Instituto Tecnológico de Costa Rica

Escuela de Ingeniería Electrónica

Diseño de un sistema de monitoreo para la determinación de la curva de vuelo de la broca en plantaciones de café mediante procesamiento digital de imágenes

Informe de Proyecto de Graduación para optar por el título de Ingeniera en Electrónica con el grado académico de Licenciatura

Karina Vanessa Aguilar Quirós

Cartago, Costa Rica

Junio, 2018

INSTITUTO TECNOLÓGICO DE COSTA RICA

ESCUELA DE INGENIERÍA ELECTRÓNICA

PROYECTO DE GRADUACIÓN

ACTA DE APROBACIÓN

Defensa de Proyecto de Graduación
Requisito para optar por el título de Ingeniero en Electrónica
Grado Académico de Licenciatura
Instituto Tecnológico de Costa Rica

El Tribunal Evaluador aprueba la defensa del proyecto de graduación denominado *Diseño de un sistema de monitoreo para la determinación de la curva de vuelo de la broca en plantaciones de café mediante procesamiento digital de imágenes*, realizado por la señorita Karina Vanessa Aguilar Quirós y, hace constar que cumple con las normas establecidas por la Escuela de Ingeniería Electrónica del Instituto Tecnológico de Costa Rica.

Miembros del Tribunal Evaluador

Ing. Miguel Hernández Rivera

Profesor lector

Ing. Carlos Mauricio Segura Quirós

Profesor lector

Ing. Sergio Arriola Valverde

Profesor asesor

Cartago, junio 2018

Declaratoria de autenticidad

Declaro que el presente Proyecto de Graduación ha sido realizado, en su totalidad, por mi persona, utilizando y aplicando literatura referente al tema e introduciendo conocimientos propios.

En los casos en que he utilizado material bibliográfico, he procedido a indicar las fuentes mediante citas.

En consecuencia, asumo la responsabilidad total por el trabajo de graduación realizado y por el contenido del correspondiente informe final.

Karina Vanessa Aguilar Quirós

Cédula: 1-1584-0765

Resumen

La conexión digital entre objetos utilizados en la vida cotidiana y el Internet (Internet de las Cosas o IoT por sus siglas en inglés), se ha llevado cada vez más a sectores de producción y áreas de industria en los últimos años. La introducción del IoT en la industria cafetalera puede resultar en mejoras en el rendimiento de los procesos y simplificar tareas, como por ejemplo, la detección y el control de plagas que puedan estar atacando a los granos de café y disminuyendo la calidad de las cosechas.

El desarrollo de este proyecto combina el uso de los medios de comunicación remota a través de Internet y los algoritmos de procesamiento digital de imágenes para proponer una solución de un sistema de monitoreo que se utilizará en las plantaciones de café para determinar la curva de vuelo de la plaga conocida como *Hypothenemus hampei* o broca del café.

El sistema de monitoreo implementado es capaz de identificar a más del 90 % de las brocas que son atrapadas por este, y registra los datos para que sean analizados por el usuario. Con la información obtenida, es posible aplicar medidas de control en la época del año en la cual se registró mayor presencia de la plaga, para reducir pérdidas económicas y evitar un mayor daño a los suelos.

Palabras clave: broca del café, conexión remota, detección de plagas, procesamiento digital de imágenes.

Abstract

The digital connection between everyday objects and Internet (Internet of Things or IoT), has been increasingly taken to production sectors and industry areas in the last years. The introduction of the IoT in the industry of coffee plantations can result in production improvements and simplify some tasks, such as the detection and control of pests that may be attacking coffee beans and decreasing the harvests's quality.

The development of this project combines the use of remote means of communication through the Internet and digital image processing algorithms to propose a monitoring system which will be used in coffee plantations to determine the flight curve of the plague known as *Hypothenemus hampei* or coffee borer.

The system implemented is able to identify more than 90 % of the coffee borers that are trapped by it, and records the data for analysis by the user. With the information obtained, it is possible to apply control measures at the time of the year in which the presence of the pest was higher, in order to reduce economic losses and avoid further damage to the soil.

Keywords: coffee berry borer, digital image processing, pest detection, remote connection.

A Axel, por ser el motor que me impulsó a lograr este objetivo.

Agradecimientos

Agradezco a mis profesores M.Sc. Sergio Arriola Valverde, Dr.-Ing, Renato Rímolo Donadio y M.Sc. Aníbal Coto Cortés, que por todo el apoyo que me brindaron y la confianza que siempre depositaron en mí a lo largo de mi carrera, los puedo llamar amigos.

A mis familiares más cercanos, por darme siempre cuanto apoyo estuvo en sus manos y porque sé que se enorgullecen de este triunfo tanto, o más que yo.

Al Ing. Daniel Ramírez Valerio, representante del Instituto del Café de Costa Rica, por su asesoría, apoyo e interés durante el desarrollo de este proyecto.

A mi mejor amiga, Alejandra, por innumerables experiencias compartidas, apoyo incondicional, y enseñarme lo que significa tener una hermana por elección.

A Grettel, Javier, Daniel León, Amit y Rolando; les agradezco por su apoyo y compañía durante todos estos semestres, y por hacer mi vida en la universidad más bonita.

Y quiero agradecer a Dios, porque sé que sin Sus bendiciones nada de esto habría sido posible.

Índice

Índice de figuras

Índice de tablas

1. Introducción

El sector de producción cafetalera es una de las principales actividades de movilización económica en Costa Rica, por lo que una disminución en su producción podría generar un fuerte impacto económico en el país. Desde sus inicios, el cultivo de café permitió hacer surgir al territorio nacional tanto económica como socialmente, y hoy día supone uno de los principales productos de exportación y sector de empleo para costarricenses e inmigrantes habitantes en nuestro país [1].

Una de las limitaciones a las que se enfrenta la industria cafetalera en cuanto al aumento del rendimiento y de la producción, es la aparición de plagas como la broca del café (*Hypothenemus hampei*). La broca (figura 1.1) ha provocado fuertes pérdidas económicas a nivel mundial, donde estudios han confirmado que se ha disminuido el rendimiento de las producciones en hasta un 55 % en algunas zonas de cultivo [1]. Esta es una plaga muy difícil de controlar debido a su reducido tamaño (con un máximo de 1.5 mm en adultos), y la regulación actual en la aplicación de medidas de control no logra evitar un impacto negativo en la economía de la zona de producción cafetalera con presencia de broca.

Figura 1.1: Broca del café (*Hypothenemus hampei*) [1]

Según estimaciones del Instituto del Café de Costa Rica (Icafé), desde el año 2012 la producción de café mostró caídas en la cosecha anual, debido principalmente a la afectación que las plagas ocasionan sobre el grano [2] y sobre las cuales no se tiene un efectivo sistema de control, razón por la cual este estudio propone el diseño e implementación de un sistema de monitoreo automatizado que permita obtener estimaciones de la época del año en la que aparece mayor cantidad de broca. Con esta información es posible regular la aplicación de medidas de control de plagas, y con ello evitar posibles daños en la economía y el medio ambiente.

El sistema propuesto utiliza una trampa sencilla para simular el medio de retención de brocas en un papel adhesivo que consiste en la zona de muestreo del proyecto. Posteriormente, se realiza la toma de fotografías de dicha zona de muestreo a las horas indicadas por el usuario, y el envío de estas imágenes a través de un sistema empotrado e Internet hacia

un servidor web. Las fotografías capturadas son procesadas en el sistema empotrado, el cual ejecuta un algoritmo de procesamiento digital de imágenes que indica al usuario la cantidad de brocas que se encontraron en la imagen al momento de la toma de la fotografía, a través de la carga de los datos a una plataforma *IoT*. Una etapa posterior a la realización de este proyecto, consiste en la optimización y automatización del medio de retención de muestras.

La verificación del sistema de monitoreo se realizará en conjunto con los encargados de las fincas cafetaleras del Icafé. El desarrollo del algoritmo de procesamiento digital de imágenes y las pruebas de envío de datos se realizarán en las instalaciones de la Escuela de Electrónica del Instituto Tecnológico de Costa Rica.

La implementación de este sistema de monitoreo involucra principalmente tres grandes conceptos: la tecnología del Internet de las Cosas *(IoT)*, la cual permite la interacción entre usuarios y dispositivos de manera remota; el procesamiento digital de imágenes, la cual es un área que permite grandes facilidades mediante el estudio detallado de las fotografías; y la agricultura de precisión, la cual pretende utilizar herramientas de índole técnico para mejorar la productividad y eficiencia en el sector agrícola.

1.1. Objetivos y estructura del documento

El objetivo principal de este proyecto es diseñar un sistema de monitoreo que brinde la información necesaria para determinar la época del año en la que aparece la mayor cantidad de broca en plantaciones de café, utilizando como tecnología principal el procesamiento digital de imágenes. El sistema de monitoreo debe capturar una imagen de las brocas atrapadas en la zona de muestreo, procesar las fotografías adquiridas mediante un algoritmo de procesamiento digital de imágenes, y enviar la información recopilada a un servidor web a través de Internet. Como parte final del proyecto, se requiere verificar el funcionamiento completo del sistema simulando el campo, incluyendo el envío de datos de forma remota desde el nodo de muestreo hacia un servidor web.

El desarrollo del documento comienza en el capítulo 2, donde se explica la teoría necesaria para comprender el desarrollo de la solución; en el capítulo 3 se describe detalladamente la solución propuesta y su metodología de diseño; en el capítulo 4 se presentan los resultados y la Verificación del sistema de monitoreo en cada una de sus partes, así como un breve análisis del costo de diseño e implementación del sistema a nivel de prototipo; y en el capítulo 5 se exponen las conclusiones y recomendaciones para posteriores trabajos en una línea de investigación similar.

2. Marco teórico

Este capítulo presenta los conceptos y detalla los temas más relevantes para la comprensión de la información presentada en este estudio. Se recomienda al lector consultar las referencias bibliográficas [3] - [24] si desea ampliar la información con respecto a algún tema en específico.

2.1. Internet de las cosas (IoT)

Este concepto, algunas veces llamado *Internet de los Objetos*, ha adquirido gran importancia en la industria tecnológica durante los últimos años. En términos generales, IoT consiste en la interconexión de dispositivos y (o) productos para lograr que su funcionamiento sea más automatizado e inteligente [3].

La forma más simple de IoT puede dividirse en tres elementos [3]:

1. Las cosas: Involucra una gran cantidad de productos que pueden controlarse automáticamente utilizando esta tecnología, donde las tendencias actualmente son los automóviles, teléfonos, casas y robots. La capacidad de estos productos se aumenta mediante el uso de sensores, motores, o lo que se considere pertinente para mejorar sus características y adaptarlas a las necesidades humanas.

2. La red: Es estrictamente necesaria en la implementación de la tecnología IoT, pues es la base de la comunicación entre los dispositivos. En las redes más estables, se utilizan routers de alta velocidad, switches y tecnología de gateway.

3. La nube: Está compuesta por los centros de datos y el software que compone la mayor parte de la lógica de los negocios en el IoT, es decir, aloja tanto a los servidores como a la infraestructura tecnológica, que incluye el soporte del equipo y de la red, así como el control de los sistemas en el ambiente.

El Internet ha tenido un fuerte impacto en la vida humana [4], por lo que es considerado una de las herramientas más poderosas actualmente. Mediante el IoT, los objetos pueden tomar decisiones y comunicarse con otros dispositivos con posibilidades prácticamente ilimitadas en cuanto a los dispositivos que se pueden controlar y sus aplicaciones finales, como se define en la figura 2.1. Con base en este concepto, se afirma que IoT brinda la posibilidad de un futuro inteligente en donde la mayoría de los dispositivos podrán ser controlados remotamente mediante el uso de computadores [5].

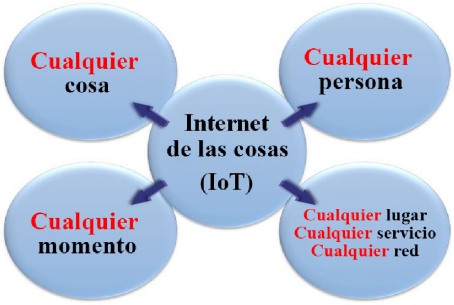


Figura 2.1: Concepto del Internet de las Cosas. Adaptado de [3].

2.2. Procesamiento digital de imágenes

El interés por el estudio de este concepto tiene dos principales áreas de aplicación [6]: la mejora de la información en las fotografías para la interpretación humana; y el procesamiento de la imagen para el almacenamiento de datos, su transmisión y representación para la percepción autónoma mediante equipo computarizado.

Para definir el procesamiento digital de imágenes se debe recurrir antes al concepto de imagen, la cual puede verse como una estructura de datos descrita como una función bidimensional $f(x, y)$, donde x y y son coordenadas espaciales y la amplitud de f en cualquier punto (x, y) se denomina intensidad de la imagen en dicho punto. Se puede calificar una imagen como imagen digital si los valores de x, y y f son todos finitos. Cuando una imagen digital es procesada a través de una computadora digital, se está trabajando en el campo del procesamiento digital de imágenes (en adelante llamado **DIP**) [8].

Los seres humanos estamos limitados a la banda visual del espectro electromagnético, sin embargo, las máquinas de imagen (por ejemplo, las computadoras) cubren desde *ondas gamma* hasta *radio ondas*, lo cual les permite trabajar con imágenes generadas por fuentes que van más allá de la asociación humana (por ejemplo, los ultrasonidos y la microscopía electrónica), dando una gran ventaja a las aplicaciones de DIP [8].

El primer trabajo relacionado con DIP involucró técnicas computacionales para mejorar la calidad de las imágenes que se transmitían desde una sonda espacial, por lo que el primer algoritmo de DIP utilizado estuvo relacionado con la corrección en la distorsión de las imágenes [5]. A partir de allí, se ha trabajado con otros algoritmos y conceptos sobre las fotografías que pueden ser estudiados con el objetivo de extraer de estas información importante para la mejora en la eficiencia del tratamiento de los datos en múltiples áreas. Algunos de estos conceptos se muestran a continuación.

2.2.1. Formación de imágenes

Las imágenes son adquiridas mediante el uso de una fuente de iluminación y la absorción o reflexión de la energía proveniente de dicha fuente por los elementos de la escena a capturar [5]. Las fuentes de iluminación pueden ser de energía acústica, cinética en haces de partículas, mecánica, electromagnética, entre otras; así mismo, los elementos de escena pueden corresponder a objetos no tradicionales como moléculas o el cerebro humano. Para capturar la información se utilizan sensores, los cuales se escogen dependiendo del tipo de fuente utilizada. Una imagen que se puede visualizar digitalmente es un arreglo rectilíneo de puntos de muestra conocidos como píxeles, en donde cada uno puede contener hasta tres muestras y define características importantes en cuanto a su tratamiento y resolución [9].

2.2.2. Representación de las imágenes

Para poder aplicar algoritmos sobre imágenes y desarrollar operaciones sobre estas, es necesario representarlas matemáticamente. Esta representación puede realizarse de cuatro formas básicas: mediante funciones, matrices, conjuntos y (o) grafos [7].

- Funciones: Una imagen definida como función en el espacio discreto se representa de la siguiente manera:

$$f : \mathbb{X} \to \mathbb{R}^n \tag{1}$$

  donde $\mathbb{X}$ es un conjunto de posiciones válidas de los píxeles en un espacio de d dimensiones, tal que $\mathbb{X} = \{0, 1, ..., D-1\}^d$. Estos conjuntos son mapeados a otro conjunto de valores vectoriales en $\mathbb{R}^n$ donde n representa la dimensión de la señal.

- Matrices: Esta representación es más conveniente debido a la facilidad de su procesamiento matemático. Para el caso de una imagen bi-dimensional, la función f se puede representar mediante la matriz $\mathbf{F}$, tal que:

$$\mathbf{F} = \begin{bmatrix} f_{0,0} & f_{0,1} & \cdots & f_{0,C-1} \\ f_{1,0} & f_{1,1} & \cdots & f_{1,C-1} \\ \vdots & \vdots & \ddots & \vdots \\ f_{R-1,0} & f_{R-1,1} & \cdots & f_{R-1,C-1} \end{bmatrix} \tag{2}$$

  donde R representa el número de filas y C el número de columnas.

- Conjuntos: Para trabajar únicamente sobre grupos de píxeles se utiliza la representación como conjuntos, donde cada uno está compuesto por un píxel $\mathbf{p}$ representado como una tupla $< \mathbf{x}, \mathbf{c} >$, donde $\mathbf{x}$ es un vector de posición y $\mathbf{c}$ un vector que representa la

composición espectral en términos de energía. Se utilizan los operadores "pos" y "val" para extraer la posición y el valor de los píxeles, respectivamente, de modo que:

$$\begin{aligned} pos(p) &= pos(< \mathbf{x}, \mathbf{c} >) = \mathbf{x} \\ val(p) &= val(< \mathbf{x}, \mathbf{c} >) = \mathbf{c} \end{aligned} \tag{3}$$

y la imagen puede definirse como:

$$I = \{\mathbf{p} \mid \mathbf{p} =< \mathbf{x}, \mathbf{c} >, x \in \mathbb{X}, \mathbf{c} = f(\boldsymbol{x})\} \tag{4}$$

- Grafos. Esta manera de representar las imágenes es utilizada en los algoritmos más complejos, por lo que su uso se orienta principalmente al área de análisis de imágenes y visión por computador, donde uno de los fines principales es detectar estructuras u objetos visuales con determinadas características en la imagen.

2.2.3. Conceptos de imágenes

Antes de adentrarse en el análisis de los algoritmos de DIP, es necesario recurrir a conceptos básicos que caracterizan una imagen digital, y en los cuales se fundamentan las operaciones más empleadas en este estudio [7].

- Tamaño: El tamaño T de una imagen se refiere a la cantidad exacta de píxeles que componen la imagen en cada una de sus dimensiones (Ejemplo: 640×480 píx).
- Dimensiones: Las dimensiones físicas D dependen del medio en el que se represente y (o) visualice la imagen, y no de la cantidad de píxeles que esta contenga (Ejemplo: 15×30 cm).
- Resolución: La resolución espacial ρ se define como la cantidad de píxeles por unidad de área, definición que permite relacionar las dimensiones físicas de la representación con el tamaño de la imagen, mediante

$$D = T \times \rho \tag{5}$$

- Resolución de intensidad: En una imagen digital, los píxeles pueden tomar distintos valores dependiendo del nivel de luz de la imagen. Esta escala de luz se puede representar mediante un número de bits, lo cual deriva el concepto de *resolución de intensidad*.
- Tipos de imagen: Los dos tipos básicos de imágenes existentes son las imágenes rasterizadas y vectoriales, donde la característica principal de las imágenes vectoriales es que se puede aumentar arbitrariamente el número de píxeles por unidad de área sin alterar el tamaño de la imagen, mientras que las imágenes rasterizadas están compuestas únicamente por celdas rectangulares con valores de píxeles previamente asignados.

A continuación se muestran los formatos más utilizados para cada uno de los tipos de imagen mencionados [7]:

- Imágenes vectoriales: SVG (*Scalable Vector Graphics*), WMF (*Windows Meta File*), EMF (*Enhanced Metafiles*), XFig y OpenDocument Graphics [10]-[11].
- Imágenes ráster: BMP (*Bitmap*), PNG (*Portable Network Graphics*), TIFF, GIF y JPEG, siendo este último caracterizado por la disminución del espacio de almacenamiento sin pérdidas significativas [7].

2.2.4. Operaciones sobre imágenes

Los algoritmos de DIP son básicamente un conjunto de operaciones que se aplica sobre la representación de las imágenes (comúnmente matrices). Estas operaciones pueden ser aritméticas o espaciales.

- Vecindades de píxeles: Los algoritmos de DIP se aplican tanto para un píxel central como para los que se encuentran a su alrededor. La figura 2.2 ilustra este concepto para una vecindad de 4 píxeles (a) y una de 8 píxeles (b).

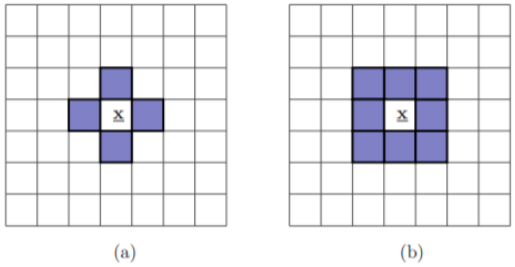


Figura 2.2: Vecindad de: (a) 4 píxeles, (b) 8 píxeles [7].

- Operaciones aritméticas. A continuación se muestran las operaciones básicas realizadas entre píxeles en el procesamiento digital de imágenes [7].

  - Suma: Esta operación se utiliza principalmente para reducir el ruido en las imágenes, en un proceso usualmente llamado *denoising*.

$$s(\underline{x}) = f(\underline{x}) + g(\underline{x}) \tag{6}$$

  - Diferenciación: Esta operación es utilizada para detectar cambios en secuencias de imágenes.

$$s(\underline{x}) = f(\underline{x}) - g(\underline{x}) \tag{7}$$

- Multiplicación: Esta operación es la más utilizada en algoritmos que utilizan *máscaras* como fundamento en sus procesos.

$$s(\underline{x}) = f(\underline{x}) \times g(\underline{x}) \tag{8}$$

2.2.5. Operaciones sobre valores de píxeles

Los algoritmos para procesamiento de imágenes utilizados en este estudio, tienen su fundamento matemático en los siguientes conceptos relacionados con operaciones sobre valores de píxeles, las cuales realizan una modificación del valor de cada píxel en función únicamente del valor correspondiente en la imagen destino [7]:

- Histogramas: Permiten visualizar la distribución estadística del valor del conjunto de píxeles y el comportamiento del contraste de la imagen. La figura 2.3 (a) muestra la distribución de píxeles en una imagen en la que no se ha aplicado una mejora en los niveles de contraste, mientras que en la figura 2.3 (b) se evidencia una distribución más uniforme del rango de píxeles debido a la optimización en el contraste de la fotografía utilizada.

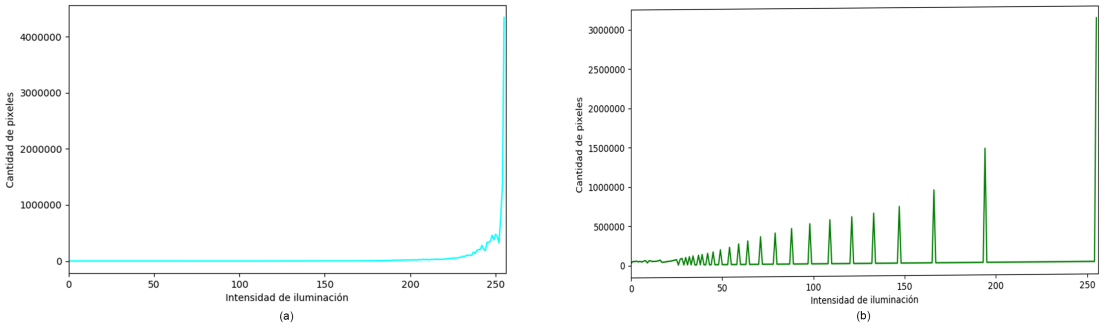


Figura 2.3: Histograma para una imagen (a) sin mejora de contraste (b) con mejora de contraste.

El histograma de una imagen en niveles de gris puede ser representado matemáticamente por [6]:

$$h(r_k) = n_k \tag{9}$$

donde r_k representa un nivel de gris y n_k el número de píxeles en la imagen que tienen dicho nivel. Comúnmente el histograma se normaliza, lo cual se logra mediante:

$$p(r_k) = \frac{n_k}{n} \tag{10}$$

donde n es el número total de píxeles en la imagen.

- Ecualización: Al ecualizar una imagen se busca que el valor de los píxeles en cuanto a su probabilidad de aparición en la imagen sea homogéneo, es decir, busca una distribución

similar a la mostrada en la figura 2.3 (b) y con ello una mejora en el contraste de la imagen. Su implementación matemática utiliza la ecuación del histograma normalizado descrito en la ecuación 10, y se representa mediante:

$$s_k = \sum_{j=0}^{k} p_r(r_j) = \sum_{j=0}^{k} \frac{n_j}{n} \tag{11}$$

donde $k = 0, 1, 2, ..., L-1$ y L representa la cantidad de niveles de gris.

- Convolución: Para realizar la operación de convolución, es necesario definir un impulso unitario aplicable a imágenes bi-dimensionales, el cual se denomina impulso espacial unitario y se define como:

$$\delta(x, y) = \delta(x)\delta(y) = \begin{cases} 1 & \text{si } x = y = 0 \\ 0 & \text{en el resto} \end{cases} \tag{12}$$

En una imagen ráster dividida por celdas rectangulares y haciendo uso del impulso unitario espacial, es posible obtener un único píxel de la imagen mediante [7]:

$$f(x, y) = f(u, v)\delta(x - u, y - v) \tag{13}$$

donde (u, v) son variables que representan la localización de los píxeles deseados.

Dado que la ecuación 13 permite obtener un único píxel de la imagen, se deduce que es posible representar la imagen completa mediante una sumatoria de esta función. Si se asume un sistema lineal e invariante en el tiempo (LTI) y se define $h(x, y) = T[\delta(x, y)]$ como una transformación del sistema, se obtiene la ecuación de convolución:

$$f(x, y) * h(x, y) = \sum_{u,v} f(u, v)h(x - u, y - v) \tag{14}$$

En el procesamiento digital de imágenes, la máscara utilizada para los procesos de convolución es usualmente llamada *kernel*, y su tamaño depende de la aplicación y de la imagen que se utilice.

- Correlación: Esta operación es similar a la convolución, con la diferencia de que no rota la máscara 180° antes de realizar las demás operaciones. Se utiliza como identificador de regiones en imágenes, es decir, la máscara se selecciona como la región de interés de búsqueda en la imagen original, y está dada por:

$$f(x, y) * h(x, y) = \sum_{u,v} f(u, v)h(x + u, y + v) \tag{15}$$

2.2.6. Filtrado de imágenes

La técnica de reducción de ruido es uno de los procesos más utilizados en el procesamiento digital de imágenes [6], y se logra mediante la aplicación de filtros. La manera más simple de reducir ruido en una imagen es mediante el filtro de media móvil o *filtro promediador*, el cual utiliza una máscara descrita por:

$$h_{ma}(n) = \begin{cases} \frac{1}{M} & \text{si } n = 0, ..., M-1 \\ 0 & \text{en otro caso} \end{cases} \tag{16}$$

donde M es el número de píxeles de la máscara, y todos sus componentes tienen el mismo valor.

El resultado de filtrar una imagen dependerá en su totalidad de la máscara o kernel que se utilice para definir el proceso matemático. Existen kernels predefinidos que se utilizan en este tipo de funciones, donde los más utilizados [7] son el kernel de Roberts:

$$h_1(n_1, n2) = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \text{ y } h_2(n_1, n_2) = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{17}$$

el kernel de Sobel:

$$D_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \text{ y } D_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{18}$$

y el kernel de Prewitt:

$$P_x = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \text{ y } P_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \tag{19}$$

Un filtro muy utilizado en la detección de bordes es el *Laplaciano*, definido para una función bidimensional como:

$$\nabla \text{f} = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{20}$$

y utiliza con frecuencia la máscara 3x3 dada por [6]:

$$L = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{21}$$

La forma de la campana de Gauss ocasiona que el filtro gaussiano sea utilizado frecuentemente en la eliminación de ruido, debido a que no contiene cambios abruptos en su forma. Este filtro utiliza una máscara que se calcula modelando la función gaussiana:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{22}$$

donde x y y corresponden a las coordenadas de los píxeles y σ es un parámetro que define el ancho de la campana gaussiana, tal como se describe en la figura 2.4.

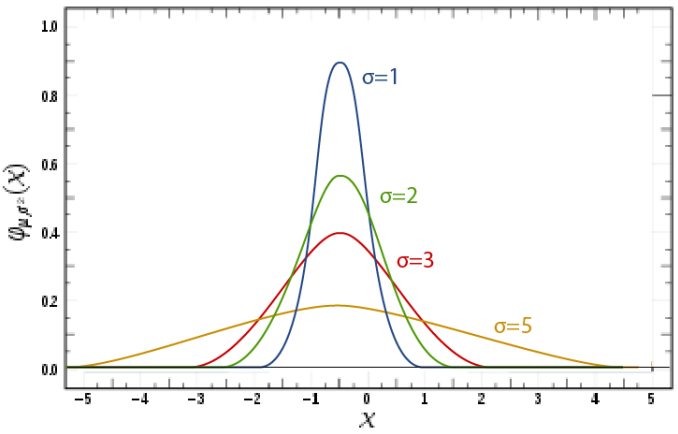


Figura 2.4: Efecto de variar el parámetro σ en la campana de Gauss.

2.2.7. Detección de discontinuidades en imágenes

La detección de discontinuidades en imágenes se logra mediante el uso de filtros [7] y busca detectar principalmente bordes (discontinuidades entre dos regiones homogéneas), líneas (discontinuidades bilaterales en un fondo homogéneo) y esquinas (píxel que tiene un borde en dos o más direcciones).

A lo largo del tiempo se han desarrollado algoritmos y métodos que permiten la detección de estas discontinuidades, algunos de los cuales se mencionan a continuación:

- **Algoritmo de Marr-Hildreth:** Este fue uno de los primeros algoritmos desarrollados, por lo que ante las constantes mejoras en el área de DIP ya se encuentra en desuso. Su problema principal fue la generación de falsos bordes en las imágenes y los errores de localización que generaba en los bordes curvos. Su base matemática consiste en la aplicación de filtros gaussianos a cada píxel individual, cuya amplitud para cada uno depende de la desviación estándar especificada (figura 2.4). Después de dicho suavizado mediante los filtros gaussianos, se aplica a la imagen el operador *laplaciano*, el cual distingue entre diferentes rangos de frecuencias para permitir la detección de los bordes según [12]:

$$\nabla^2 G(r) = -\frac{1}{\pi\sigma^4(1 - r^2/2\sigma^2)} e^{-r^2/2\sigma^2} \tag{23}$$

definido como el algoritmo de Marr-Hildreth, donde r indica el valor del píxel individual y σ el ancho de la campana gaussiana para el filtrado de la imagen.

- **Algoritmo de Canny:** Este algoritmo permite detectar bordes con una respuesta clara y localización precisa [6]. Un diagrama de flujo que representa el proceso del algoritmo de Canny se muestra en la figura 2.5.

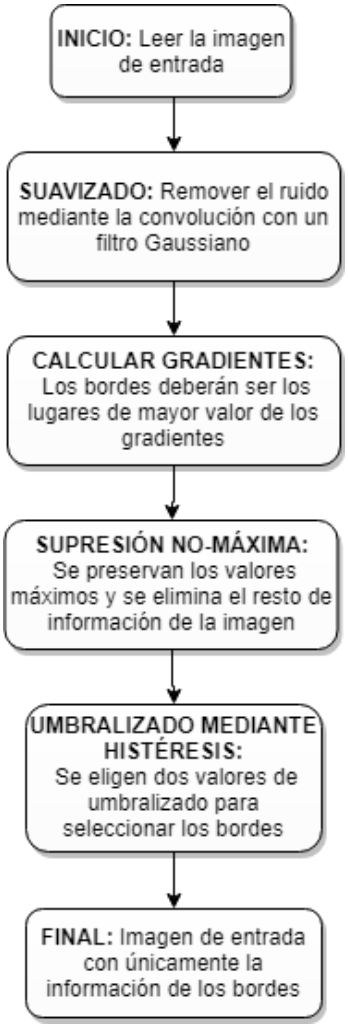


Figura 2.5: Diagrama de flujo para el algoritmo de detección de bordes de Canny. Adaptada de [13].

En el algoritmo de Canny se ejecuta como proceso inicial un suavizado para reducir los niveles de ruido en la imagen, mediante la convolución de la imagen de entrada $I(x, y)$ con un filtro gaussiano G (ecuación 22). La imagen resultante está dada por:

$$F(x, y) = G * I(x, y) \tag{24}$$

El *cálculo de los gradientes* de la imagen se realiza para encontrar los lugares en los que el cambio de intensidad de los niveles de gris es más pronunciado (máximo), y se realiza mediante el operador de Sobel, el cual se convoluciona con la imagen suavizada. El resultado de la operación son los gradientes en las direcciones x y y, tal que:

$$\begin{aligned} G_x &= D_x * F(x,y) \\ G_y &= D_y * F(x,y) \end{aligned} \tag{25}$$

La magnitud del gradiente (la intensidad del borde encontrado) de un píxel está dada por:

$$G = \sqrt{G_x^2 + G_y^2} \tag{26}$$

y su dirección por:

$$\theta = arctan\left(\frac{G_x}{G_y}\right) \tag{27}$$

La *supresión de no máximos* se realiza para conservar únicamente la información en la que los cambios en los niveles de intensidad es mayor, y eliminar así los bordes más delgados o contornos no significativos [13]. El gradiente de la imagen se rota $\theta = 45°$, y posteriormente se compara el gradiente de cada píxel para determinar si su valor es mayor o menor al de los píxeles vecinos. En caso de ser menor, el píxel es descartado como borde.

En el *umbralizado por histéresis* se establecen dos valores de umbral (un máximo y un mínimo) para determinar cuáles están realmente ligados a un borde. Si el valor del gradiente de un píxel es menor que el umbral mínimo establecido, entonces es descartado; si por el contrario el valor es mayor, se mantiene el píxel como un borde. En el caso en el que el valor del gradiente del píxel se encuentre entre los valores establecidos para los umbrales mínimo y máximo, se evalúa si en la vecindad 3×3 del píxel existen valores definidos como borde, en cuyo caso se establece también dicho píxel como píxel de borde.

- **Transformada de Hough:** Este algoritmo resulta útil cuando se conoce la forma geométrica *exacta* que se busca en una imagen. Utiliza el espacio paramétrico polar y es menos ruidoso que el algoritmo de Canny, sin embargo, la restricción en cuanto a la parametrización de las formas buscadas no lo hace funcional en muchos casos. Para una forma circular, la figura geométrica debe presentarse primero como:

$$(x-a)^2 + (y-b)^2 = r^2 \tag{28}$$

y después se calcula el número de circunferencias que podrían pasar por cada píxel de contorno, según:

$$cx = x_0 + cos\theta \cdot r$$
$$cy = y_0 + sin\theta \cdot r \tag{29}$$

Finalmente, se escogen como bordes los puntos de la imagen con mayor número de circunferencias resultantes del proceso anterior [14].

2.2.8. Fundamentos del color

Muchos algoritmos de DIP trabajan con imágenes en escala de grises debido a la facilidad computacional que esto ofrece con respecto al procesamiento de imágenes a color, pues la diferencia entre estos dos tipos de imagen radica en el *espacio de color* que lo domina [16]. El espectro continuo de colores está dividido en seis regiones (figura 2.6), cada una con diferentes longitudes de onda que van desde los 400 nm (violeta) hasta los 700 nm (rojo).

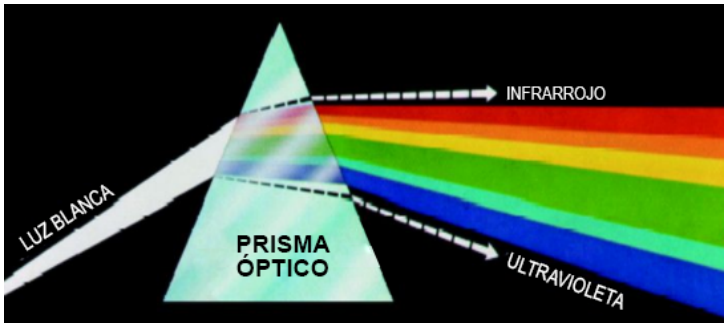


Figura 2.6: Espectro de color resultante al pasar luz blanca a través de un prisma . Adaptada de [6].

Los *colores primarios* son colores a partir de los cuales se puede obtener el resto de combinaciones cromáticas existentes, y en una imagen obtenida a partir de un sensor digital sus longitudes de onda son:

- Rojo 700 nm
- Verde 546.1 nm
- Azul 435.8 nm

El espacio de color al que normalmente pertenecen las imágenes visualizadas en una computadora, es el espacio **RGB** *(rojo-verde-azul)*, el cual depende del dispositivo que capture la imagen. Para conocer más acerca de conceptos y fundamentos de imágenes a color, el lector puede referirse a la referencia bibliográfica [6] de este documento.

2.3. Frameworks para procesamiento digital de imágenes

2.3.1. Python

Python es un lenguaje intérprete que permite una abstracción de alto nivel, por lo que su uso es sencillo y aplicado en múltiples áreas de desarrollo mediante la instalación de bibliotecas en su entorno. Una ventaja de este lenguaje es que no realiza cálculos innecesarios que podrían ocupar una gran cantidad de espacio en memoria y por ende ralentizar los procesos [17].

2.3.2. OpenCV

OpenCV (*Open Source Computer Vision Library*) es una biblioteca de código abierto y adaptable a múltiples lenguajes de programación, orientada al campo de la visión por computador. Se recomienda al lector referirse al documento [19], el cual contiene información sobre las principales funciones de OpenCV utilizadas en este estudio.

2.4. Sensado remoto

El sensado remoto es la tecnología que permite obtener información acerca de un objeto sin estar en contacto directo con este, y es posible afirmar que las primeras apariciones de esta técnica iniciaron con la toma de fotografías, en donde las cámaras actúan como sensores remotos que capturan información de la zona de enfoque o medio de destino [21]. Los tipos de sensores remotos clasificados como pasivos, aprovechan la energía natural que es reflejada a la superficie terrestre, por ejemplo, una cámara que captura fotografías a la luz del Sol.

2.4.1. Resolución espacial

El término *resolución espacial* generalmente se caracteriza mediante la (*Ground Sample Distance*) o también llamada *tamaño del píxel de las imágenes* [22]. La GSD es el tamaño mínimo que puede ser detectado de las imágenes en el suelo, aunque la efectividad en la interpretación de su cálculo depende de variables tales como el ruido, resolución del sensor, nitidez y contraste.

Uno de los estándares más comunes para interpretar y representar la resolución espacial es el esquema NIIRS (National Imagery Interpretability Rating Scale), el cual es utilizado por analistas de imágenes al asignar un número para calificar la interpretabilidad de una imagen dada [23].

Los conceptos de resolución espacial y esquema NIIRS proveen una aproximación sistemática para medir la calidad de las fotografías de imágenes digitales, el rendimiento de los dispositivos para capturar las imágenes, y los efectos de los algoritmos de procesamiento

digital de imágenes.

Los parámetros necesarios para realizar el cálculo de la GSD se muestran en la figura 2.7, y estos se utilizan para realizar la operación según se indica en la ecuación 30.

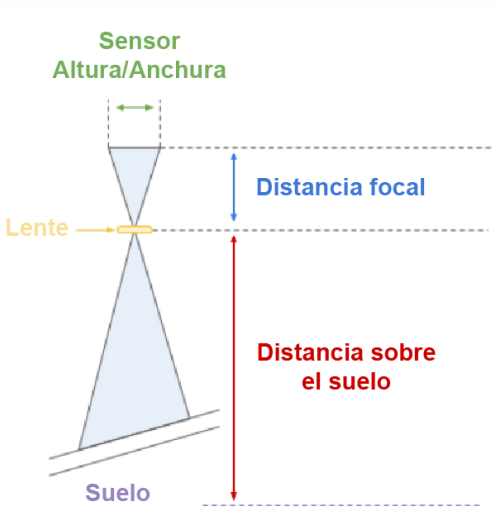


Figura 2.7: Parámetros para el cálculo de la GSD. Adaptado de [24]

$$GSD_h = \frac{Distancia * AlturaSensor}{DistanciaFocal * AlturaImagen}$$
$$GSD_w = \frac{Distancia * AnchuraSensor}{DistanciaFocal * AnchuraImagen} \quad (30)$$

Un píxel se puede *aproximar* como un cuadrado, sin embargo, al no serlo por completo, la ecuación 30 utiliza cálculos tanto para altura como para anchura, los cuales (probablemente) diferirán en algunas de sus cifras. Para la elección del resultado final se seleccionará el valor más grande, dado que este representaría el peor escenario, es decir, el caso en el que cada píxel representa una mayor distancia con respecto al suelo y, por ende, una menor resolución espacial.

Los valores de altura y anchura del sensor, distancia focal y altura y anchura de la imagen, dependen del sensor utilizado y deben estar especificados en la hoja de datos del fabricante. Estos cálculos indican únicamente una aproximación, dado que la calidad obtenida puede variar dependiendo de factores tales como el enfoque y el tipo de escenario en el que se encuentre.

2.5. Agricultura de precisión

La *agricultura de precisión* es un término que se utiliza para referirse a la aplicación de la tecnología en el sector agrícola, tanto en cuanto al uso de dispositivos electrónicos como al de productos y servicios. Su objetivo principal es aplicar la cantidad correcta de insumos en el momento y el lugar exactos, para optimizar el manejo de grandes terrenos mediante el uso de la tecnología y fomentando paralelamente el aumento tanto de la calidad como de la cantidad de productos obtenidos [25].

A continuación se mencionan algunos proyectos que involucran la agricultura de precisión relacionada con DIP en la detección de plagas, para mostrar el alcance que la combinación de software con sistemas electrónicos puede tener en el sector agrícola.

- **Detección de plagas en campos de arroz:** Utiliza técnicas de extracción e identificación de objetos mediante la comparación entre dos muestras de imágenes en escala de grises: una de estas completamente lisa y la otra con muestras de las plagas [27]. Si el píxel de la imagen analizada no supera el umbral mínimo de la imagen de referencia, el píxel se mantiene en la imagen resultante y las plagas son extraídas mediante un escaneo vertical y horizontal de la imagen. Estudios similares se han aplicado para monocultivos en distintas partes del mundo [28].
- **Trampa automática para la detección de polilla en el manejo integrado de plagas:** Busca la automatización de las trampas existentes para capturar polilla en los árboles de manzana y pera. El sistema toma fotografías de la trampa y las envía remotamente para su análisis por computador, de modo que se permite aproximar el ciclo de reproducción y los daños que provocará en la temporada. No trabaja con algoritmos DIP, sin embargo, su similitud con el presente estudio radica en la implementación de una trampa automática [29].
- **Clasificación de café verde:** Busca implementar un algoritmo de bajo costo mediante MATLAB y OpenCV para la detección de defectos en los granos de café, y utiliza técnicas de DIP como análisis de imágenes por color y extracción de fondo por contraste [30].
- **Detección de enfermedades en plantas:** Algunos estudios presentan métodos innovadores como el uso de los histogramas para detectar enfermedades como el *oídio* en las plantas [31], y técnicas de reconocimiento de patrones para detectar el tipo de insecto que provoca el daño [32].

Mediante la lectura de los estudios anteriores se comprueba que las técnicas de procesamiento de imágenes han contribuido grandemente en la mejora de los procesos relacionados con la agricultura, al utilizar algoritmos que logran optimizar el costo tanto computacional como económico en los cultivos [25]-[32].

3. Diseño e implementación del sistema de monitoreo de brocas

Al realizar un análisis en conjunto con el Icafé sobre las principales funciones con las que debería contar un sistema efectivo de monitoreo de brocas, resultó posible segmentar el problema en diversos componentes, cuyo resultado final busca un sistema de monitoreo efectivo en el campo, con un buen rendimiento según los requerimientos preestablecidos y con un bajo costo de producción (menos de $200, según criterio de expertos).

Las principales características con las que debe contar el sistema de monitoreo remoto de brocas en plantaciones de café se listan en la tabla 3.1.

Tabla 3.1: Requerimientos del sistema de monitoreo

ID	Requerimiento
SM-01	Capturar una imagen de las brocas atraídas y retenidas en la zona de muestreo de la trampa
SM-02	Informar sobre al menos el 80 % de la cantidad de brocas presentes al momento del monitoreo
SM-03	Utilizar un único sistema empotrado para realizar todos los procesos requeridos
SM-04	Almacenar y cargar la información en un servidor a través de Internet
SM-05	Sistema portable y energéticamente independiente

3.1. Análisis y selección final de la solución

A continuación se evalúan tres posibles soluciones con el objetivo de seleccionar el método más adecuado para desarrollar e implementar el sistema de monitoreo propuesto en este estudio.

- **Solución 1: Sistemas comerciales de monitoreo.**

  Los sistemas disponibles actualmente en el mercado tienen un principio de funcionamiento similar: se captura la muestra, se envía remotamente la información a través de conexiones inalámbricas, y se informa al usuario en tiempo real acerca del comportamiento de la plaga.

  El sistema *Spensa Sentinel* es un sistema automatizado que utiliza algoritmos de *deep learning* para monitorear insectos en tiempo real. El sistema es capaz de reconocer y descartar tipos de insectos que no son del interés final, además, cuenta con la capacidad de manualmente identificar a la plaga en caso de que el sistema pase por alto a algún insecto presente en la trampa.

  El sistema *Z-trap* trabaja mediante sensores de bioimpedancia para reconocer las especies que entran y generan discontinuidades en la corriente eléctrica de un sensor ubicado en el sistema. Estas trampas utilizan baterías recargables de larga duración (pueden

mantenerse durante todo un período de cosecha), y se personaliza según el tipo de cultivo y de plaga. El precio más económico para cualquiera de estos programas es de $45.306, con un mínimo recomendado de 30 trampas para 90 000 hectáreas de cultivo.

- **Solución 2: Sistema de monitoreo *full-custom*.**

  Es posible implementar un sistema que satisfaga por completo los requerimientos listados en la tabla 3.1 mediante el desarrollo de plataformas de hardware y software que se adapten completamente a las necesidades específicas. El sistema empotrado puede desarrollarse con una interfaz sencilla para el usuario, que permita programar los tiempos de monitoreo y reiniciar el sistema mediante botones.

- **Solución 3: Sistemas de monitoreo con plataformas *open hardware-software*.**

  El sistema que se propone en este estudio para el monitoreo de brocas en campos de café, debe cumplir esencialmente los requisitos **SM-01** al **SM-05** listados en la tabla 3.1. La limitación del sistema a identificar únicamente brocas permite el desarrollo de un sistema más sencillo que los mencionados anteriormente, debido a que las condiciones de monitoreo no varían significativamente y se puede trabajar con tamaños únicos y conocidos del insecto. Debido a que se busca realizar un sistema de bajo costo, se pueden utilizar herramientas de hardware y software libre, muchas de las cuales permiten realizar análisis estadísticos completos que brinden al usuario la información necesaria sobre el estado de sus cultivos en tiempo real. Además, al adaptar el sistema para el reconocimiento únicamente de brocas, es posible buscar los sensores adecuados para la captura que brinden las características suficientes por el menor costo posible.

La evaluación de las soluciones anteriores se muestra en la tabla 3.2, donde 1 corresponde a una característica *deficiente* y 5 a una *satisfactoria*.

Tabla 3.2: Comparación de las posibles soluciones para un sistema de monitoreo de brocas

Característica	**Solución 1**	**Solución 2**	**Solución 3**
Funcionalidad	5	5	5
Consumo energético	3	5	5
Costo económico	1	1	5
Tiempo de desarrollo	5	1	3
Complejidad del procesamiento	5	1	4
Total	19	13	22

La solución 1 cumple los requerimientos establecidos por el usuario para el sistema de monitoreo, mientras que las soluciones 2 y 3, al ser implementadas desde cero, se pueden construir de modo que también satisfagan dichos requerimientos, lo que hace a las 3 completamente funcionales. El consumo energético de los sistemas que integran la solución 1

depende del producto evaluado, por ejemplo, el sistema Spensa Sentinel se alimenta mediante baterías recargables a través de paneles solares, mientras que Z-trap trabaja con baterías de larga duración, lo cual podría no ser suficiente para períodos continuos de cosecha.

El costo de implementación de la solución 1 para un período de cosecha está muy por encima del requerimiento establecido por el usuario, y el costo económico de producir plataformas de hardware y software también colocan a la solución 2 dentro de esta característica. La solución 3, por el contrario, utiliza plataformas ya existentes que no requieren del pago de licencias, por lo que es muy probable lograr su producción por un costo menor a $200 (según el requerimiento inicial).

El tiempo de desarrollo del sistema es una variable compleja de medir, pues para enviar un producto al mercado se requieren de múltiples evaluaciones para asegurar la funcionalidad. Por otra parte, el desarrollo de plataformas de hardware es un proceso lento, ya que requiere de múltiples verificaciones antes de enviar a fabricar las placas y, una vez recibidas, el proceso de pruebas de funcionalidad. La solución 3 no requiere de procesos de fabricación, sin embargo, los sistemas existentes se deben adaptar a las necesidades finales del usuario.

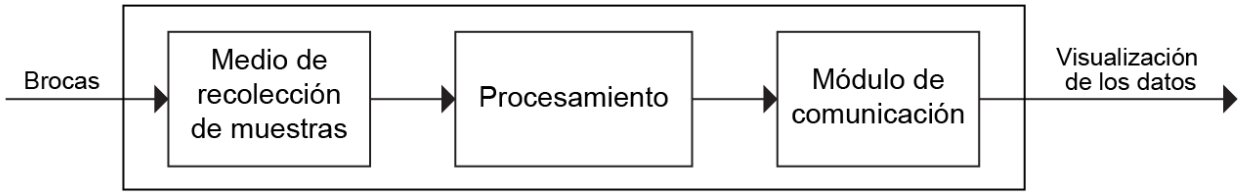
Debido a que la solución 2 busca implementar los algoritmos de procesamiento desde cero, la complejidad del procesamiento se eleva en consecuencia de los múltiples cálculos y pruebas necesarias. Por el contrario, al utilizar algoritmos de procesamiento de imágenes *open source* ya existentes, la complejidad en el desarrollo de un algoritmo que se ajuste a los requerimientos de la tabla 3.1 es menor.

La cuantificación de las características listadas en la tabla 3.2, permite concluir que la solución más viable para desarrollar el algoritmo propuesto en el presente estudio es la solución 3, es decir, utilizar plataformas de *open software-hardware*.

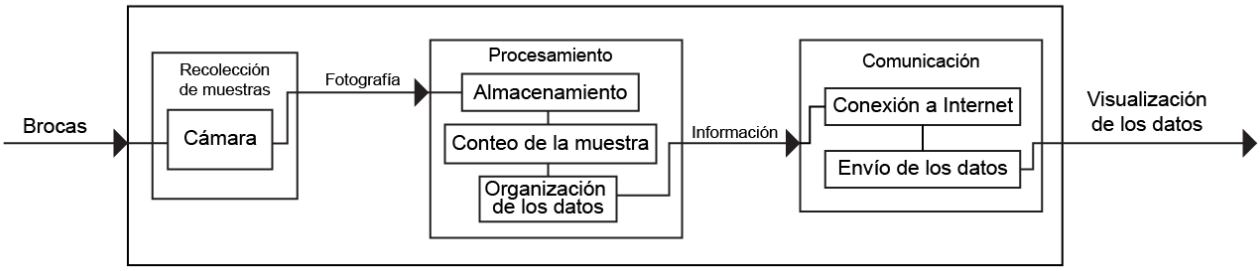
3.2. Desarrollo del concepto de diseño

En esta sección se explica con detalle el desarrollo del sistema de monitoreo de brocas, segmentando cada una de sus partes conforme se avanza en el trabajo para una mejor compresión del método por parte del lector.

Un diagrama que ilustra el funcionamiento principal del sistema de monitoreo propuesto, se muestra en la figura 3.1.



(a)



(b)

Figura 3.1: Diagrama de alto nivel del sistema de monitoreo: a) alto nivel, b) detalles.

A continuación se describe la función de cada módulo, así como las entradas y salidas que los conforman.

3.2.1. Módulo de recolección de muestras

Está compuesto por la trampa del sistema, en la cual se encuentra empotrada una cámara cuya área de enfoque consiste en el medio de retención de la plaga, y cuyo diseño se muestra en la figura 3.2. Las dimensiones se eligieron según el criterio de expertos del Icafé, quienes han utilizado trampas tipo botella para la recolección de la broca. Con base en el tamaño de estas trampas, se elaboró un diseño en material *acrílico transparente* para el sistema de monitoreo en cuestión y se realizaron pruebas en los campos de café, para poder determinar

si la trampa era lo suficientemente precisa como para permitir la recolección del insecto.

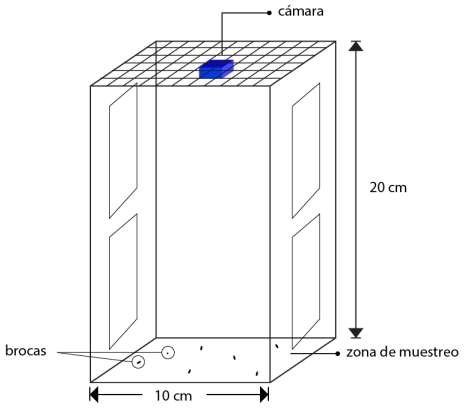


Figura 3.2: Simulación de la primera etapa del sistema de monitoreo

.

- Entrada al módulo: Brocas atraídas a la trampa.
- Salida del módulo: Fotografía de la zona de muestreo.

3.2.2. Módulo de procesamiento

Este módulo almacena cada fotografía obtenida del *módulo de recolección* dentro de un sistema empotrado, el cual será el núcleo de funcionamiento del sistema de monitoreo. Por cada fotografía tomada, se realiza el procesamiento de la imagen de forma que se contabilice la cantidad de brocas atrapadas en la zona de muestreo de la trampa. La segunda etapa del sistema asume que todos los insectos capturados en la primera etapa corresponden a brocas, para poder así aprovechar el alto contraste entre el color oscuro de las brocas y un papel adhesivo de color sólido y, a su vez, reducir la complejidad del algoritmo al descartar las dificultades que puede generar el proceso de reconocimiento y discriminación de especies en una misma zona de muestreo.

El módulo de procesamiento organiza los datos obtenidos por cada tiempo de monitoreo, lo cual incluye el nombramiento de las imágenes según la fecha y hora en la que fueron capturadas; un registro en un archivo de texto de la cantidad de brocas encontradas en cada fotografía, así como la fecha y hora de la toma de la muestra; y la preparación de esta misma información para ser visualizada gráficamente por el usuario a través de Internet.

- Entrada al módulo: Fotografía de la zona de muestreo.
- Salida del módulo: Información recopilada.

3.2.3. Módulo de comunicación

El componente principal del *módulo de comunicación* es la conexión a Internet, la cual permitirá la carga de las imágenes y registros a un servidor, así como la carga de la información estadística a una plataforma de visualización según el tiempo de monitoreo establecido. Este módulo finaliza las funciones del sistema de monitoreo, pues devuelve como salida la visualización de los datos al usuario, permitiéndole así trabajar y analizar la información a conveniencia.

- Entrada al módulo: Información recopilada.
- Salida del módulo: Visualización gráfica de los datos.

3.3. Selección del hardware y software para el sistema de monitoreo

En la subsección anterior se describió el proceso de funcionamiento del sistema de monitoreo, sin embargo, es necesario definir con exactitud cuáles herramientas de hardware y (o) software se encargarán de cumplir el objetivo especificado en cada módulo, a fin de realizar el sistema que mejor satisfaga los requerimientos listados en la tabla 3.1.

3.3.1. Sistema empotrado

Como punto de partida se tomaron en cuenta cuatro de los sistemas más utilizados en la actualidad para desarrollar aplicaciones electrónicas de bajo costo (menos de $100), según criterio de expertos: la plataforma electrónica de prototipos Arduino [39], y las plataformas de desarrollo Raspberry Pi [33], Orange Pi [40] y Beagle Bone [41], las cuales se muestran en la figura 3.3. La tabla 3.3 realiza un resumen de las principales características que se busca evaluar en el sistema empotrado que se seleccione para el sistema de monitoreo, y su descripción para cada una de las plataformas mencionadas anteriormente.

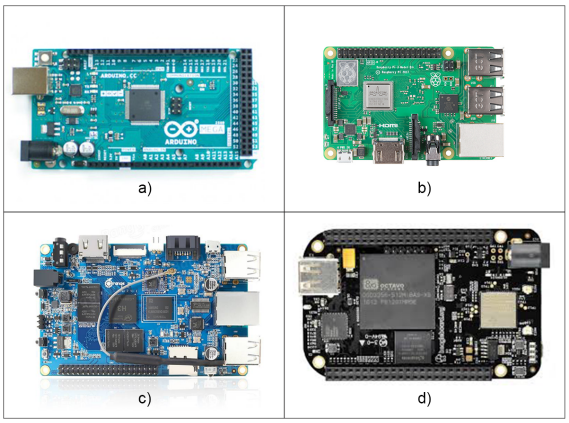


Figura 3.3: Placas de bajo costo a) Arduino Mega 2560 [39], b) Raspberry Pi 3 Model B+ [33], c) Orange Pi Plus2 [40] y d) Beaglebone Black Wireless [41]

.

Tabla 3.3: Comparación de características para la selección de hardware

Característica	**Arduino Mega 2560**	**Raspberry Pi 3 Model B+**	**Orange Pi Plus2**	**Beagle Bone Wireless**
Velocidad de procesamiento	16 MHz	1.2 GHz	1.6 GHz	1 GHz
SRAM	8 KB	1 GB	2 GB	512 MB
Sistema Operativo	Arduino, DuinOS	Raspbian, Fedora, Windows IoT Core	Ubuntu, Debian, Raspbian, Android	Linux, Windows, Android, Neutrino
Conexión de módulos externos	Sí	Sí	Sí	Sí
Interfaz de cámara	No	Sí	Sí	Sí
Módulo Wi-Fi integrado	No	Sí	Sí	Sí
Aplicaciones DIP realizadas	No	Sí	Sí	Sí
Disponibilidad en CR	No	Sí	No	No
Peso	37 g	45 g	83 g	41 g
Costo	$\sim$ \$40	$\sim$ \$40	$\sim$ \$60	$\sim$ \$90
Alimentación	$7 \sim 12$ V	5 V, 2.5 A	5 V, 2 A	5 V, 1 A

La placa Arduino fue descartada debido a que su velocidad de procesamiento es la más lenta entre los dispositivos analizados, y la cantidad de memoria del microprocesador no es suficiente para realizar múltiples operaciones sobre imágenes de manera rápida [42]. Las tarjetas de evaluación Raspberry Pi, Orange Pi y Beagle Bone poseen características similares que permiten el desarrollo de la programación requerida para el sistema de monitoreo en cualquiera de estas.

Tomando como referencia la información de la tabla 3.3, se decide utilizar como sistema principal para el procesamiento de la información y desarrollo de los algoritmos de captura, procesamiento de la imagen y envío remoto, un sistema empotrado tipo *Raspberry Pi 3 Model B+*. Sus características permiten la integración de interfaces de cámara, así como la instalación de sistemas operativos de distribuciones Linux, el cual es muy utilizado en el desarrollo de aplicaciones de procesamiento digital de imágenes [7].

3.3.2. Sensor de imagen

El tipo de sensor requerido para capturar los datos en el sistema de monitoreo propuesto, es una cámara que enfoque la zona de muestreo en la que se encuentran atrapadas las brocas. Debido a que el sistema empotrado escogido es una Raspberry Pi, la primera cámara evaluada fue el dispositivo de Raspberry *Pi NoIR Camera v2*, el cual se muestra en la figura 3.4, y

cuyos drivers son 100 % compatibles y físicamente adaptable al modelo Raspberry Pi 3 B+.

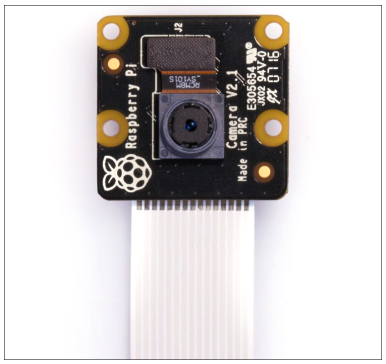


Figura 3.4: Pi NoIR Camera v2 [43]

Debido a que el tamaño máximo que puede alcanzar la broca es de 1.5 mm, se requiere de una cámara cuya resolución espacial teórica permita obtener detalles de objetos ubicados a una distancia de 0.20 m del sensor (figura 3.2). Para obtener dicha información se evalúan las características principales del dispositivo según la hoja de datos del fabricante, las cuales se listan en la tabla 3.4 junto a otras características de importancia.

Tabla 3.4: Características de la cámara NoIR v2

Característica	**Camera Module v2**
Costo	$25
Tamaño	$25 \times 24 \times 9$ mm
Peso	3 g
Resolución estática	8 Mpx
Resolución del sensor	3280×2464 px
Área de imagen del sensor	$3{,}68 \times 2{,}76$ mm
Distancia Focal	3.04 mm

Se utiliza la ecuación 30 para determinar la GSD que se puede obtener mediante este sensor a una altura de 20 cm (simulación de la primera etapa), de modo que para unidades en centímetros (cm) se tiene:

$$\begin{aligned} GSD_h &= \frac{20*0{,}276}{0{,}304*2464} \\ GSD_w &= \frac{20*0{,}3687}{0{,}304*3280} \end{aligned} \tag{31}$$

para obtener

$$GSD_h = 73,6928\mu m/pix$$
$$GSD_w = 73,8125\mu m/pix \tag{32}$$

Al escoger el peor de los casos entre los resultados anteriores, se concluye que

$$GSD = 73,8125\mu m/pix \tag{33}$$

El resultado anterior permite concluir que es posible obtener detalles para objetos de 1.5 mm atrapados en el papel adhesivo con facilidad. Sin embargo, para comprobar esta conclusión, se realiza el procedimiento inverso al colocar la altura máxima a la que debe colocarse el sensor para obtener una resolución espacial de justamente 1.5 mm/pix.

Al despejar la altura de colocación del sensor de la ecuación 30, se obtiene

$$Altura = (DistanciaFocal * AnchuraImagen)\frac{GSD}{AnchuraSensor} \tag{34}$$

La ecuación 34 utiliza los valores de anchura del sensor y anchura de la imagen, debido a que el caso escogido para el valor final de la GSD corresponde a los valores para dicha característica, sin embargo, para este caso se sustituirá la GSD por los 1.5 mm/pix requeridos. Colocando los valores en la ecuación 34 se obtiene:

$$Altura = (0,304 * 3280)\frac{1{,}5m}{0{,}368} \tag{35}$$

Con lo que se obtiene

$$Altura = 4,06m \tag{36}$$

Del resultado anterior se puede concluir que si el sensor escogido se coloca a una distancia aproximada de 4 metros de la zona de muestreo, aún sería posible obtener imágenes con una resolución que permita distinguir los *puntos broca* en el papel adhesivo mediante software. Sin embargo, colocar la cámara a esa altura no sería viable desde el punto de vista de construcción, pues se requeriría de una mayor cantidad de material para la elaboración de la trampa, lo cual aumentaría los costos de producción. Además, debido a su tamaño se complicaría el

transporte y la colocación entre las plantas de café, y mediante las pruebas realizadas para la construcción de la trampa, se concluyó que las dimensiones propuestas en la figura 3.2 son adecuadas para la aplicación.

A pesar de que este dispositivo es el único que se sometió a evaluaciones, se concluyó que es una buena opción para utilizar en el sistema de monitoreo ya que su peso lo hace un dispositivo ligero, y su bajo costo permite aún al sistema mantenerse en un rango de precio menor a los $200 establecidos en los requerimientos de la tabla 3.1.

3.3.3. Módulo de comunicación

La Raspberry Pi 3 Model B+ cuenta con un módulo Wi-Fi integrado y con un puerto RJ45 para lograr la conexión a Internet, además de pines GPIO que se pueden utilizar para conectar módulos que funcionen con red celular. Sin embargo, las conexiones alámbricas disminuirían la portabilidad del sistema, y para utilizar la conexión de módulos externos se debe tomar en cuenta la cantidad de corriente consumida por el dispositivo utilizado, los cuales generalmente requieren de una fuente de energía adicional a la proporcionada por la placa [33].

Actualmente, la tecnología ofrece dispositivos que pueden funcionar como punto de acceso a través de la red celular, por lo que mediante un módem con dicha característica o un *datacard* configurado en un lugar cercano, se podría generar un punto de red Wi-Fi y conectar a este el sistema empotrado, para aprovechar así su característica de Wi-Fi integrado y poder acceder a Internet desde prácticamente cualquier lugar del país. Las bandas de frecuencia en las que trabajan las principales redes telefónicas en Costa Rica se muestran en la tabla 3.5 [36].

Tabla 3.5: Bandas en las que operan las distintas compañías de telefonía en Costa Rica

	2G (MHz)	**3G (MHz)**	**4G LTE (MHz)**
Kölbi	850, 1800	1800	2600
Claro	1800	1800, 2100	1800
Movistar	850, 1800	850, 1800, 2100	1800

El tipo de conexión de la red celular dependerá del lugar en el que se coloque el sistema, de la estabilidad del proveedor de la red, y de la calidad del dispositivo utilizado como módem, sin embargo, se escoge el uso de la conexión mediante Wi-Fi como la solución más viable debido a que se puede establecer fácilmente si se aprovechan las tecnologías existentes.

3.3.4. Selección del framework de software

El lenguaje de programación debe permitir realizar de manera eficiente todas las funciones que controlan la toma temporizada de las fotografías, el procesamiento de las imágenes, la

organización de la información, y el envío de datos a través de Internet. La tabla 3.6 lista una serie de características que se buscan en el lenguaje de programación que se utilice, para respaldar la elección del software utilizado en este estudio.

Tabla 3.6: Características requeridas en el lenguaje de programación

ID	Característica	Descripción
LP-01	Alto nivel	Permite abstraer varios aspectos que simplifican la programación
LP-02	Multiparadigma	Permite varios estilos de programación
LP-03	Utilizado en DIP	Se han desarrollado aplicaciones en el área con el software
LP-04	Sencillo de adquirir	Software libre, con el objetivo de reducir costos
LP-05	Cuenta con bibliotecas	Permite instalar bibliotecas en su entorno
LP-06	Buen manejo de memoria	Permite un manejo eficiente de los recursos
LP-07	Elementos de lenguaje	Realizar comentarios, definir variables, ciclos, funciones, etc.
LP-08	Simplicidad	Aumenta la legibilidad y facilidad de escritura
LP-09	Multipropósito	No fue desarrollado para un área específica

Utilizar un lenguaje de programación de alto nivel, simple, y que permita la integración de elementos de lenguaje, disminuye la complejidad en el desarrollo de los procesos, y supone una ventaja para el equipo de trabajo que se involucre en el proceso de mejora y atención de las recomendaciones del presente proyecto. Es importante que al lenguaje de programación elegido se le puedan incluir bibliotecas para descartar errores en el manejo de los módulos externos y en el algoritmo desarrollado para procesar las imágenes.

Debido a que el sistema de monitoreo está previsto para trabajar continuamente durante largos períodos de tiempo, se tomará una cantidad de fotografías a elección del usuario final que estarán en constante tratamiento para la extracción de información, por lo que es prescindible un manejo óptimo de la memoria. A su vez, el sistema deberá evaluar constantemente si el tiempo de monitoreo ya ha sido cumplido, por lo que los recursos deben organizarse de una manera eficaz para no crear saturaciones que ralenticen el sistema y puedan generar atascos en los procedimientos, y con ello errores en el monitoreo.

Tras definir las características más importantes con las que debe contar el lenguaje de programación, se decidió utilizar *Python* en este proyecto, pues cumple en su totalidad con los requerimientos **LP-01** al **LP-09** listados en la tabla 3.6. También se analizaron y descartaron otros lenguajes como Matlab (a pesar de ofrecer una gran cantidad de funciones, su libre uso requiere de compra de licencias) y R (que a pesar de ser sencillo y similar a Matlab, fue creado con propósitos de análisis estadístico).

3.3.5. Bibliotecas para procesamiento de imágenes

El uso de Python se ha extendido al área de procesamiento digital de imágenes a través del uso de la biblioteca *OpenCV* [18], lo cual se puede comprobar mediante la lectura de investigaciones en aplicaciones similares ([26]-[27]). Tras estudiar las características y funciones principales de esta biblioteca [19], se decidió desarrollar el algoritmo del sistema de monitoreo a través de sus funciones.

3.3.6. Servidor web

Las fotografías obtenidas son almacenadas en un servidor web para tener una mejor administración de los datos capturados por el sistema. Un resumen de las características requeridas se muestra en la tabla 3.7.

Tabla 3.7: Características requeridas para la elección del servidor web

ID	Característica
WEB-01	Creación de carpetas para organizar la información
WEB-02	Espacio mínimo de 365 MB de almacenamiento
WEB-03	Bajo costo o gratuito
WEB-04	Enlace con Linux
WEB-05	Enlace con Python

Debido a que se espera que el sistema remoto capture información de todo un período de cosecha, se requiere de un mínimo de 365 imágenes para el almacenamiento durante un año al monitorear la trampa cada 24 horas, razón por la cual se buscará obtener imágenes con un tamaño máximo de 1 MB y un servidor que tenga espacio de al menos 365 MB para cubrir todo el período de monitoreo. Dado que se busca que el sistema sea óptimo también en cuanto a costo de producción, la mejor alternativa es encontrar un servidor gratuito o del menor costo posible, y de fácil acceso para todos los usuarios.

El servidor web debe poder enlazarse a Linux desde el lenguaje de programación para poder realizar la carga de las imágenes al servidor desde un único programa principal, característica que se tomó como el punto de partida para la evaluación de servidores debido a que constituye parte fundamental del funcionamiento.

El primer gestor de datos analizado fue Dropbox, para el cual se encontró una aplicación web desarrollada en C++ llamada *Dropbox Uploader* [44], que permite enlazar al servidor para carga y descarga de archivos con sistemas operativos de la distribución Linux. Dropbox satisface de igual manera las características **WEB-01** a **WEB-03** y puede ser utilizado de forma gratuita para un tamaño máximo de almacenamiento de 2 GB [45], lo cual está muy por encima del límite mínimo de 365 MB del requerimiento **WEB-02**.

Por las razones mencionadas anteriormente y dado que se cumplen satisfactoriamente los requisitos de la tabla 3.7, se decide utilizar Dropbox como servidor web para el almacenamiento de datos del presente proyecto.

3.3.7. Visualización de las estadísticas

La información obtenida desde el sistema remoto debe ser mostrada al usuario a través de un medio interactivo el que resulte fácil comprender, analizar y visualizar los datos obtenidos. Esta descripción coincide con la definición de plataforma *IoT*, la cual es una interfaz creada entre los sensores remotos de los dispositivos y el almacenamiento de datos mediante una red.

Actualmente, existen múltiples plataformas *IoT* con características innumerables, desarrolladas por reconocidas empresas como Microsoft (Microsoft Azure IoT), Amazon (Amazon Web Services IoT) y Hewlett-Packard Enterprise (Universal of Things Platform) [46]. Estos sistemas, sin embargo, están diseñados para grandes aplicaciones que requieren la visualización e interacción con grandes cantidades de datos. Además, el uso de algunos de estos está limitado a la compra de sus respectivas licencias, por lo que aumentaría el costo del sistema desarrollado en este estudio.

Alternativamente, existen plataformas de uso libre que pueden ser utilizadas gratuitamente en línea. Una de ellas es ThingSpeak, una plataforma *IoT* abierta que permite el uso de Matlab Analytics y el almacenamiento de estadísticas *on-line* [47]. ThingSpeak permite la visualización de la información de entrada mediante gráficos, así como de los datos del momento de captura de la información. Además, permite obtener archivos descargables de los registros de la base de datos, con el objetivo de que el usuario tenga un mayor control sobre estos. Dado que con estas características se cumple el propósito de información al usuario de la etapa final del sistema de monitoreo, se decide utilizar ThingSpeak como la plataforma *IoT* de este estudio.

3.4. Desarrollo del Software de Control

Para facilitar el proceso de desarrollo del algoritmo principal del sistema de monitoreo, se utiliza una metodología *top-down* para validar todos los procesos necesarios con un mejor control de los procedimientos. Una primera formulación del sistema se muestra en el diagrama de flujo de la figura 3.5.

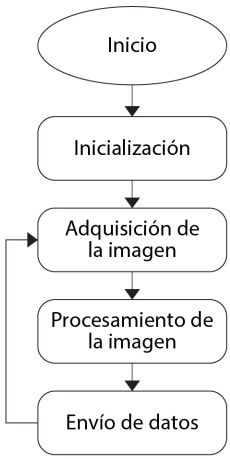


Figura 3.5: Diagrama de flujo principal del algoritmo de control del sistema de monitoreo

El módulo de *inicialización* detallado en la figura 3.6, realiza la carga de las bibliotecas necesarias para el funcionamiento adecuado del programa, como la biblioteca *PiCamera* para el funcionamiento adecuado de la cámara NoIR v2 a través de Python; la biblioteca *time* para obtener el tiempo del sistema y evaluar el cumplimiento de los tiempos de monitoreo; la biblioteca *subprocess* para realizar llamadas al sistema desde Python; *cv2* para obtener acceso a las funciones de OpenCV; y *urllib2*, *os* y *psutil* para la carga de datos a ThingSpeak. Posteriormente se definen e inicializan las variables necesarias, se crea el archivo de texto para el registro, y se definen los intervalos de monitoreo especificando los dígitos de horas, minutos y segundos.

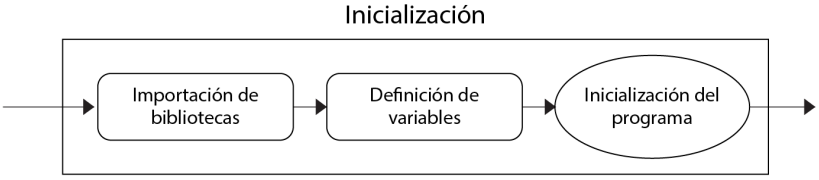


Figura 3.6: División del módulo de inicialización

En el módulo de *adquisición de datos* se evalúa si ya se cumplió el tiempo de monitoreo. En caso de no haberse cumplido, el programa espera en el módulo de *inicializaicón* hasta obtener una respuesta afirmativa. Cuando el tiempo se cumple, se captura la imagen y se

almacena en el sistema empotrado, como se ilustra en la figura 3.7.

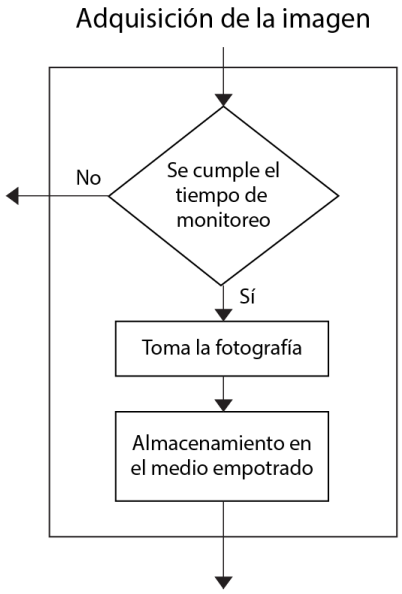


Figura 3.7: División del módulo de adquisición de la imagen

El módulo *procesamiento de la imagen* se divide según se muestra en la figura 3.8, en donde se selecciona la región de interés de la imagen (ROI) mediante el *zoom* que permite el sensor de imagen y a través de las funciones de OpenCV. Los demás procesos del módulo se realizan con esta nueva imagen modificada, a la cual se realiza la conversión a escala de grises para trabajar únicamente sobre un canal de imagen, y posteriormente se ecualiza para mejorar sus niveles de contraste.

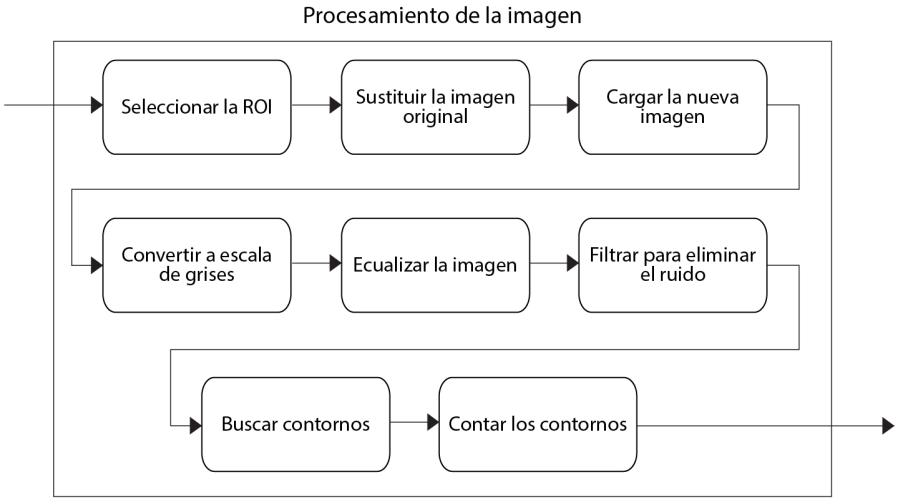


Figura 3.8: División del módulo de procesamiento de la imagen

El algoritmo realiza un filtrado cuyos parámetros dependen específicamente de la imagen utilizada, por lo que se deben realizar pruebas de Verificación para determinar el valor más conveniente. Para buscar los contornos en la imagen, se decidió utilizar el detector de bordes de Canny (figura 2.5) (el algoritmo de Marr-Hildreth fue descartado por ser un algoritmo lento e ineficiente, mientras que la aplicación de la transformada de Hough se descartó porque no es posible obtener una parametrización de la forma en la que la broca quedará atrapada en el sistema), y la cantidad de brocas se obtiene realizando el conteo del largo de la matriz de contornos externos obtenida tras aplicar el algoritmo.

En el módulo de *envío de datos* se escribe la información obtenida en el archivo de texto de respaldo, y carga las imágenes según son capturadas a Dropbox mediante una llamada al sistema utilizando la función **call** de la biblioteca *subprocess*, según el formato:

```
Upload = ''dropbox_uploader.sh upload ruta_archivos nombre_archivo''
call([Upload], shell=True)
```

para después escribir la cantidad de brocas halladas por cada tiempo de monitoreo en ThingSpeak. Al finalizar este proceso el programa regresa al módulo de *inicialización* para reanudar el flujo, como se representa mediante la figura 3.9.

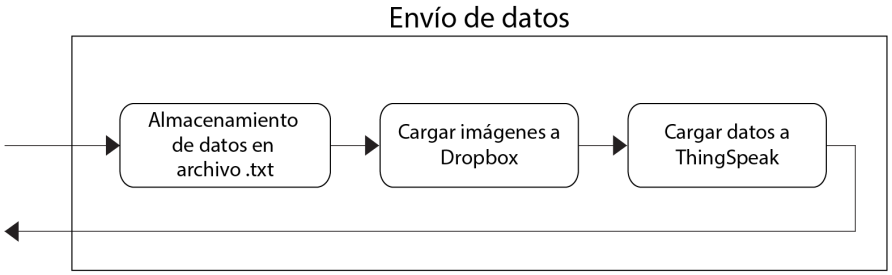


Figura 3.9: División del módulo de envío de datos

4. Verificación y análisis del sistema de monitoreo

En este capítulo se presentan y analizan los resultados obtenidos durante las pruebas de funcionamiento del sistema de monitoreo, mediante resultados como la calidad de las fotografías obtenidas, el alcance del procesamiento de las imágenes, la efectividad en la organización de los datos, así como las pruebas de velocidad para diferentes tipos de conexiones a Internet y el funcionamiento del sistema durante un día completo de monitoreo.

La figura 4.1 muestra el prototipo del sistema de monitoreo utilizado para realizar la verificación del sistema.

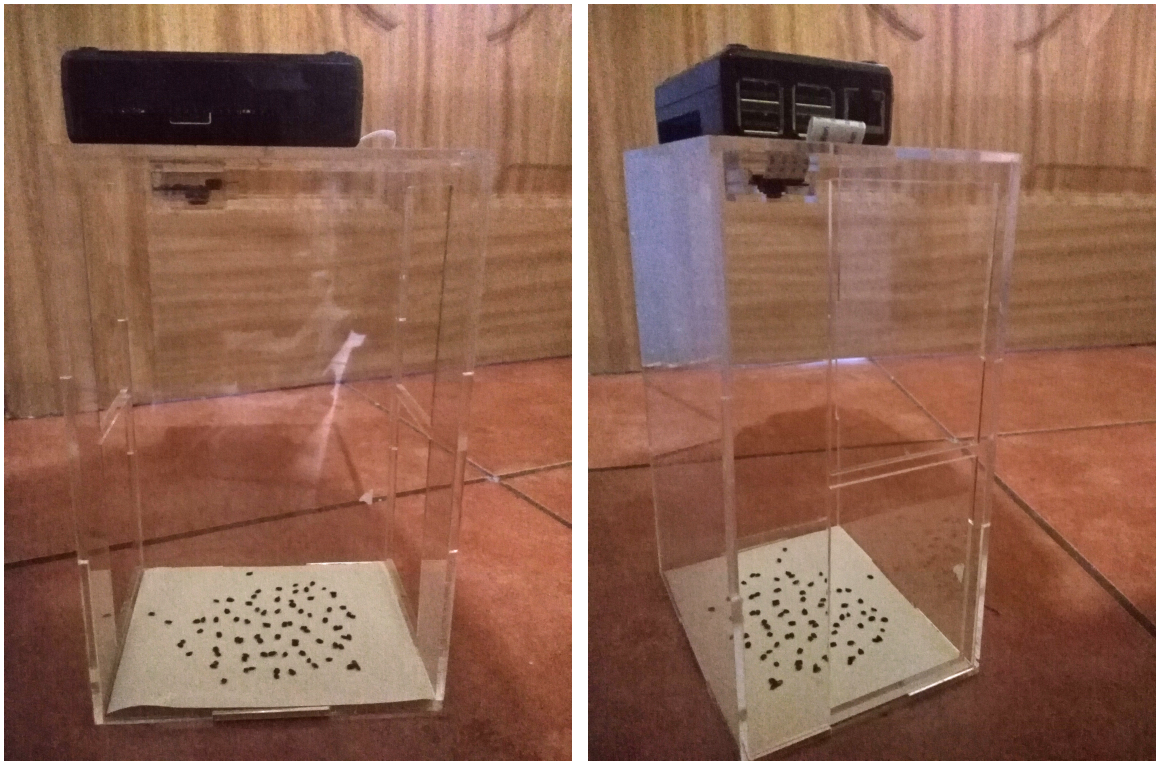


Figura 4.1: Prototipo del sistema de monitoreo diseñado.

4.1. Evaluación de las fotografías obtenidas

Como prueba inicial se capturó una fotografía para evaluar la calidad aportada por el sensor escogido. Cabe mencionar que en el medio de simulación se utilizaron semillas de chan para simular las brocas en la zona de muestreo, esto debido a su oscuro color y reducido tamaño. La fotografía obtenida con las configuraciones iniciales de la cámara *NoIR v2*, se muestra en la figura 4.2.

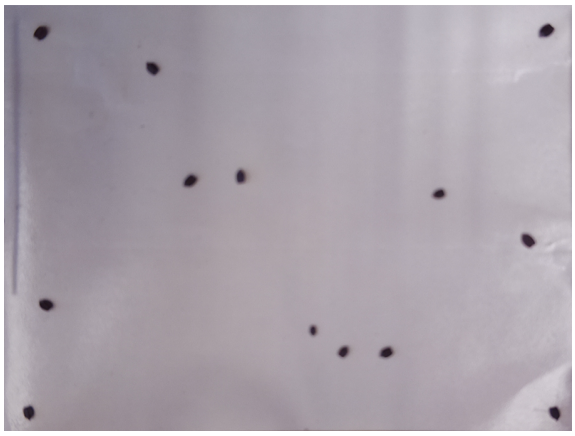


Figura 4.2: Primera captura de la zona de muestreo

A pesar de la reducida cantidad de detalles con la que cuenta la fotografía, es posible apreciar que la imagen se encuentra desenfocada. Debido a la poca manipulación física que se puede lograr con el sensor de imagen escogido, no es posible realizar un cálculo de la posición y (o) distancia óptima para tener un buen enfoque de la zona de muestreo, por lo que el proceso de mejora se debe realizar cualitativamente. La hoja de datos del sensor [43] sugiere una lista de funciones que pueden utilizarse para mejorar el aspecto de la imagen, tales como la modificación de la cantidad de brillo, contraste, el modo de exposición y la eliminación de ruido de la imagen. Estos parámetros se modificaron desde la vista previa de la cámara en tiempo real, sin embargo, no se logró mejorar el aspecto del enfoque obtenido en la figura 4.2, esto debido a la cercanía entre el sensor y la zona de muestreo (20 cm).

Tras evaluar las características físicas de la cámara con el objetivo de hallar una manera de mejorar el enfoque a través de hardware, se encontró que es posible modificar la posición angular del lente en el módulo de la figura 3.4, de modo que se pueda obtener una mejor calidad de imagen para objetos más cercanos al sensor. Sin embargo, esta práctica no se puede realizar de una manera precisa debido a la falta de indicadores para la cantidad de grados de giro a una distancia específica en la hoja de datos del fabricante, y tampoco se puede saber con exactitud cuántos grados se está girando el lente a falta de una escala impresa en la placa. Por esa razón, es necesario realizar pruebas girando el lente de forma aleatoria, hasta encontrar un enfoque satisfactorio. Al girar el lente del módulo en el sentido contrario a las manecillas del reloj, se obtiene una fotografía con el enfoque mostrado en la figura 4.3.

Figura 4.3: Segunda captura de la zona de muestreo: modificación de la posición del lente en sentido contrario a las manecillas del reloj.

Al comparar la primera imagen obtenida con la fotografía mostrada en la figura 4.3, resulta evidente que el desenfoque en esta última es mayor. Por esa razón, se decide girar el lente en el sentido de las manecillas del reloj, pues se espera el resultado opuesto y lograr obtener una mejor calidad de imagen. El resultado de esta prueba se muestra en la figura 4.4.

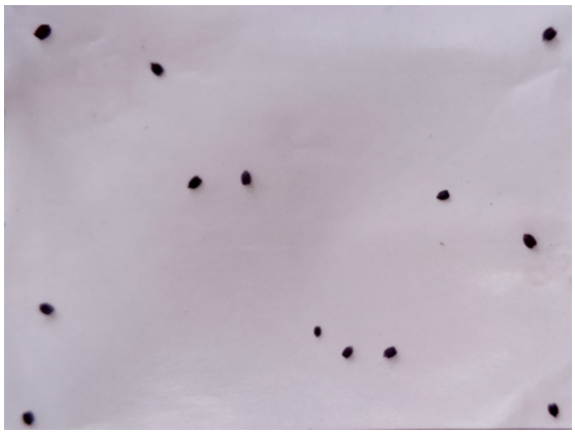


Figura 4.4: Tercera captura de la zona de muestreo: modificación de la posición del lente en sentido CW.

Mediante el giro en el sentido de las manecillas del reloj se logró un mejor enfoque, lo cual se puede determinar a través de la visualización de la fotografía. A pesar de que aún se podría mejorar o modificar la posición del lente para buscar mejores resultados, se decidió no seguir adelante con tales pruebas debido a que no se cuenta con las herramientas correctas para realizar el giro del lente (como el ajustador de lentes mostrado en la figura 4.5), por lo

que existe un riesgo de dañar el hardware.

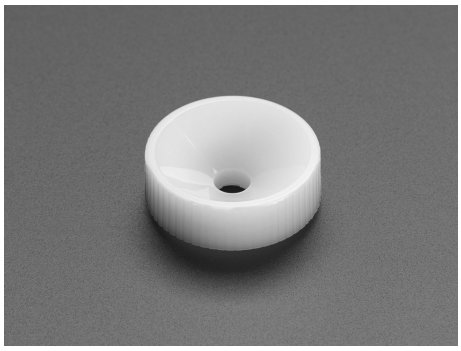
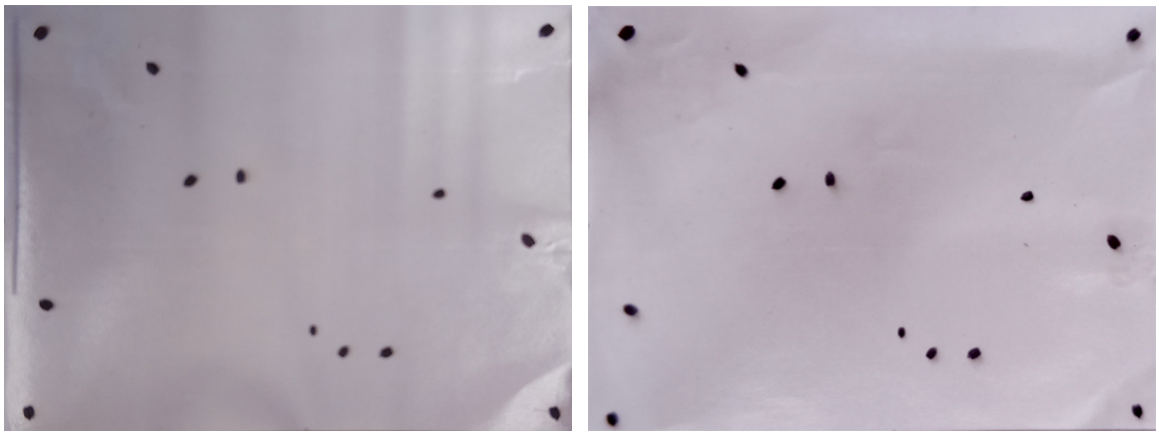


Figura 4.5: Herramienta para modificar la posición de un lente en los módulos de cámara [48].

La figura 4.6 muestra la mejora visual que se logró en el enfoque de la fotografía a través del giro del lente, por lo que se decidió mantener dicha configuración para la toma de fotografías en las pruebas faltantes del sistema.



(a) (b)

Figura 4.6: Mejora entre muestras de fotografías tras el giro manual del lente de la cámara. a) Imagen original, b) imagen con modificación de la posición del lente.

Las pruebas de enfoque de la cámara permitieron determinar también que la selección de la ROI propuesta inicialmente fue adecuada, debido a que se cubre por completo y únicamente la zona de muestreo de la trampa.

4.2. Procesamiento de la imagen

Para evaluar el funcionamiento del algoritmo de procesamiento de la imagen para el conteo de brocas, se utiliza la imagen mostrada en la figura 4.7, la cual cuenta con una cantidad

escogida aleatoriamente de 10 elementos.

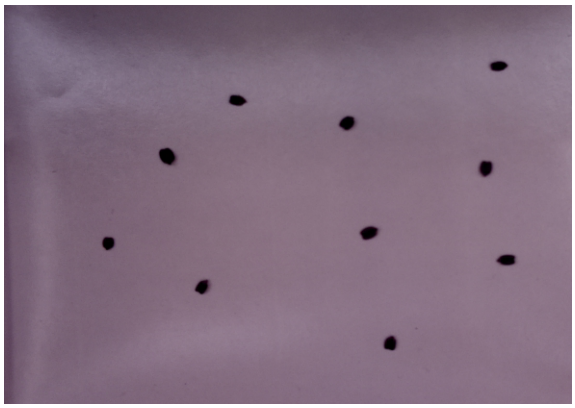
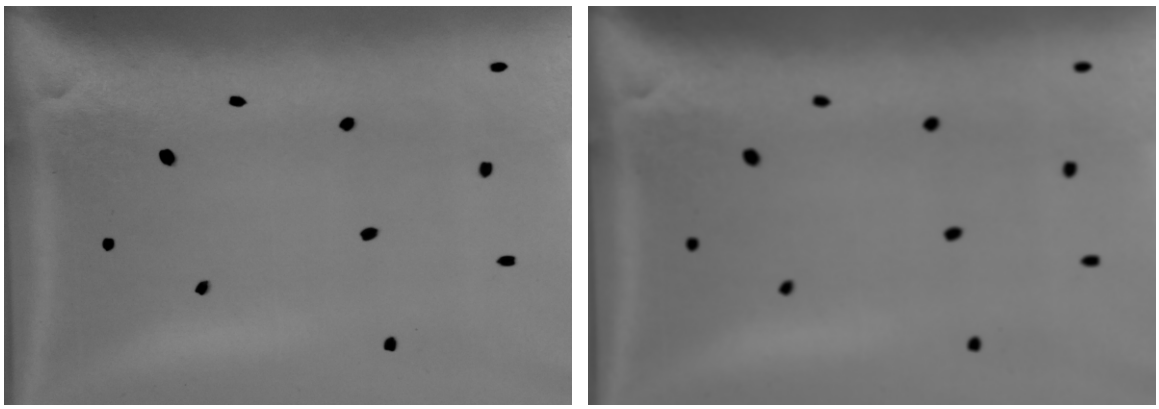


Figura 4.7: Imagen de prueba para la evaluación del algoritmo de conteo de brocas.

La imagen resultante tras la conversión a escala de grises se muestra en la figura 4.8a, y la salida de la etapa de filtrado se muestra en la figura 4.8b.



(a) (b)

Figura 4.8: Imagen de prueba para la evaluación del algoritmo de conteo de brocas. a) Escala de grises, b) filtro gaussiano.

Aplicar el filtro gaussiano a la figura 4.8a suaviza el granulado que estaba presente en la fotografía (debido en su mayoría a la textura del papel adhesivo), como se visualiza en la figura 4.9 al comparar los histogramas para ambas imágenes. Los cambios en los histogramas son muy sutiles debido al poco detalle que contiene la imagen; sin embargo, se observa que para la imagen filtrada los trazos tienen una mayor cantidad de pendientes cercanas a 0, lo cual indica que hay menos cambios de intensidad de gris, es decir, un mayor suavizado y por ende una disminución de los detalles no deseados.

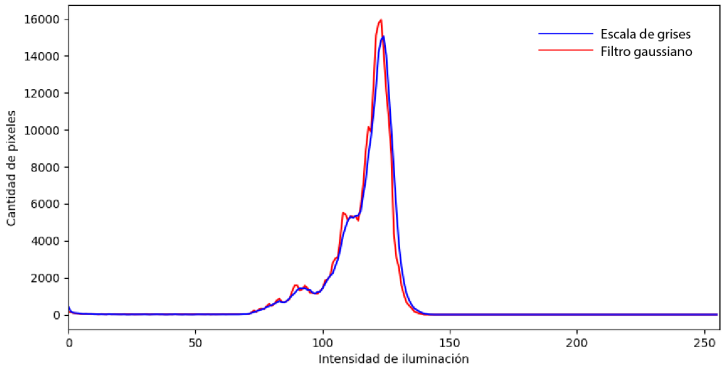


Figura 4.9: Imagen de prueba para la evaluación del algoritmo de conteo de brocas.

Para escoger el tamaño adecuado del kernel se utilizaron 4 valores distintos de máscaras, correspondientes a los más comunes utilizados en la práctica [7]. Los histogramas de las imágenes tras aplicar el filtro con distintas máscaras se muestra en la figura 4.10 .

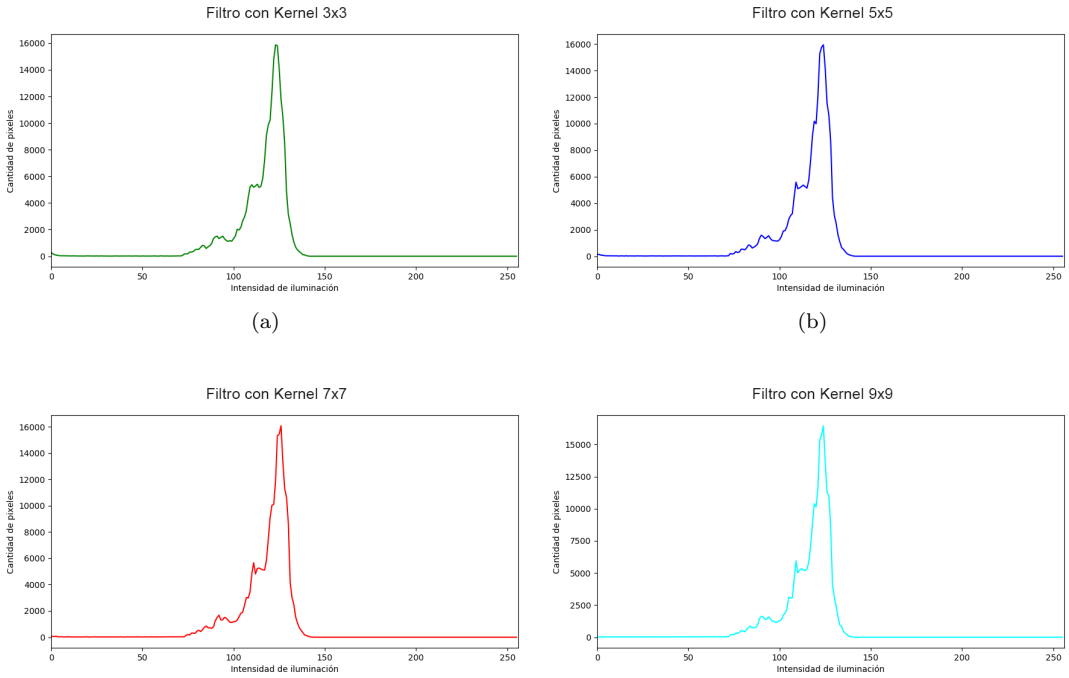


Figura 4.10: Histogramas para imagen filtrada con distintos tamaños de máscara.

Del procesamiento de las imágenes obtenidas del sistema de monitoreo, se espera que el filtro aplicado elimine detalles como las texturas generadas por el papel adhesivo y el efecto de las sombras en la fotografía. Al analizar los histogramas de las figuras 4.10a - 4.10d se observa que los cambios más significativos se dan alrededor de 110 en intensidad de iluminación, donde al aumentar el tamaño del kernel los cambios resultan más abruptos. Estos

cambios representan un mayor suavizado en los detalles de la imagen, lo cual puede ocasionar la pérdida de los contornos que se requieren contar para el desarrollo del algoritmo. Por esa razón, se decidió utilizar una máscara de tamaño 3×3 en el proceso de filtrado, ya que representa una mayor uniformidad en la distribución de píxeles en las zonas de cambio de intensidad, y la pendiente cercana a cero en dichas zonas indica el suavizado aplicado que permite eliminar detalles no deseados como se mostró mediante la figura 4.9.

El siguiente parámetro que puede especificarse al aplicar el filtrado es la desviación estándar. La documentación de OpenCV [49] provee la ecuación 37 para calcular la desviación estándar en términos generales al basarse en un tamaño de kernel específico, aunque este puede ser modificado a elección del usuario.

$$\sigma = 0{,}3 * ((ksize - 1) * 0{,}5 - 1) + 0{,}8 \tag{37}$$

Cabe destacar que OpenCV utiliza esta ecuación para satisfacer la condición de que el área bajo la curva de la campana gaussiana sea igual a 1, es decir

$$\sum G_i = 1 \tag{38}$$

donde G_i representa a la función Gaussiana descrita por la ecuación 2.4.

Debido a que el valor encontrado es cercano a 1, el ancho de la Campana Gaussiana no es tan pronunciado, lo cual justifica su aplicación sobre pequeñas vecindades de píxeles (figuras 2.4 y 2.2).

El siguiente paso en el procesamiento de la imagen es la ecualización, la cual es analizada frecuentemente con los histogramas de la imagen. La figura 4.11 muestra el histograma de la imagen original convertida a escala de grises (figura 4.8a), para la cual la región con mayor cantidad de píxeles se ubica en el término medio de la escala, es decir, no hay un exceso de

luces o de sombras en la fotografía.

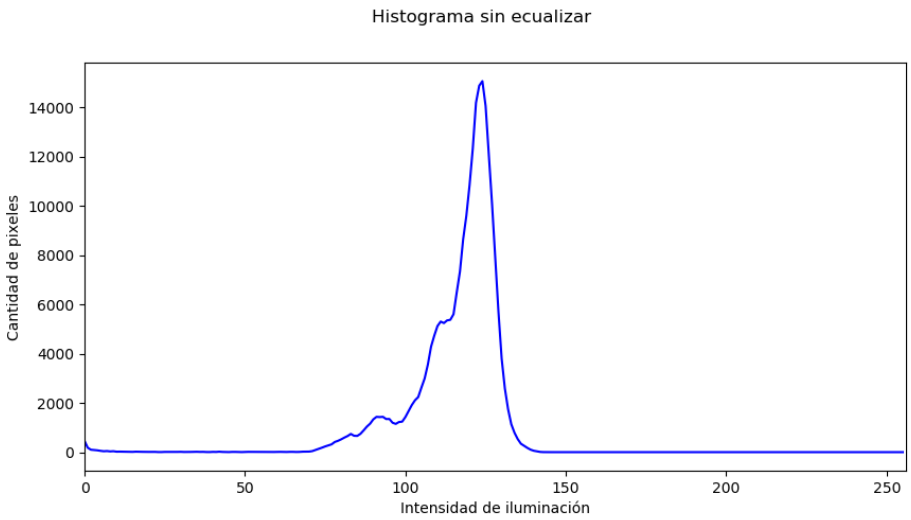
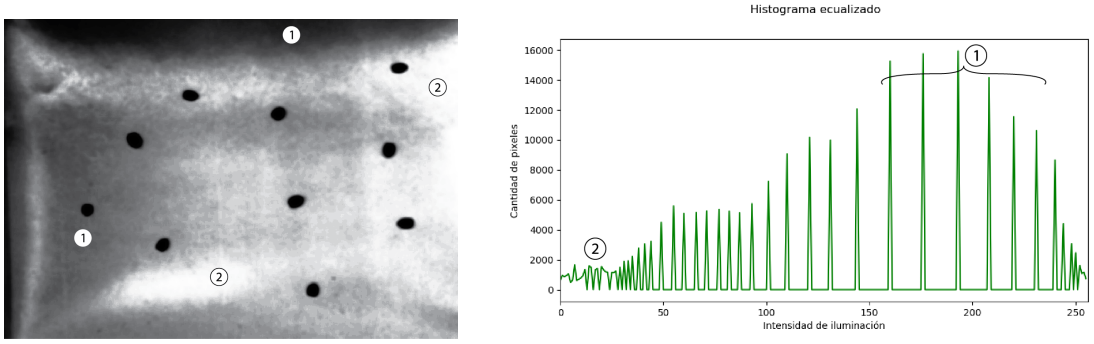


Figura 4.11: Histograma para la imagen de muestra original en escala de grises.

Al ecualizar la imagen se busca mejorar el contraste al generar una mejor distribución de los píxeles a lo largo de todos los niveles de intensidad de la fotografía. Esta práctica se realiza normalmente en fotografías con mayor detalle (por ejemplo, paisajes [6]), sin embargo, se evaluará el efecto que tiene sobre la imagen de muestra de este estudio (figura 4.7) para determinar si es efectiva su utilización en el conteo de brocas. El resultado de ecualizar la imagen de muestra se presenta en la figura 4.12a, en donde se observa a simple vista que la ecualización generó la acentuación de detalles no deseados en la imagen, tales como marcas en el papel adhesivo y sombras. Por otra parte, en el histograma resultante de la figura 4.12b se observa una mayor cantidad de píxeles en la zona de las sombras (etiqueta 1) y otras regiones con saturación lumínica (etiqueta 2).

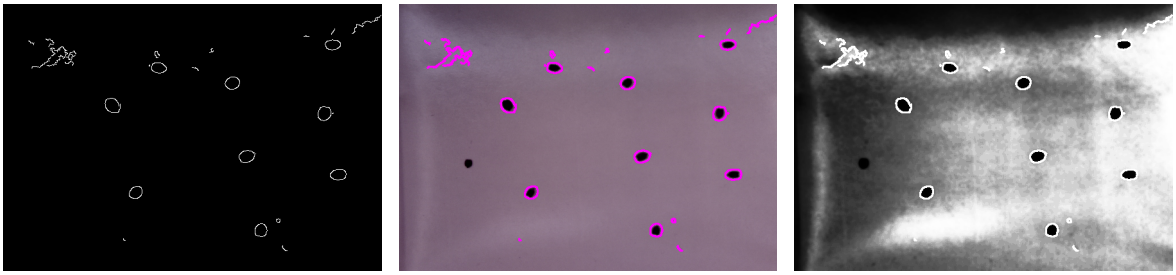


(a) (b)

Figura 4.12: Imagen de muestra tras aplicar ecualización de histograma. a) Imagen, b) histograma.

A pesar de que la práctica de ecualizado puede resultar conveniente en algunas aplicaciones, en este estudio en particular ocasiona resultados indeseados que podrían afectar el desempeño del algoritmo de Canny. Por esta razón, se evaluará la efectividad del algoritmo con la imagen ecualizada y sin ecualizar, para poder encontrar la mejor opción en el desarrollo del proyecto.

La figura 4.13a muestra una imagen binaria de los bordes que fueron detectados tras la aplicación del algoritmo de Canny posterior a la ecualización, mientras que la figura 4.13b muestra la imagen original con los contornos encontrados dibujados en esta, y la figura 4.13c los mismos contornos con la imagen ecualizada para una mejor apreciación de los resultados.

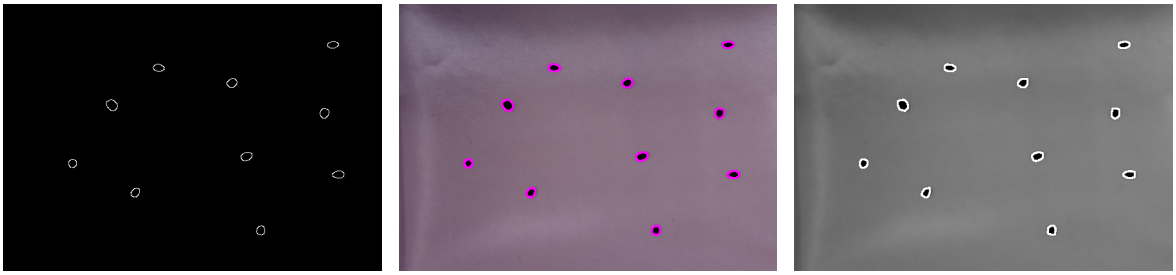


(a) (b) (c)

Figura 4.13: Resultado de aplicar el algoritmo de Canny en la imagen de muestra ecualizada.

Tras aplicar el algoritmo, en la imagen se encontraron 34 elementos a pesar de que el resultado correcto corresponde a 10. Mediante la figura 4.13 se observa que en la fotografía se detectaron como bordes las características del papel adhesivo que fueron acentuadas tras el proceso de ecualización, y no se detectaron los elementos que se encuentran en las zonas cuyo nivel de sombra fue acentuado al ecualizar.

Tras el deficiente resultado obtenido al procesar la imagen ecualizada, se decide realizar el conteo de elementos con la imagen sin ecualizar, es decir, únicamente con la conversión de espacio de color y el filtro gaussiano (figura 4.8b). Los resultados se muestran en la figura 4.14.



(a) (b) (c)

Figura 4.14: Detección de bordes de Canny para imagen de muestra ecualizada.

Bajo estas condiciones de prueba el algoritmo detectó la presencia de 10 elementos, lo cual es el resultado esperado. Adicionalmente, se observa en las figuras 4.14b y 4.14c que todos los bordes encontrados corresponden a las semillas colocadas, y no a detalles indeseados de la zona de muestreo. Esto es posible de lograr gracias a los valores de umbralización escogidos en el algoritmo de Canny, para el cual se eligió una cantidad exacta de 100 como valor óptimo para el umbral mínimo, y el umbral máximo en una cantidad de 200 tras seguir la recomendación de establecer una relación de 2:1 según la guía de uso de funciones mencionada en la documentación de OpenCV [49]. Cabe mencionar que la definición de la cantidad para el umbral mínimo es fundamentada mediante el histograma de la figura 4.11, donde los cambios en la cantidad de píxeles (es decir, donde se sugiere la presencia de objetos) comienzan a aparecer alrededor del valor 100 de intensidad de iluminación.

Tras la realización de las pruebas anteriores, se concluye que el algoritmo para realizar el conteo de brocas es más efectivo si no se ecualiza la imagen de muestra antes de procesarla con el detector de bordes de Canny.

4.3. Organización de los datos

La tarea principal que se realiza en el proceso de organización de los datos, consiste en salvar las fotografías tomadas con etiquetas para cada imagen, es decir, que este sea indicativo de la fecha y de la hora en la que fue capturada. Una vez que se realizó el procesamiento de esta (una vez que se obtiene el número de elementos que han sido encontrados en la imagen), la información es respaldada en el sistema empotrado mediante un archivo de texto con la información de la fecha del monitoreo y la cantidad de elementos que fueron encontrados para ese momento. Por ejemplo, para la imagen de muestra utilizada en la subsección 4.2, se

obtiene el archivo de texto que se muestra en la figura 4.15.

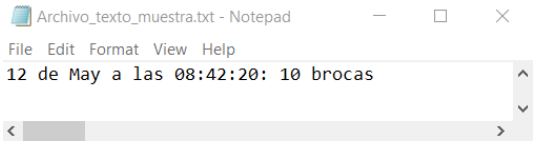


Figura 4.15: Archivo de texto como respaldo de la información.

Mediante este proceso se asegura un adecuado control de las variables que informan la fecha, hora y cantidad, de modo que es posible preparar los datos necesarios para realizar la carga a Dropbox mediante el nombre de las imágenes, y la carga de la información sobre la cantidad de elementos a ThingSpeak a través de la conexión a Internet.

4.4. Pruebas de velocidad

En esta sección se realizará la evaluación de cada una de las conexiones a Internet que se pueden lograr con el sistema empotrado, para determinar cuál opción brinda una mayor estabilidad bajo las mismas condiciones de prueba. Todas las conexiones se establecieron desde el Laboratorio de Fotogrametría del Instituto Tecnológico de Costa Rica, adicionalmente, en cada caso se realizó una cuantificación de la temperatura del sistema empotrado para poder evaluar si a través del tiempo se generan sobrecalentamientos en el sistema que deban ser tratados. El tamaño de las imágenes también es tomado en cuenta en la realización de las pruebas del proceso, pues es altamente influyente en el tiempo de la carga.

Cada prueba de velocidad se realizó para la captura, procesamiento, y carga de datos de 10 imágenes con una cantidad escogida al azar de 10 elementos. Para este punto, la cantidad de elementos no es relevante, pues lo que se requiere es determinar el tiempo que tarda el algoritmo desde que captura la fotografía hasta que finaliza la carga de los datos a ThingSpeak y no la efectividad del algoritmo DIP. Al final de la sección, se discuten los resultados obtenidos.

Ethernet

La primera conexión que se sometió a prueba fue la transferencia de datos vía Ethernet, a través del puerto RJ45 en la Raspberry Pi 3 Model +. Para el momento en el que se realizó la prueba, se analizó la velocidad de carga de la conexión, la cual resultó en 2 Mbps. Los datos obtenidos para esta conexión se muestran en la tabla 4.1.

Tabla 4.1: Prueba de velocidad del sistema de monitoreo para una conexión vía Ethernet

Muestra	**Temperatura RPI** $°C$	**Tamaño de imagen (KB)**	**Tiempo de ejecución (h:mm:ss)**
01	61.2	278.3	0:01:49
02	61.8	260	0:04:20
03	62.8	259.3	0:00:51
04	63.4	259.3	0:00:09
05	62.3	260.7	0:07:28
06	62.3	259.3	0:03:04
07	62.8	259.5	0:02:06
08	62.8	258.9	0:02:18
09	63.4	258.6	0:00:47
10	63.4	259.3	0:00:28

En esta prueba no se logró realizar la carga de todas las imágenes en el servidor de Dropbox: la fotografía obtenida en la muestra 02 y 05 no se logró cargar debido a la inestabilidad de la conexión durante las pruebas realizadas, sin embargo, la información de la cantidad de elementos en la imagen sí fue cargada en la plataforma *IoT*. La información obtenida tras la prueba es insuficiente para determinar las causas del fallo en este procedimiento, ya que mediante la tabla 4.1 se descarta el tamaño de la imagen como factor influyente, y también se descarta la existencia de un límite de tiempo para realizar el proceso, pues entre las muestras 02 y 05 hay una diferencia de 3 minutos y ninguna de las imágenes pudo ser cargada. Al tomar múltiples fotografías entre tiempos de monitoreo y realizar el cálculo de un promedio, la pérdida de un dato no sería significativa.

La tendencia en la variación de la carga de los datos, así como el valor promedio de carga por fotografía para este tipo de conexión, se muestra gráficamente en la figura 4.16

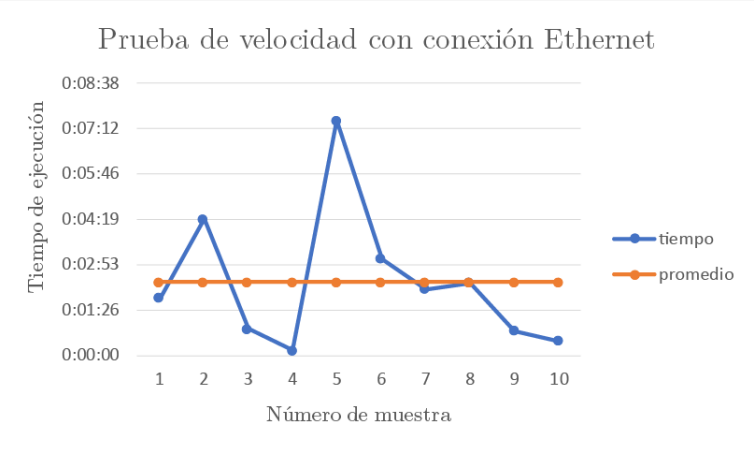


Figura 4.16: Tiempo de carga de imágenes con una conexión Ethernet.

El tiempo promedio de carga para esta conexión resulto en 0:02:20, para un tiempo total de ejecución de 0:23:20. Se utiliza el *modelo de regresión lineal* [50] para estimar el tiempo que le tomaría al algoritmo desarrollado cargar una cantidad de x fotografías conectado a una red Ethernet con las características mencionadas anteriormente. Esto se realiza con el objetivo de predecir la eficiencia del sistema conectado a cierta red, y poder dar al usuario una información certera de las condiciones de funcionamiento del sistema de monitoreo.

Con el modelo de regresión lineal se obtiene la ecuación 39.

$$tiempo = 2{,}61 * (muestras) - 0{,}033 \tag{39}$$

Para analizar la efectividad de las conclusiones que se pueden obtener mediante la ecuación 39, se obtiene el coeficiente de regresión lineal r, tal que

$$r = -0{,}235 \tag{40}$$

El signo negativo del resultado anterior indica que para este grupo de datos la tendencia fue la disminución del tiempo de carga conforme aumentó el número de muestras, sin embargo, al ser su valor cercano a 0, se concluye que la correlación entre las variables es débil, lo cual se fundamenta con el gráfico de la figura 4.16 al observar la separación entre las muestras. Para obtener una mejor caracterización de los resultados, conviene calcular un porcentaje de error con respecto a los valores experimentales, tal que:

$$\%error \approx 12\,\% \tag{41}$$

Con los datos de la tabla 4.1 se observa que el tamaño de la imagen no influye significativamente en la velocidad de carga, pues para algunas imágenes con mayor tasa de información se toma un menor tiempo de carga, y viceversa; sin embargo, este hecho se analizará al final de todas las pruebas, para obtener un resultado concluyente.

Wi-Fi

Tras realizar las pruebas de velocidad para la red Wi-Fi utilizada en el Laboratorio de Fotogrametría de la Escuela de Electrónica del Instituto Tecnológico de Costa Rica, se determinó su velocidad de carga en 10.42 Mbps. Los datos obtenidos para estas pruebas de velocidad se muestran en la tabla 4.2.

Tabla 4.2: Prueba de velocidad del sistema de monitoreo para una conexión Wi-Fi local

Muestra	**Temperatura RPI** $°C$	**Tamaño de imagen (KB)**	**Tiempo de ejecución (h:mm:ss)**
01	63.4	274.1	0:00:14
02	62.8	262.0	0:00:10
03	63.4	264.0	0:00:10
04	63.4	260.2	0:00:10
05	62.8	260.3	0:00:10
06	62.8	259.2	0:00:12
07	62.3	260.2	0:00:13
08	63.4	259.6	0:00:25
09	63.4	260.6	0:00:11
10	62.3	260.5	0:00:10

Contrario al comportamiento de la red Ethernet, para la red Wi-Fi local utilizada todas las fotografías se cargaron al servidor de Dropbox. La representación gráfica del tiempo de ejecución del programa para la conexión, se muestra en la figura 4.17.

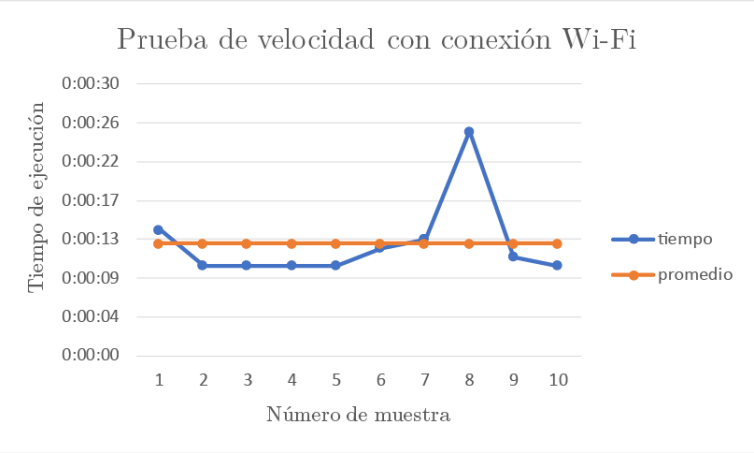


Figura 4.17: Tiempo de carga de imágenes con una conexión Wi-Fi.

El tiempo promedio de carga por fotografía resultó de 0:00:13. El tiempo total de ejecución del programa fue de 0:02:05. Al utilizar el modelo de regresión lineal para obtener una ecuación aproximada según los datos de la tabla 4.2, se obtiene

$$tiempo = 0{,}252 * (muestras) - 0{,}215 \tag{42}$$

El coeficiente de regresión lineal r para los datos de la tabla 4.2 es:

$$r = 0{,}226 \tag{43}$$

Debido a que el valor del coeficiente de regresión lineal es cercano a 0, se concluye que para este grupo de datos la correlación entre las variables también es débil, con una tendencia a aumentar el tiempo de carga conforme aumenta el número de muestra. El porcentaje de error con respecto a los valores experimentales es de

$$\%error \approx 18\,\% \tag{44}$$

Red celular Kölbi ICE

La conexión a la red celular de las tres compañías telefónicas principales (tabla 3.5), se realizó mediante una conexión Wi-Fi desde la Raspberry utilizando un celular utilizado como punto de acceso.

Al medir la velocidad de conexión con la red Kölbi, se obtienen los datos mostrados en la tabla 4.3.

Tabla 4.3: Prueba de velocidad del sistema de monitoreo para una conexión a red Kölbi

Muestra	**Temperatura RPI** $°C$	**Tamaño de imagen (KB)**	**Tiempo de ejecución (h:mm:ss)**
01	62.3	278.2	0:00:25
02	62.3	262.9	0:00:24
03	62.8	260.8	0:00:47
04	63.4	259.4	0:00:20
05	62.8	259.6	0:00:25
06	62.8	258.7	0:00:45
07	62.3	260.3	0:00:25
08	62.3	258.0	0:00:23
09	63.4	259.1	0:00:23
10	63.4	259.0	0:00:34

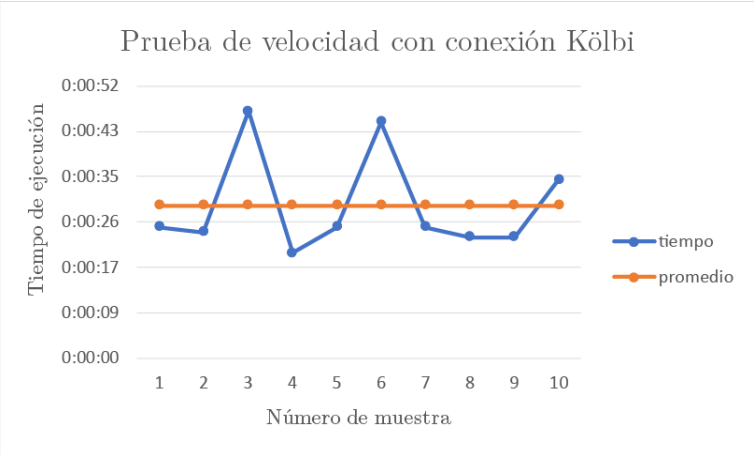


Figura 4.18: Tiempo de carga de imágenes con una conexión de red celular Kölbi.

El tiempo promedio de carga para esta conexión resultó de 0:00:25, para un tiempo de ejecución total de 0:04:51. Realizando el análisis mediante regresión lineal, se obtiene el tiempo de carga para una muestra x según

$$tiempo = 0{,}492 * (muestras) - 0{,}027 \tag{45}$$

El coeficiente de regresión lineal r para los datos de la tabla 4.3 es:

$$r = -0{,}021 \tag{46}$$

Debido a que el valor del coeficiente de regresión lineal es cercano a 0, se concluye que para este grupo de datos la correlación entre las variables también es débil, con una tendencia a disminuir el tiempo de carga conforme aumenta el número de muestra. El porcentaje de error con respecto a los valores experimentales es de:

$$\%error \approx 0{,}89\,\% \tag{47}$$

Al comparar el porcentaje de error con el obtenido para resultados anteriores, la regresión lineal de esta ecuación supone una muy buena aproximación para los datos obtenidos.

Red celular Claro

La tabla 4.4 muestra los resultados obtenidos tras la prueba de velocidad con una red celular Claro como punto de acceso a través de un celular, el mismo utilizado para la prueba de la conexión anterior. Por su parte, en la figura 4.19 se observa gráficamente la cuantificación del tiempo de ejecución, para un tiempo promedio de 0:00:20 y un tiempo total de 0:03:16. En esta, se visualiza una mayor estabilidad en los datos con respecto a las conexiones anteriores, pues únicamente la muestra 04 se encuentra por encima del promedio.

Tabla 4.4: Prueba de velocidad del sistema de monitoreo para una conexión a red Claro

Muestra	Temperatura RPI $°C$	Tamaño de imagen (KB)	Tiempo de ejecución (h:mm:ss)
01	62.3	271.1	0:00:24
02	63.4	261.3	0:00:16
03	62.3	259.9	0:00:16
04	62.3	258.2	0:00:33
05	62.8	257.4	0:00:18
06	62.3	257.9	0:00:18
07	62.3	257.8	0:00:18
08	62.8	257.5	0:00:19
09	63.4	257.5	0:00:17
10	62.8	257.9	0:00:17

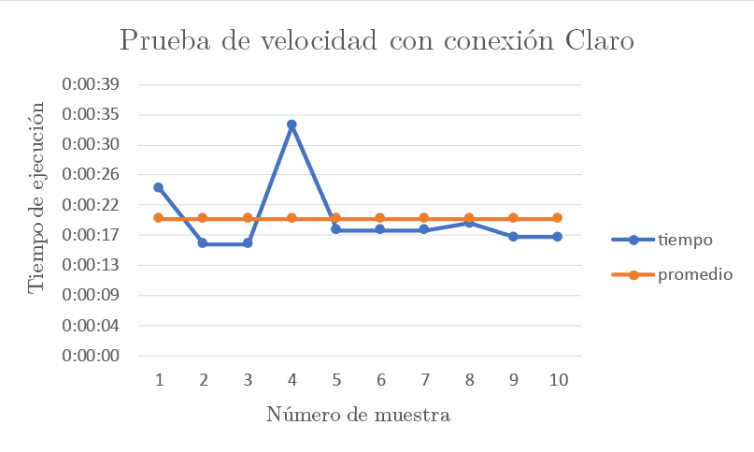


Figura 4.19: Tiempo de carga de imágenes con una conexión de red celular Claro.

Tras el análisis de regresión lineal, se obtiene para esta conexión

$$tiempo = 0{,}331 * (muestras) + 0{,}120 \tag{48}$$

El coeficiente de regresión lineal r para los datos de la tabla 4.4 es:

$$r = -0{,}302 \tag{49}$$

Debido a que el valor del coeficiente de regresión lineal es cercano a 0, se concluye que para este grupo de datos la correlación entre las variables también es débil, con una tendencia a disminuir el tiempo de carga conforme aumenta el número de muestra. El porcentaje de error con respecto a los valores experimentales es de:

$$\%error \approx 6{,}73\,\% \tag{50}$$

Red celular Movistar

Los datos obtenidos para esta conexión de red se muestran en la tabla 4.5. La figura 4.20 muestra la visualización gráfica de los datos, con los que se obtuvo un tiempo promedio de carga de 0:00:17 y un tiempo de ejecución total de 0:02:46.

Tabla 4.5: Prueba de velocidad del sistema de monitoreo para una conexión a red Movistar

Muestra	**Temperatura RPI** $°C$	**Tamaño de imagen (KB)**	**Tiempo de ejecución (h:mm:ss)**
01	63.4	271.1	0:00:13
02	62.8	260.3	0:00:14
03	62.3	258.9	0:00:14
04	62.3	260.2	0:00:23
05	62.3	260.4	0:00:18
06	62.3	258.9	0:00:17
07	62.8	262.8	0:00:15
08	63.4	261.5	0:00:16
09	62.8	259.5	0:00:21
10	62.8	257.9	0:00:15

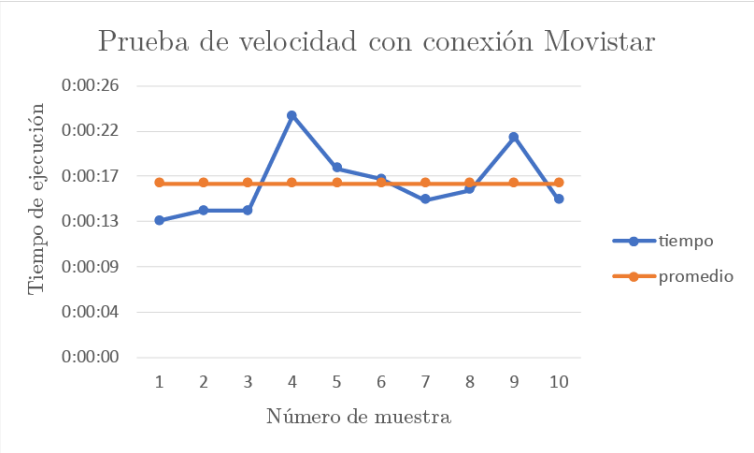


Figura 4.20: Tiempo de carga de imágenes con una conexión de red celular Movistar.

Mediante el análisis de regresión lineal se obtiene

$$tiempo = 0{,}278 * (muestras) + 0{,}011 \tag{51}$$

El coeficiente de regresión lineal r para los datos de la tabla 4.5 es:

$$r = 0{,}295 \tag{52}$$

Debido a que el valor del coeficiente de regresión lineal es cercano a 0, se concluye que para este grupo de datos la correlación entre las variables también es débil, con una tendencia

a aumentar el tiempo de carga conforme aumenta el número de muestra. El porcentaje de error con respecto a los valores experimentales es de:

$$\%error \approx 0{,}76\,\% \tag{53}$$

el cual es el valor más bajo de porcentaje de error obtenido durante las pruebas.

Resultados de las pruebas de velocidad

El menor tiempo de ejecución del programa para una muestra de 10 fotografías fue logrado para una red Wi-Fi local con una velocidad aproximada de carga de 10 Mbps. A pesar de que durante las pruebas el sistema se sometió a las mismas condiciones (lugar y hora del día), concluir que la conexión a una red Wi-Fi es la mejor opción no es un resultado preciso, pues la velocidad de carga dependerá del proveedor de Internet y del plan de datos que se haya contratado.

El valor del coeficiente de regresión lineal obtenido para cada una de las pruebas, indica que la relación entre el número de muestras y el tiempo de carga es muy débil, razón por la cual las ecuaciones obtenidas tras aplicar el modelo de regresión lineal no deben considerarse una aproximación precisa.

Debido a que la conexión de red Ethernet fue la más lenta de las cinco pruebas, las tres conexiones restantes son las de las redes celulares. Este resultado es positivo, debido a que estas redes proveen un punto de acceso desde prácticamente cualquier parte del país, y mediante los módem y *datacard* es posible obtener una conexión Wi-Fi a la cual ligar el sistema empotrado. De las tres redes evaluadas, la que resultó en una mejor conexión y una ecuación de regresión lineal con menor porcentaje de error para el área de cobertura de la zona de pruebas, fue la red Movistar 4G, por lo que se elige como la opción más viable para conectarse a la red.

Los datos obtenidos para la temperatura y tamaño de las imágenes de la tabla 4.1 a la tabla 4.5 se mantuvieron en un rango estable durante todo el proceso de prueba. La temperatura del sistema empotrado se mantuvo dentro de un rango de $61{,}5°C$ y $63{,}5°C$ para todas las pruebas realizadas, con lo cual se puede concluir que el sistema no requiere de un disipador de calor, pues con todos los módulos conectados y un funcionamiento en ambientes al aire libre, su temperatura no superó los $75°C$ [33]. Por su parte, los datos para el tamaño de las imágenes se mantuvieron en un rango de 255 a 280 KB, lo cual es determinante en la velocidad de carga. Este es un resultado positivo que se quería lograr al seleccionar una ROI adecuada al inicio de las pruebas, pues mediante estos ajustes se logra reducir el tamaño de la imagen tomada desde el sensor para una manipulación más rápida tanto en procesos de cálculo (algoritmo DIP) como en respaldo de datos (carga de imágenes al servidor).

4.5. Funcionamiento autónomo del sistema

La propuesta para realizar el monitoreo de la plaga en el campo, es utilizar una batería portátil con las características de energía suficientes para garantizar el correcto funcionamiento de la Raspberry (5V y 2.5 A) [33]. A su vez, esta batería podría conectarse a un cargador solar para garantizar la energía suficiente durante todo el período de cosecha. Estas consideraciones, sin embargo, no se analizan con profundidad al no formar parte directamente de los objetivos de este estudio. Las pruebas realizadas en esta sección utilizan una batería portátil de 20 000 mAh para alimentar al sistema empotrado.

Tras haber comprobado el funcionamiento de los módulos individualmente, se somete el sistema a pruebas de funcionamiento durante varias horas para evaluar al sistema bajo distintas condiciones de luz natural y la interacción con el usuario en los tiempos especificados para la toma de muestras, además del rendimiento al conectar la Raspberry Pi 3 a una fuente de alimentación externa. Las pruebas se realizaron desde las 08:00 hasta las 18:00, pues en horas posteriores las condiciones de iluminación no permitían una captura eficiente de la zona de muestreo.

A través de los datos cargados a ThingSpeak, fue posible obtener los datos según el tiempo de monitoreo establecido. Estos resultados se muestran en la figura 4.21.

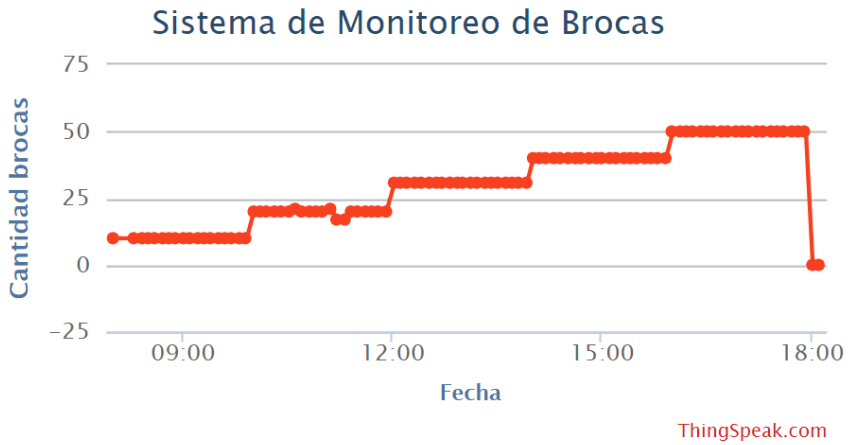


Figura 4.21: Comportamiento de los datos para un día completo de pruebas continuas.

Para movilizar la cantidad de elementos dentro de la zona de muestreo y para comprobar nuevamente el funcionamiento adecuado del algoritmo de DIP, se decidió aumentar la cantidad de semillas en el papel adhesivo cada 2 horas, de modo que fuera posible obtener 20 fotografías de la zona de muestreo entre cada tramo de monitoreo (es decir, cada 6 minutos).

El primer tramo de evaluación (08:00-10:00) se realizó con una cantidad de 10 elementos. Una de las tomas para estas condiciones se muestra en la figura 4.22.

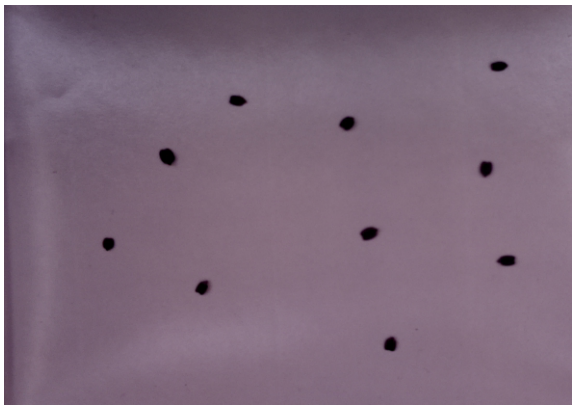


Figura 4.22: Zona de muestreo para pruebas de 08:00-10:00.

La figura 4.23 muestra un acercamiento de la primera sección de pruebas de la figura 4.21, en donde se observa una discontinuidad en las primeras muestras. Esta discontinuidad se originó debido a la pérdida de conexión de la red, pues el celular utilizado como punto de acceso tuvo que ser nuevamente configurado a solicitud del sistema operativo, sin embargo, estas dificultades técnicas son ajenas al funcionamiento del sistema y dependen más de la calidad del módem utilizado, como se mencionó en secciones anteriores. Además, la figura 4.23 también muestra la efectiva contabilización de los elementos en la zona de muestreo.

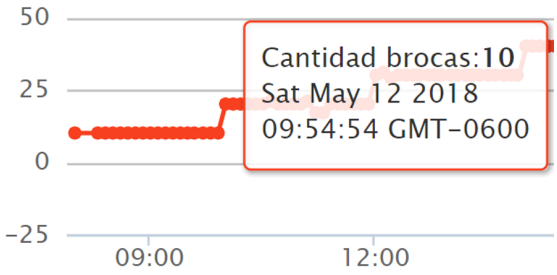
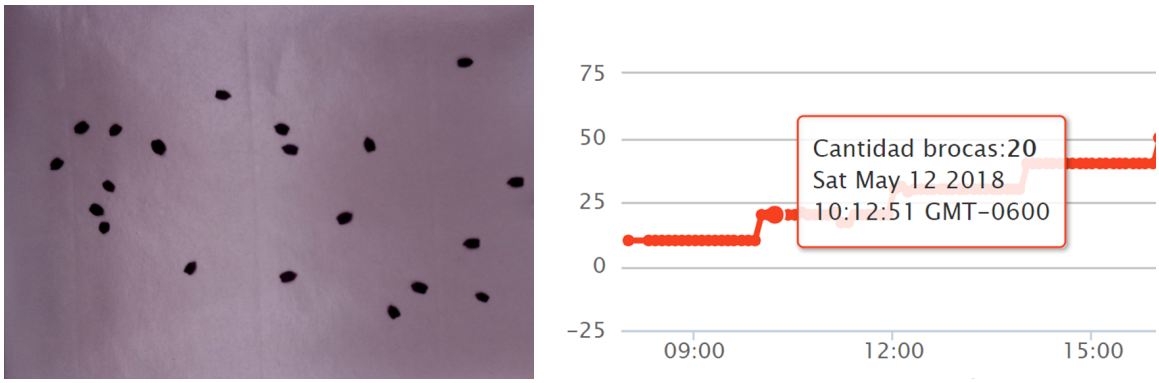


Figura 4.23: Datos obtenidos para pruebas de 08:00-10:00.

En el segundo tramo de pruebas en la figura 4.21 se aumentó en 10 la cantidad de elementos en la zona de muestreo, para un total de 20 (figura 4.24a). La figura 4.24b ilustra la efectividad en el conteo del algoritmo DIP, al detectar 20 elementos.



(a) (b)

Figura 4.24: Pruebas de monitoreo de 10:00-12:00: a) zona de muestreo, b) acercamiento de datos en ThingSpeak.

Las discontinuidades presentes en los datos en horas posteriores son variaciones en la cantidad de elementos detectados. La primera de estas se dio a causa de una hormiga que se introdujo en el papel adhesivo, como se muestra en la figura 4.25.

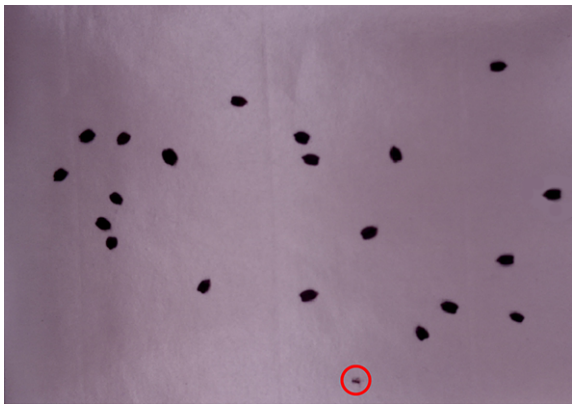


Figura 4.25: Muestra del 12 de Mayo a las 10:36:20.

Este resultado destaca la efectividad del algoritmo al detectar pequeños insectos en la zona de muestreo, lo cual es una ventaja debido a que el tamaño de las semillas de chan utilizadas representan el tamaño máximo de la broca, sin embargo, el insecto en su etapa adulta puede variar entre un tamaño de 0.85 mm y 1.5 mm.

Las siguientes discontinuidades que se observan en el segundo tramo de datos (figura 4.21) se deben a la variación intencional del número de muestras en el papel adhesivo, para evaluar la efectividad del sistema al obtener un promedio ante errores de este tipo. Para la evaluación, se modificó aleatoriamente la cantidad de muestras en la zona de muestreo de modo que la cantidad de muestras corresponden según la tabla 4.6.

Tabla 4.6: Cantidad de elementos en la zona de muestreo para el periodo entre 10:00-12:00

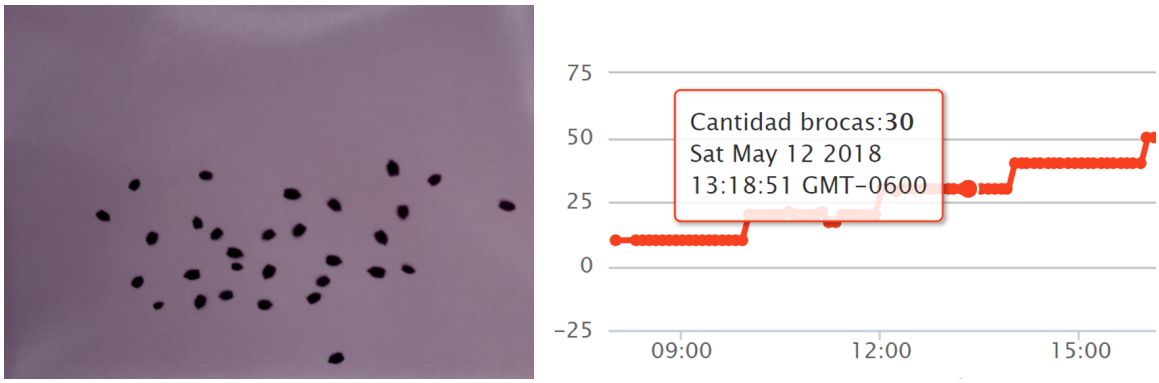
Muestra	**Cantidad de elementos**
01	20
02	20
03	20
04	20
05	20
06	20
07	21
08	20
09	20
10	20
11	20
12	20
13	21
14	20
15	17
16	17
17	20
18	20
19	20
20	20

Para el comportamiento mostrado en la tabla 4.6, se obtiene un promedio de elementos en el tiempo de monitoreo de

$$elementos = 19{,}8 \approx 20$$

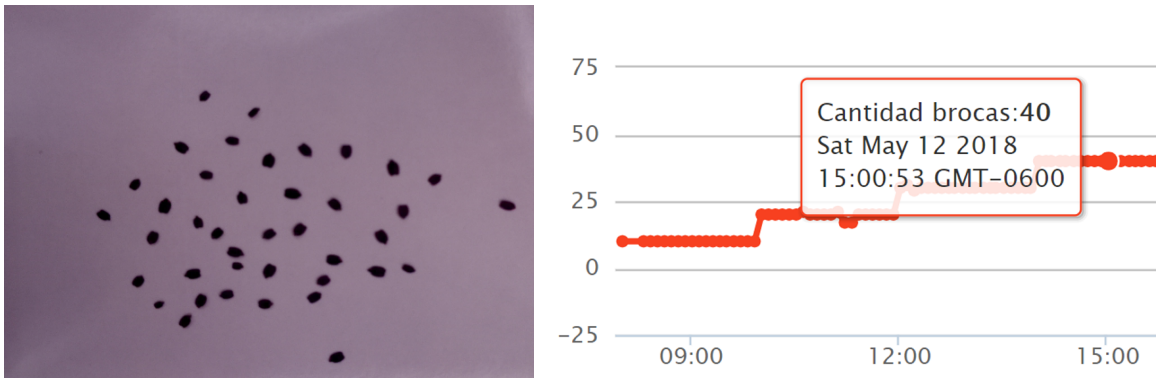
Mediante esta prueba, es posible observar que los pequeños errores se minimizan fácilmente al obtener el promedio de todas las muestras durante un único período.

Para los siguientes tiempos de monitoreo de la trampa, se decidió continuar únicamente con el incremento de 10 elementos cada 2 horas, y vigilar el sistema continuamente para evitar la introducción de algún insecto u objeto no deseados. El algoritmo fue capaz de contar con efectividad la cantidad de elementos presente para cada uno de los tiempos, como se evidencia en las figuras 4.26-4.28.



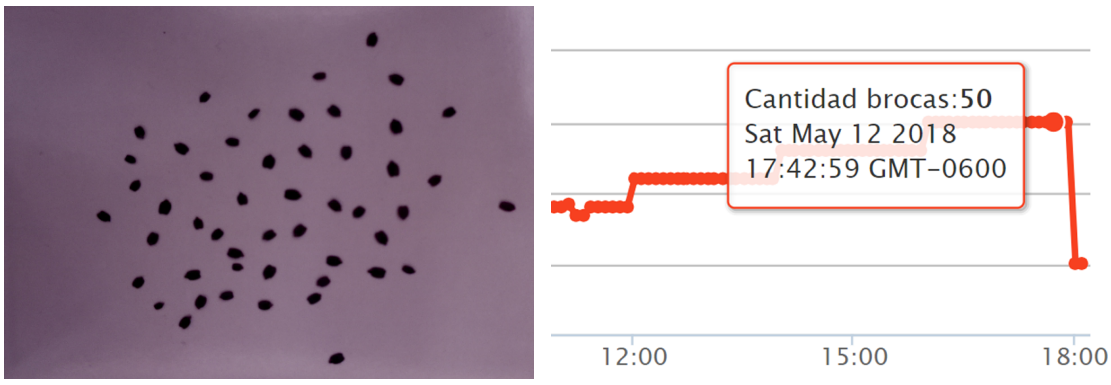
(a) (b)

Figura 4.26: Periodo de monitoreo de 12:00-14:00.



(a) (b)

Figura 4.27: Periodo de monitoreo de 14:00-16:00.

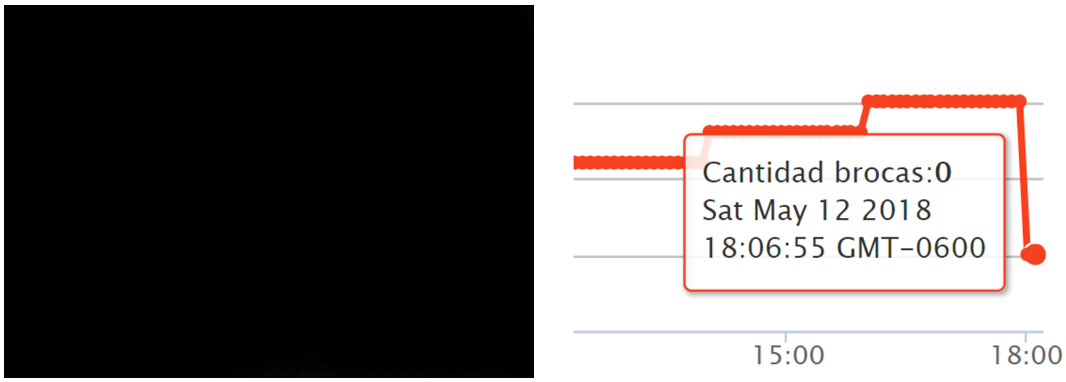


(a) (b)

Figura 4.28: Periodo de monitoreo de 16:00-18:00.

El monitoreo del sistema se realizó hasta las 18:00 debido a que las condiciones de luz no serían suficientes para obtener una captura efectiva de los elementos en la zona de muestreo,

sin embargo, se realizó el incremento de 10 elementos y se capturó la muestra para comprobarlo. Esto se evidencia con la figura 4.29, en donde se aprecia únicamente una imagen oscura y un conteo de 0 elementos.



(a) (b)

Figura 4.29: Muestra de elementos después de las 18:00.

La tabla 4.7 resume los resultados de las pruebas realizadas, mediante los cuales se puede concluir que el sistema funciona con una precisión mayor al 95 % bajo condiciones ideales de pruebas.

Tabla 4.7: Resultados de las pruebas de funcionamiento autónomo del sistema de monitoreo

Periodo	**Cantidad de elementos en la muestra**	**Cantidad promedio de elementos detectados**	**Porcentaje de error**
08:00-10:00	10	10	0
10:00-12:00	20	19.8	1 %
12:00-14:00	30	30	0
14:00-16:00	40	40	0
16:00-18:00	50	50	0

4.6. Verificación del sistema en condiciones extremas

Las pruebas presentadas se han realizado con una distribución estratégica de elementos en la zona de muestreo, sin embargo, en los campos de café las condiciones no son tan ideales. Para describir el comportamiento del sistema en estas condiciones, se realizan pruebas simulando el traslape de semillas en la zona de muestreo, como se muestra en la figura 4.30.

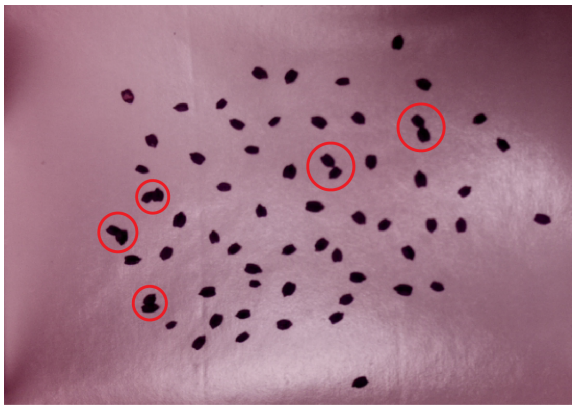


Figura 4.30: Muestra de semillas traslapadas.

En esta evaluación se tomó un período de muestreo de dos horas, dividido en cuatro secciones con una duración de 30 minutos cada una, sin embargo, los resultados obtenidos difieren en gran medida de los esperados, como se evidencia en la figura 4.31 y en la tabla 4.8.

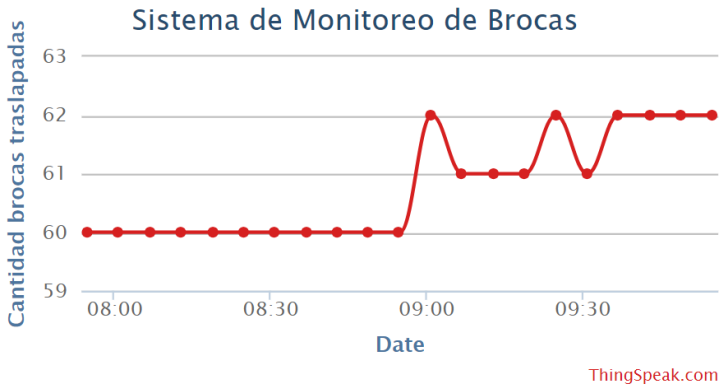


Figura 4.31: Resultado del conteo en una muestra de semillas traslapadas.

Tabla 4.8: Resultados de las pruebas con elementos traslapados

Periodo	**Cantidad de elementos en la muestra**	**Cantidad promedio de elementos detectados**	**Porcentaje de error**
08:00-08:30	65	60	8 %
08:30-09:00	70	60	14 %
09:00-09:30	75	61	19 %
09:30-10:00	80	61.8	23 %

Se analiza la salida del detector de bordes de Canny del algoritmo de procesamiento de imágenes para la muestra de la figura 4.30, la cual se muestra en la figura 4.32.

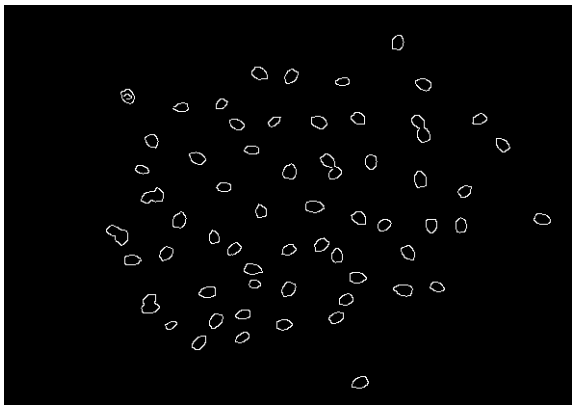
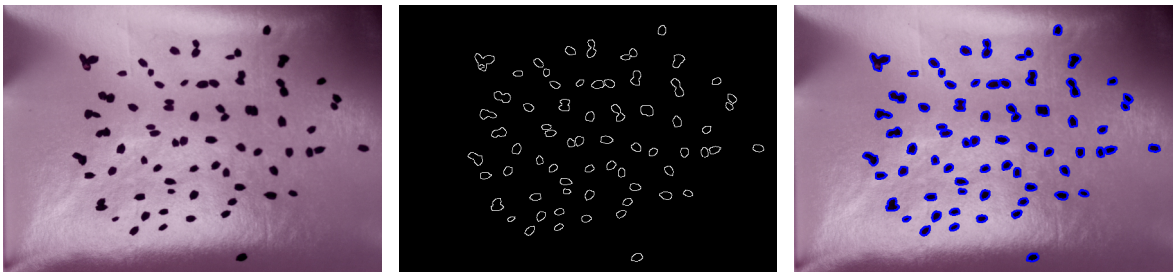


Figura 4.32: Detector de bordes de Canny para muestra de semillas traslapadas.

La figura 4.32 muestra los contornos detectados en la imagen, en donde se evidencia que los elementos traslapados son tratadas como un único contorno, esto debido a que sus niveles de intensidad de luz no difieren lo suficiente como para ser tratados como contornos individuales. De igual manera, se muestra en la figura 4.33 la fotografía real y los bordes detectados para la muestra de 80 elementos, de la cual únicamente fueron detectados 62.



(a) (b) (c)

Figura 4.33: Conteo de la plaga para una muestra con elementos traslapados.

Tras analizar los resultados, se concluye que el sistema no discrimina efectivamente a los elementos que se encuentran muy cercanos unos de otros, por lo que el conteo en esos casos no resulta efectivo. Esta dificultad podría solucionarse al implementar un algoritmo que segmente los objetos dentro de la fotografía, sin embargo, la muestra que se obtiene tras el análisis, es suficiente para determinar una cantidad promedio de brocas en dicho tiempo y obtener resultados concluyentes del comportamiento de la broca en dicho período.

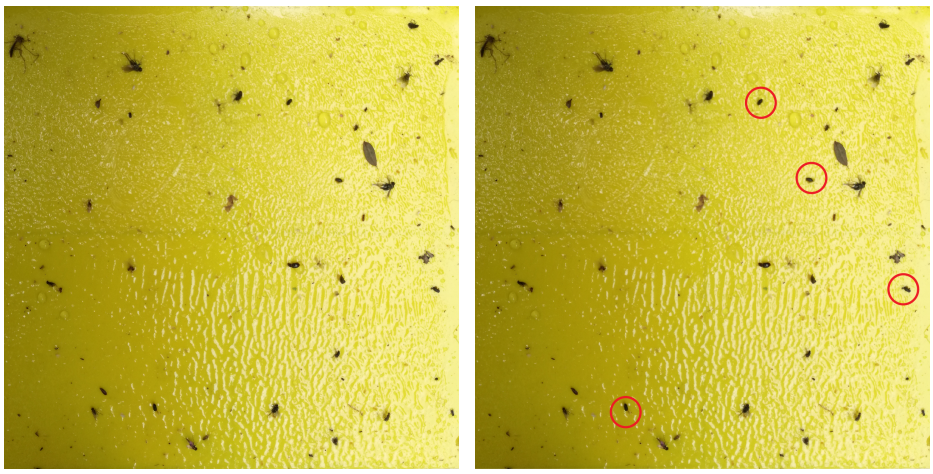
Las oscilaciones que se muestran en el período 09:00-09:30 se deben principalmente a la configuración de las condiciones de prueba y los materiales utilizados para la simulación. Debido a la textura sólida de las semillas de chan y a la poca fuerza de amarre del papel adhesivo utilizado, las semillas en algunas ocasiones cambiaban la posición original en la que

fueron colocadas (por ejemplo, se colocaron en una posición vertical que resultó inestable debido a la forma de la semilla y, con el tiempo, terminaron en una posición vertical sobre el papel), generando más traslapes o bien, provocando una distancia significativa con respecto a otras semillas.

4.7. Verificación del sistema en condiciones reales

Para realizar la evaluación del tamaño óptimo del medio de simulación, la trampa se colocó en una de las plantaciones de café del Instituto del Café de Costa Rica. Es importante mencionar que esta trampa simulada no brinda los resultados que se esperan obtener con la primera etapa del sistema de monitoreo, pues permite la entrada de especies de diversos tamaños a la zona de muestreo, dificultando el reconocimiento de únicamente brocas. Sin embargo, se analiza el comportamiento del sistema para elaborar las recomendaciones y consideraciones del proyecto tanto para la primera como para la segunda etapa de monitoreo.

La figura 4.34a muestra una captura de la zona de muestreo tras su permanencia en una de las plantaciones del Icafé, y la figura 4.34b muestra las brocas que se encuentran en la zona de captura.



(a) (b)

Figura 4.34: Muestra de brocas obtenida mediante la trampa simulada (1).

Al aplicar el algoritmo a la imagen obtenida de un ambiente real, este informa al usuario de la presencia de 452 brocas, es decir, un error de 112 unidades. Los contornos detectados por el algoritmo se muestran en la figura 4.35, en donde se evidencia que la alta de cantidad de brocas detectadas se debe principalmente a la no-discriminación de objetos, pues detecta como brocas a todos los insectos atrapados, hojas, y textura del papel adhesivo utilizado.

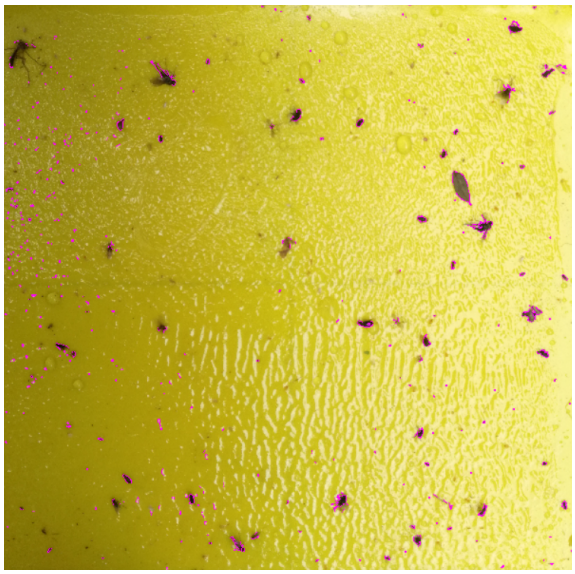
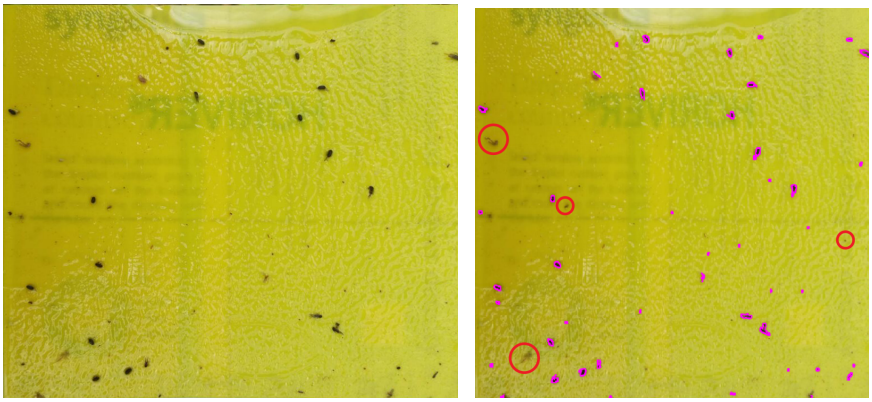


Figura 4.35: Brocas detectadas en una muestra real obtenida mediante la trampa simulada.

Para mejorar la respuesta del algoritmo ante estas condiciones, se aumentó el tamaño del kernel utilizado en el proceso de filtrado a 5×5, y se evaluó el comportamiento en la imagen de muestra de la figura 4.36a. Las brocas detectadas se muestran en la figura 4.36b.



(a) (b)

Figura 4.36: Muestra de brocas obtenida mediante la trampa simulada (2).

En la zona de muestreo se encuentran presentes 13 brocas, y el algoritmo informó de una cantidad de 46, para un porcentaje de error de 253 %. A pesar de que el error es menor, al aumentar el tamaño de la máscara se discriminan algunos objetos dentro de la imagen que podrían ser brocas debido a su tamaño, como los encerrados en la figura 4.36b. Por esta razón, se concluye que modificar el algoritmo no es una solución viable, sino que se debe mejorar el medio de recolección de muestras para filtrar lo máximo posible a los objetos que no corresponden a brocas y obtener una mayor precisión en el conteo.

4.8. Costo del prototipo del sistema de monitoreo de brocas

Uno de los requerimientos evaluados en conjunto con el Icafé para el sistema de monitoreo, implica que el sistema desarrollado tenga un costo de diseño a nivel de prototipo menor a $200, por lo que se realiza una evaluación del material utilizado mediante la tabla 4.9.

Tabla 4.9: Análisis de costo del sistema de monitoreo

Recurso	**Costo**
Raspberry Pi 3 Model B+	$54.95
SIM Card	$1
NoIR Camera V2	$25
Power Bank con cargador solar	$40.99
Impresión en acrílico de la trampa	$30
Total	$151.94

El costo de $200 como máximo fue establecido por los expertos del Icafé con base en el gasto en trampas manuales por período de cosecha. El sistema desarrollado tiene un costo aproximado de $152, lo cual está por debajo del requerimiento establecido y se puede aprovechar en la compra del material adhesivo y de las sustancias atrayentes de brocas.

5. Conclusiones

La segunda etapa del sistema de monitoreo propuesto en este estudio se desarrolló utilizando un sistema empotrado como núcleo central de funcionamiento. El sistema permitió programar el disparo automático de una cámara de 8 Mpx integrada en un medio de recolección de muestras para después procesar la imagen obtenida, e informar al usuario con una efectividad mayor al 90 % sobre la cantidad de brocas que encontró en la imagen en condiciones ideales. El tiempo de ejecución de un ciclo de pruebas según la cantidad de muestras requeridas se logró aproximar mediante

$$tiempo = 0{,}278 * (muestras) + 0{,}011$$

Las fotografías que se obtuvieron de la zona de muestreo cuentan con una GSD de 73,8125 μm por píxel, lo cual permitió al algoritmo de procesamiento digital de imágenes obtener los contornos de las brocas, cuyo tamaño varía entre 0.85 y 1.5 mm, es decir, una única broca puede abarcar 20 píxeles en la imagen. Esto hace posible el reconocimiento de objetos pequeños a una distancia de 20cm, distancia evaluada como óptima entre el sensor y la zona de muestreo.

Para condiciones de prueba ideales, es decir, cuando no se da un traslape en la ubicación de las muestras sobre el papel adhesivo, el algoritmo detectó las brocas presentes con una efectividad mayor al 90 %, sin embargo, para condiciones no ideales, no fue posible cuantificar la efectividad del sistema debido a la ubicación aleatoria que pueden tomar las brocas dentro de la zona de muestreo a través del tiempo. Bajo condiciones de prueba con imágenes de muestras reales, el sistema no funcionó eficazmente debido a que se dio la detección de insectos y objetos que no corresponden a brocas. Además, el material adhesivo utilizado contiene texturas que fueron identificadas como contornos por el algoritmo de procesamiento de la imagen.

Mediante una conexión a Internet a través de la red celular 4G de Movistar y el uso de un celular como módem, el algoritmo fue capaz de realizar la carga de las imágenes obtenidas al servidor de Dropbox y la información de la cantidad de brocas a la base de datos de ThingSpeak, lo cual permitió la visualización de las estadísticas de una manera interactiva para el usuario.

El sistema desarrollado es capaz de:

- Capturar imágenes de la zona de muestreo a la hora indicada por el usuario.
- Contar los objetos presentes en la imagen, correspondan o no a brocas, con una precisión mayor al 90 % en condiciones ideales.
- Almacenar los datos en un registro y enviar la información al usuario a través de Internet.

- Realizar todas las funciones anteriores remotamente y por un costo menor a $200.

5.1. Recomendaciones

La primera etapa del sistema debe filtrar lo máximo posible las especies que son atrapadas. Utilizando agujeros más pequeños a los lados, podría evitarse la entrada de hojas e insectos de gran tamaño. Además, al utilizar un papel adhesivo sin textura, se puede mejorar la detección por parte del algoritmo DIP.

Para disminuir la frecuencia de interacción entre el usuario y la trampa de la primera etapa del sistema, se recomienda automatizar el proceso mediante el uso de motores giratorios para cambiar el papel adhesivo de la zona de muestreo, según el tiempo de monitoreo establecido.

La calidad de las imágenes obtenidas puede mejorarse al utilizar un sensor con mejores características físicas, por ejemplo, la capacidad de auto-enfoque y una herramienta de zoom más precisa.

Es posible discriminar entre brocas traslapadas mediante la implementación de algoritmos más complejos, en los que se aproveche el detalle brindado por un mejor sensor para aplicar técnicas como el algoritmo de watershed, o bien, se puede utilizar una alternativa al procesamiento digital de imágenes, como la aplicación del deep learning.

Para evitar inestabilidad en las condiciones de la red debido a la lejanía del punto de acceso Wi-Fi, es recomendable conectarse mediante un módem de mejor calidad que un celular, o utilizar un módulo GSM de bajo consumo energético y conectarlo a los GPIO del sistema empotrado.

La instalación de las bibliotecas necesarias para el procesamiento de las imágenes, deben realizarse utilizando la capacidad de los 4 núcleos de procesamiento de la Raspberry Pi 3 Model B+, para garantizar una mayor rapidez en el proceso con los mismos resultados.

El sistema puede alimentarse autónomamente mediante una batería portátil con paneles solares, para obtener energía durante todo el período de cosecha.

6. Bibliografía

[1] CATIE. (1994). *"Manejo Integrado de Plagas".* Informe final del Instituto Nicaragüense de Tecnología Agropecuaria.

[2] Rojas, M. (2015). *"Manejo Integrado de la Broca del Café en Costa Rica".* Centro de Investigaciones en Café, Icafé.

[3] ANSYS. Inc. Engineering of the Internet of Things. White Paper. 2016.

[4] Aldein, M., Ali, E. (2017). *"Internet of Things Applications, Challenges and Related Future Technologies".* Electrical and Electronic Engineering Department, Red Sea University, Sudan.

[5] Derguech, W., Burke, E. et Curry, E. (2014). *"An Autonomic Approach to Real Time Predictive Analytics Using Open Data and Internet of Things".* Insight Centre for Data Analytics, National University of Irerland, Galway.

[6] González, R., Woods, R. (s.f.). *"Digital Image Processing".* Prentice Hall, New Jersey. 2da. Edición.

[7] Alvarado, P. (s.f.). *"Procesamiento Digital de Imágenes".* Escuela de Ingeniería Electrónica, Tecnológico de Costa Rica. Cartago, Costa Rica.

[8] McFarlane (1972). *"Digital Pictures Fifty Years Ago".*

[9] Smith, A. (1995). *"A Pixel Is Not A Little Square, A Pixel Is Not A Little Square, A Pixel Is Not A Little Square! (And a Voxel is Not a Little Cube)".* Microsoft Technical Memo.

[10] W3C, (2009). *"Scalable Vector Graphics".* XML Graphics for the Web [online].

[11] Microsoft Corporation, (2017). *"Windows Metafile Format".* Documentations at Developer Network.

[12] Marr, D., Hildreth, E. (1980). *"Theory of Edge Detection".* Proc. Royal Soc.London, vol. 207, 187-217.

[13] Rashmi, Mukesh, K., Saxena, R. (2004). *"Algorithm and technique on various edge detection: a survey".* Department of Electronics and Communication Engineering, SHIATS-Allahabad, UP.-India.

[14] Urrea, J., Ospina, E. (2013). *"Implementación de la transformada de Hough para la detección de líneas para un sistema de detección de bajo nivel".* Universidad Tecnológica de Pereira.

[15] Herrnstein, R., Murray, C. (1994). *"The Bell Curve: Intelligence and Class Structure in American Life".* The Free Press.

[16] García, G. (s.f.) *"Procesamiento Audiovisual: Espacios de Color y el Dominio Frecuencial.* Universidad de Murcia, Departamento de Informática y Sistemas.

[17] Gómez, R. (s.f.). *"White Paper: Lenguajes de Programación".* Revista Digital.

[18] Herrera, J., Medina, S. (2015). *"Diseño de un sistema automático de selección de frutos de café mediante técnicas de visión artificial".* Universidad Autónoma del Caribe. Facultad de Ingeniería, Ingeniería Mecatrónica, Barranquilla.

[19] Aguilar, K. (2018). *"Funciones de OpenCV en el diseño de un sistema de monitoreo para la determinación de la curva de vuelo de la broca en plantaciones de café mediante procesamiento digital de imágenes"*. Instituto Tecnológico de Costa Rica.

[20] Bhadane, G., Sharma, S., Nerkar, V. (2013). *"Early Pest Identification in Agricultural Crops using Image Processing Techniques"*. Department of Electronics and Communication, R.K.D.F. Institute of Science and Technology, Bhopal, (MP).

[21] Campbell, J. (2007). *"Introduction to Remote Sensing."* Fourth Edition, The Guildford Press.

[22] DigitalGlobe. (s.f.). *"Remote Sensing Technology Trends and Agriculture"*. White Paper.

[23] Leachtenauer, Jon C. (1996). *"National Imagery Interpretability Rating Scales Overview and Product Description."* ASPRS/ACSM Annual Convention & Exposition Technical Papers. Bethesda: ASPRS/ACSM, 1996. 1:262-272.

[24] Aero, P. (2018). *"What is Ground Sample Distance (GSD) and How Does it Affect Your Drone Data?"*. Electronic Magazine.

[25] García, E., Flego, F. (2012). *"Tecnología Agropecuaria: Agricultura de Precisión"*. Facultad de Ingeniería, Universidad de Palermo.

[26] Xia, Ch., Chon, T. (2014). *"Automatic Identification and Counting of Small Size Pests in Greenhouse Conditios with Low Computational Cost"*. The Research Center for Coastal Environmental Engineering and Technology of Shandong Province, Yantai Institute of Coastal Zone Research, Chinese Academy of Sciences, Yantai 264003, P.R. China.

[27] Miranda, J., Gerardo, B. (2014). *"Pest Detection and Extraction Using Image Processing Techniques"*. International Journal of Computer and Communication Engineering, Vol. 3, No. 3.

[28] Krishnan, M., Jabert, G. (2013). *"Pest Control in Agricultural Plantations Using Image Processing"*. M.E.Communication Systems (Electronics and Communication), Bannari Amman Institute of Technology.

[29] Guarnieri, A., Maini, S. y Molari, G. (2011). *"Automatic Trap for Moth Detection in Integrated Pest Management"*. Dipartimento di Economia e Ingegneria Agrarie, Università di Bologna, Italy.

[30] Herrera, J., Medina, S. (2015). *"Diseño de un Sistema Automático de Selección de Frutos de Café mediante Técnicas de Visión Artificial."* Universidad Autónoma del Caribe, Facultad de Ingeniería Mecatrónica, Barranquilla.

[31] Vijayakumar, J., Arumugam, S. (2014). *"Powdery Mildew Disease Identification in Karpoori Variety of Betel Vine Plants using Histogram Based Techniques."* ASP/ECE, Nandha College of Technology, Erode, Tamil nadu, India.

[32] Patil, J., Kumar, J. (2011). *"Advances in Image Processing for Detection of Plant Diseases."* Bharti Vidyapeeth C.O.E. Kolhapur, Bhatati Vidyapeeth (Deemed Univ.) Pune.

[33] Upton, E., Halfacree, G. (2014). *"Raspberry Pi: Guía del usuario"* 2da edición, Obtenido de https://raspberrypi.org.

[34] ITEAD Studio. (s.f.). *"Raspberry Pi GSM Add On"* Technical Support. Obtenido de https://raspberrypi.org.

[35] Dirks, B. (1999). *"Video for Linux Two"* Driver Writer's Guide. Obtenido de Linuxtv.org

[36] TicoStar Costa Rica. (2014). *"Bandas MHz de celulares en Costa Rica"* Obtenido de ticostar.com

[37] Spensa Digital Technologies. (s.f.). *"Sentinel: Automated and connected insect monitoring"*. Obtenido de spensatech.com/z-trap/

[38] Spensa Digital Technologies. (s.f.). *"Trapview: Automated Pest Monitoring System"*. Obtenido de: spensatech.com/sentinel/

[39] Arduino. *"Arduino Mega 2560"*. Obtenido de store.arduino.cc/usa/

[40] Orange Pi. *"Orange Pi Plus2"*. Obtenido de orangepi.org/orangepiplus2/

[41] Beagle Board. *"Beagle Bone Black Wireless"*. Obtenido de beagleboard.org/black-wireless

[42] Borenstein, G. (2012). *"Making Things See: 3D Vision with Kinect, Processing, Arduino, and MakerBot"*. MakerMedia, Sebastopol, CA.

[43] Raspberry Pi. *"Camera Module Documentation"*. Obtenido de raspberrypi.org.

[44] GitHub. *"Dropbox Uploader"*. Obtenido de github.com/andreafabrizi/Dropbox-Uploader.

[45] Dropbox. *"Elige el Dropbox más adecuado para ti"*. Obtenido de dropbox.com.

[46] Singh, S. (2016). *"Top 20 IoT Platforms in 2018"*. Obtenido de internetofthings.com.

[47] ThinkSpeak. *"The open IoT platform with MATLAB analytics."*. Obtenido de thingspeak.com.

[48] Adafruit. *"Lens Adjustment Tool for Raspberry Pi Camera."*. Obtenido de adafruit.com/product/3518.

[49] OpenCV. (2018). *"The OpenCV Reference Manual"*. Obtenido de opencv.org

[50] Sabogal, O., Hincapié, J. (2015). *"Modelos de Regresión Lineal para Estimación de Tiempos de Viaje en Sistemas de Transporte Masivo."*. Ciencia e Ingeniería Neogranadina, 25, pp.77-89.

7. Apéndices

7.1. Simulación de la primera etapa

Para la simulación de la primera etapa del sistema de monitoreo se diseñó una trampa elaborada en material acrílico transparente, la cual se muestra en la figura 7.1.

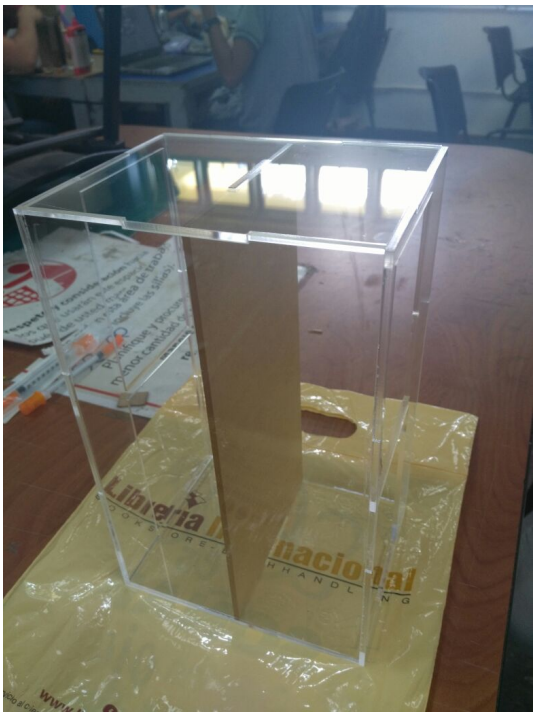


Figura 7.1: Diseño de trampa para simulación de la primera etapa.

Las figuras 7.2-7.3 muestran los resultados de colocar la trampa en el campo de siembra de café, para la cual se obtuvieron muestras reales del insecto.

Figura 7.2: Muestra de brocas para la trampa en condiciones reales (1).

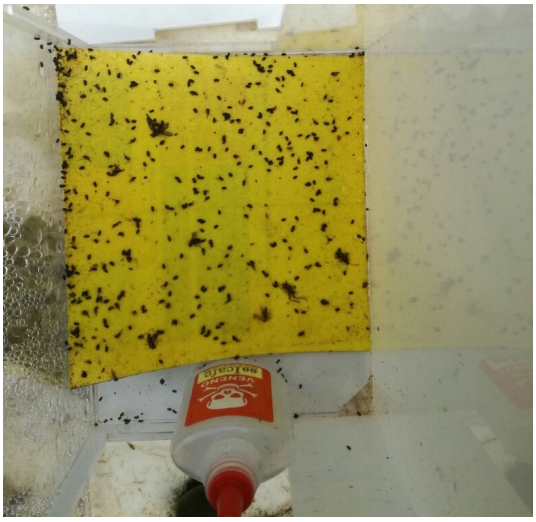


Figura 7.3: Muestra de brocas para la trampa en condiciones reales (2).

El sistema con la electrónica colocada utilizado para realizar todas las pruebas expuestas, se muestra en la figura 7.4.

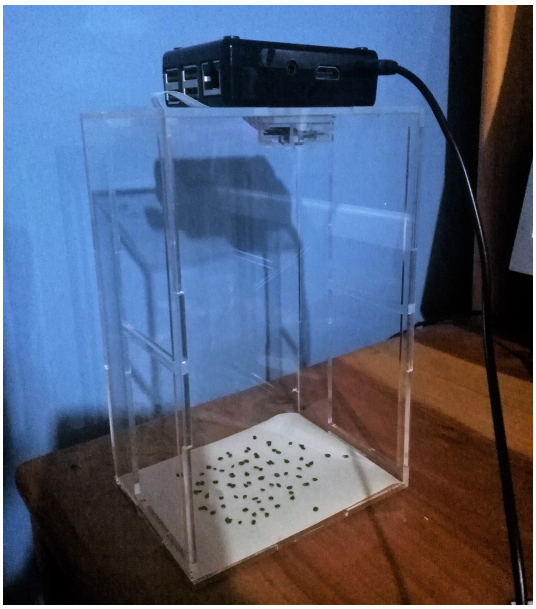


Figura 7.4: Sistema de monitoreo finalizado.