35 37 31 34 32 38 35 37 31 35	data: "1PK09.81785714285714"
20:12:14	12	2	payload: 49 50 25 35 32 2E 38 30 39 35 32 33 38 30 39 35 32 33 38	data: "1PK32.809523895238"
20:07:15	11	2	payload: 49 50 25 39 32 2E 34 31 36 36 36 36 36 36 36 36 36 37	data: "1PK92.416666666666667"
20:02:36	10	2	payload: 49 50 25 32 32 2E 31 38	data: "1PK22.18"
20:01:35	9	2	payload: 49 50 25 35 35 2E 34 35	data: "1PK55.45"
20:01:27	8	2	payload: 49 50 25 35 35 2E 34 35	data: "1PK55.45"

Figura 5.6: Llegada de datos a la Aplicación TTN

5.4 Prototipo



Figura 5.7: PROTOTIPO FINAL

Capítulo 6

Conclusiones y Trabajo Futuro

En este punto se realiza un balance general del desarrollo llevadas a cabo en el presente trabajo de fin máster y las conclusiones obtenidas. Además se analizan una serie de perspectivas a futuro relacionados con el tema de investigación abordados así como sus posibilidades de ampliación.

6.1 Conclusiones

En este trabajo se logró integrar las tecnologías, protocolos y conceptos propuestos para el sistema IoT tal es el caso de LoRa, MQTT, OpenCV, Fog Computing, MycroPython con el objetivo de detectar un índice referencial de plaga en la planta y transmitir dicha información a la nube para el usuario.

Los dispositivos seleccionados como a Raspberry Pi, Lopy de Pycom, etc., trabajan sobre la infraestructura TTN que satisfacen los objetivos propuestos en principio, sin embargo el coste, consumo y largo alcance tiene algunas limitaciones en la actualidad respecto al tamaño de mensajes que se puede transmitir y tiempo de transmisión de extremo a extremo, obligando al sistema a reducir a un escalar la información a transmitir u buscar un escenario que se acomode a éstas principales características.

La característica del tamaño de mensajes a transmitir por LoRa lleva al sistema a proponer el concepto de Fog Computing en el que la Raspberry Pi trabaja la parte de procesamiento de imágenes y extracción de información resumida a un índice de afección porcentual de plaga en la planta, resolviendo esta parte en el nodo final para luego cumplir las normas del protocolo LoRaWAN y plataforma TTN.

El escenario propuesto fue una zona abierta con mucha variabilidad de luz,

temperatura, clima, altura, etc., siendo la luminosidad un problema crucial que alteraba los parámetros de la configuración de la cámara y parámetros en los algoritmos trabajados en OpenCV. Por un lado se sigue una estrategia sugerida en [7] que trata de aplicar condiciones de luminosidad en la adquisición, esta parte se trabajó con la luminancia promedio de una imagen basada en el modelo YUV , escogiendo entre los mejores resultados de reconocimiento de plaga analizadas, un rango de luminancia tolerable entre 36 a 42 siendo 38 el valor mejor presentado para la adquisición de las demás imágenes. Además que éstos valores de luminancia promedio mejor presentados son encontrados en periodos muy específicos durante el día y las pruebas mostradas en los resultados fueron tomadas en días despejados, poco viento, estación verano, con mejor posición del sol a las 19:00 a 21:00 horas. Por otro lado, para las cámaras que no permiten una modificación de los parámetros como el tiempo de exposición, ganancia, brillo, etc, no es factible trabajar este tipo de escenarios.

En lo referente a las técnicas de visión por computador empleados para el procesamiento de imágenes se usó OpenCV ya que tiene una variedad de algoritmos resueltos para tratar las imágenes. De acuerdo a la zona de interés que se tenía y el objeto a extraer, los mejores algoritmos empleados y que llevaron a una mejor respuesta son modelos de color HSV y RGB debido a que el objeto no tiene una forma, tamaño, un patrón de expansión mas que el color que los diferencia de la planta. Como conclusión en base a la luz y una cámara de bajo coste se prefiere adquirir imágenes con poca luminancia debido a que la luz puede producir falsos positivos.

Los resultados de adquisición de imágenes muestran que el sistema captura imágenes de alta calidad en un promedio de 11.5 segundos como mínimo aplicando las condiciones de luminosidad y lógicamente en menor tiempo la captura de imágenes de menor resolución sin embargo los resultados del índice deafección porcentual de plaga es muy inferior a una imagen de alta calidad debajo de los 50 por ciento, quedando como conclusión que a mejor calidad se detecta mayor cantidad de píxeles de plaga pero se pierde mucho tiempo en el procesamiento.

El transporte de los mensajes basados en el protocolo LoRaWAN sobre la plataforma TTN requiere ciertos procedimientos de solicitud e identificación lo que hace un poco tardío la conexión pero al momento de transmitir los mensajes previo a una conexión los resultados son tiempos muy cortos en milisegundos. Sin embargo de acuerdo al consumo de energía y disponibilidad de la red en un caso de varios nodos, el sistema solo debe enviar una a dos veces al día lo que hace que el sistema debe volver a conectarse y registrarse en el gateway y servidor cloud para enviar sus mensajes entonces sumados todo el tiempo de conexión y proce-

samiento el transmitir demora unos 15 segundos promedio según los resultados obtenidos. Para nuestros resultados se realizó pruebas en vacío sin ningún otro nodo enviando mensajes.

6.2 Trabajos Futuros

A la vista de los resultados obtenidos y teniendo en cuenta la problemática del trabajo en entornos de exterior, la investigación llevada a cabo ha dado lugar a la aparición de nuevas líneas de interés a ser consideradas en el futuro:

La seguridad en los mensajes está basado en las claves sin embargo pueden haber mejoras en protocolo MQTT y probar con los otros puertos de cifrado SSL. Además se puede obviar el uso MQTT insertando un hardware de módulo LoRa directamente a la Raspberry Pi.

En la adquisición de imágenes se puede añadir un sistema experto en software en base patrones a seguir para que el sistema tome mejores decisiones con el objetivo de reducir tiempo y tomas repetitivas de capturas a la zona de interés.

En cuanto a la cámara seleccionada existen otras de mejor utilidad específicas y de tratamiento de imágenes para estos entornos que son compatibles a sistemas operativos de software libre.

De cara al futuro, para otros posibles proyectos o una ampliación de este, sería interesante ampliar el estudio sobre las bondades del módulo LoRa, como el consumo para distintos niveles de potencia de transmisión, un número mayor de canales LoRaWAN y variaciones en el ciclo de trabajo de cada uno de esos canales.

Aplicación del sistema en otros cultivo, los métodos propuestos en la presente memoria se han diseñado específicamente para la detección de blancos en el tallo de una planta de frutos cítricos de campo abierto con fácil acceso pero poca seguridad para realizar pruebas. Una opción interesante a la par que viable es la adaptación de los métodos a otros cultivos con mas disponibilidad de realizar pruebas.

Para el tratamiento de imágenes se puede construir una base de datos y un software de aprendizaje para el reconocimiento del objeto de interés.

Sería interesante trabajar pruebas de alcance de LoRaWAN y trabajar con una

Wireless Sensor Networks (WSN) de nodos LoRa.

Se puede crear una interfaz gráfica para trabajar el sistema con la participación del usuario y cambiar de modo automático a manual.

Una propuesta interesante sería transmitir imágenes por LoRa.

Lista de Acrónimos

ABP Activation By Personalization.

AP Agricultura de Precision

GRC Grupo de Computadores y Redes

HSV Hue Saturation Vue

IO Innovation in the Open

IoT Internet of Things

IP Internet Protocol

LoRa Long Range

LoRaWAN Long Range Wide Area Network

LPWAN Low Power Wide Area Network

M2M Machine to Machine

MQTT Message Queuing Telemetry Transport

OpenCV Open Source Computer Vision Library

OTAA Over The Air Activation

REPL Read Evaluate Print Loop

RGB Red Green Blue

ROI Region of Interest

TTN The Things Network

UPV Universitat Politècnica de Valencia

WSN Wireless Sensor Networks

Bibliografía

- [1] Jose Manuel Fernández. (Enero, 2018). [Online]. Available: <http://www.grupofertiberia.com>
- [2] R. P. Foundation. [Online]. Available: <https://www.raspberrypi.org/magpi/raspberry-pi-3-specs-benchmarks/>
- [3] S. H. R. I. (2002). [Online]. Available: www.agriculturadeprecision.org
- [4] D. V. Valdiviezo, “Diseño de una Red una Red de Sensores Inalámbrica para Agricultura de Precisión,” Master’s thesis, Tesis para optar el título de Ingeniero Electrónico Pontificia Universidad Católica del Perú., Junio 2009.
- [5] Gartner. [Online]. Available: <https://www.gartner.com/en>
- [6] R. A. R. Gómez, “Diseño De Un Sistema De Registro De Imágenes Orientado A La Agricultura De Precisión,” Master’s thesis, Tesis para optar el título de Ingeniero Electrónico Pontificia Universidad Católica del Perú., Septiembre 2009.
- [7] J. M. G. Hernández, “Sistema de visión para agricultura de precisión: Identificación en tiempo real de líneas de cultivo y malas hierbas en campos de maíz,” Ph.D. dissertation, Tesis Doctoral. Universidad Complutense de Madrid Facultad de Informática Departamento De Ingeniería Del Software E Inteligencia Artificial Madrid, 2015.
- [8] J. E. B. Escudero, “REGISTRO DE IMÁGENES PARA AGRICULTURA DE PRECISIÓN USANDO LENGUAJE C,” Master’s thesis, Tesis para optar el título de Ingeniero Electrónico Pontificia Universidad Católica del Perú., Marzo 2012.
- [9] M. E. G. Daniel Chora Garcia 1, SGuido Álvarez Martínez 2, “Raspberry pi y arduino: semilleros en innovación tecnológica para la agricultura de precisión,” Instituto Tecnológico Superior Babahoyo, Babahoyo, Ecuador, Tech. Rep., 01 2018.

-
- [10] M. D. P. Liceaga, "Diseño y Despliegue de arquitectura para la recogida y presentación de medidas de sensores IoT," Master's thesis, Trabajo de Fin de Master en Universidad del País Vasco., 2017.
- [11] P. P. Garcés, "Redes de Área Extensa para aplicaciones de IoT: modelado de comunicaciones Sigfox," Master's thesis, Trabajo de Fin de Máster en Politècnica Universitat de València., Julio 2017.
- [12] K. C. Z. T. Y. Z. Y. Z. Jianhua He, Jian Wei, "Multitier fog computing with large-scale iot data analytics for smart cities," *IEEE Internet of Things Journal*, vol. 5, no. 2, April 2018.
- [13] Semtech. (2018). [Online]. Available: <https://www.semtech.com/lora>
- [14] V. P. Alexandru Lavric, "Internet of things and loraTM low-power wide-area networks: A survey," *IEE International Symposium on Signals, Circuits and Systems (ISSCS)*, September 2017.
- [15] M. D. P. Liceaga, "Diseño y Despliegue de arquitectura para la recogida y presentación de medidas de sensores IoT," Master's thesis, Trabajo para optar el título Máster Universidad del País Vasco, 2017.
- [16] LoRa Alliance. (2018) <https://lora-alliance.org/lorawan-for-developers>.
- [17] A. I. P. Alexandru Lavric, "Lorawan communication protocol: The new era of iot," *IEE International Conference on Development and Application Systems (DAS)*, June 2018.
- [18] G. P. y. A. d. I. E. Enrique Alegre, *Conceptos y Métodos en Visión por Computador*, 1st ed., España, Junio 2016.
- [19] P. B. K. Rajesh S. Sarkate, N. V. Kalyankar, "Application of computer vision and color image segmentation for yield prediction precision," 2013.
- [20] O. L. R. S. Camilo Andrés Cáceres Flórez, Darío Amaya Hurtado, "Procesamiento de imágenes para reconocimiento de daños causados por plagas en el cultivo de begonia semperflorens link & otto," *Computer*, vol. 64, no. 3, pp. 273–269, 2015.
- [21] O. E. A. P. J. M. S. Andrés Fernando Jiménez López, Melanie Jisell Quiroz Medina, "Diagnóstico de cultivos utilizando procesamiento digital de imágenes y tecnologías de agricultura de precisión," vol. 11, no. 1, pp. 63–71, 2015.

-
- [22] L. U. R. C. Dra. Nora La Serna Palomino, “Técnicas de segmentación en procesamiento digital de imágenes,” *Revista de Ingeniería de Sistemas e Informática*, vol. 6, no. 2, 2009.
- [23] R. M. G. C. A. R. Nayid Triana, Andrés E. Jaramillo, “Thresholding techniques for digital image processing of gem-foils,” *Ingeniería electrónica*, Universidad Antonio Nariño, Bogotá, Colombia, Tech. Rep. 4, 2016.
- [24] F. Garcia-Marí, *PLAGAS DE LOS CÍTRICOS. Gestión Integrada en países de clima mediterráneo*, 1st ed. Valencia España: PHYTOMA-España, 2012.
- [25] Raspberry Pi Foundation. [Online]. Available: <https://www.raspberrypi.org/>
- [26] R. P. Foundation. [Online]. Available: <https://www.raspberrypi.org/guia-completa-raspberry-pi.php>
- [27] Raspberry Pi Foundation. [Online]. Available: <https://www.raspberrypi.org/products/raspberry-pi-3-model-b/>
- [28] Pycom. [Online]. Available: <https://pycom.io/hardware/lopy-specs/>
- [29] Logitech. [Online]. Available: <https://www.logitech.com/es-es/product/hd-pro-webcam-c920>
- [30] MycroPython. [Online]. Available: <http://micropython.org/>
- [31] J. H. Joe Minicho, *Learning OpenCV 3 Computer Vision with Python*, 2nd ed. Birmingham: Packt Publishing Ltd., September 2015.
- [32] OpenCV. [Online]. Available: <http://www.opencv.org/>
- [33] Pypi. [Online]. Available: <https://pypi.org/project/paho-mqtt/>
- [34] Pycom. [Online]. Available: <https://github.com/pycom/pycom-libraries>
- [35] Pypi. [Online]. Available: <http://test.mosquitto.org/>
- [36] Ubuntu. [Online]. Available: <http://manpages.ubuntu.com/manpages/cosmic/man1/v4l2-ctl.1.html>
- [37] OpenCV. [Online]. Available: <https://docs.opencv.org/2.4/modules/imgproc/doc/filtering.html?highlight=gaussianblur#gaussianblur>

-
- [38] B. P. Siti Mushonnifah, Hendro Nurhadi, "Conceptual machine vision design for day and night based on experiment approach," *2017 International Conference on Advanced Mechatronics, Intelligent Manufacture, and Industrial Automation (ICAMIMIA)*, October 2017.
- [39] A. Bovik, *Handbook of Image and Video Processing (Communications, Networking and Multimedia)*, 2nd ed. Academic Press, June 2005.
- [40] TTN. [Online]. Available: <https://www.thethingsnetwork.org/article/the-things-network-architecture-1>
- [41] ——. [Online]. Available: <https://www.newieventures.com.au/blogtext/2018/2/26/lorawan-otaa-or-abpl>
- [42] Pycom. [Online]. Available: <https://forum.pycom.io/topic/429/otaa-connection-to-ttn>
- [43] ——. [Online]. Available: <https://docs.pycom.io/tutorials/lora/lorawan-otaa>