UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN MECATRÓNICA

INFORME FINAL DEL TRABAJO DE INTEGRACIÓN CURRICULAR, MODALIDAD DE PROYECTO DE INVESTIGACIÓN

TEMA:

"ALGORITMO PARA LA CLASIFICACIÓN DE AGUACATES TIPO FUERTE SEGÚN EL ESTADO DE MADUREZ MEDIANTE VISIÓN ARTIFICIAL"

Trabajo de titulación previo a la obtención del título de: *Ingeniero en Mecatrónica*

Línea de investigación: *Prototipos Industriales*

Autor: *Sayri Santiago Yamberla De La Torre*

Director: *Ing. Marco Remigio Pusdá Chulde. MSc*

Asesor: *Ing. Luz María Tobar Subía. MSc*

Ibarra - 2023

IDENTIFICACIÓN DE LA OBRA

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

DATOS DE CONTACTO			
CEDULA DE IDENTIDAD:	1004200786		
APELLIDOS Y NOMBRES:	Yamberla De La Torre Sayri Santiago		
DIRECCIÓN:	San Roque		
EMAIL:	ssyamberlad@utn.edu.ec		
TELÉFONO FIJO:		**TELÉFONO MÓVIL:**	0985610857

DATOS DE LA OBRA	
TÍTULO:	"ALGORITMO PARA LA CLASIFICACIÓN DE AGUACATES TIPO FUERTE SEGÚN EL ESTADO DE MADUREZ MEDIANTE VISIÓN ARTIFICIAL"
AUTOR:	Yamberla De La Torre Sayri Santiago
FECHA:	08 /06 /2023
SOLO PARA TRABAJOS DE TITULACIÓN	
PROGRAMA:	■ PREGRADO POSGRADO
TÍTULO POR EL QUE OPTA:	Ingeniero en Mecatrónica
ASESOR /DIRECTOR:	MSc. Marco Remigio Pusdá Chulde

CONSTANCIAS

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

Ibarra, a los 08 días del mes de junio de 2023.

El AUTOR

Firma: .....

Yamberla De La Torre Sayri Santiago

CERTIFICACIÓN DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

Ibarra, 06 de junio de 2023

MSc. Marco Remigio Pusdá Chulde

DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

CERTIFICA:

Haber revisado el presente informe final del trabajo de titulación, el mismo que se ajusta a las normas vigentes de la Unidad Académica de la Universidad Técnica del Norte; en consecuencia, autorizo su presentación para los fines legales pertinentes.

.....

MSc. Marco Remigio Pusdá Chulde

C.C.: 0401200925-1

APROBACIÓN DEL COMITÉ CALIFICADOR

El Tribunal Examinador del trabajo de titulación "ALGORITMO PARA LA CLASIFICACIÓN DE AGUACATES TIPO FUERTE SEGÚN EL ESTADO DE MADUREZ MEDIANTE VISIÓN ARTIFICIAL", previo a la obtención del título del INGENIERO EN MECATRÓNICA, aprueba el presente informe de investigación en nombre de la Universidad Técnica del Norte:

.....

Ing. Marco Remigio Pusdá Chulde, MSc
C.C.: 040120095-1

.....

Ing. Luz María Tobar Subía, MSc
C.C.: 1002444204

AGRADECIMIENTO

Primeramente, Agradezco a Dios por ser el dador de todo lo que tengo hasta ahora, a mi madre, la luz de mi camino, por ser la columna inquebrantable de la familia, sin mi madre Luz no podría estar donde estoy, la agradezco por ser padre y madre, que me haya inculcado valores y a no rendirme jamás.

Agradezco a cada uno de mis hermanos José, Carlos, Alexandra, Mateo, Pakcha y Tupac, que me han dado el apoyo incondicional.

A mis amigos.

Sayri Santiago Yamberla De La Torre

DEDICATORIA

Dedicado para mi madre Luz, mis hermanos, Ñusta y Tania, con todo el corazón y el alma, pues sin ellos no habría llegado tan lejos. Le dedico con todo el amor del mundo madre mía, la amo.

RESUMEN

En la zona de Guayllabamba, la producción y el comercio de aguacates es la mayor fuente de ingreso económico. La variedad de aguacate tipo Fuerte, es la más apetecida por su sabor y textura, es demandada por los comensales en el arte culinario. Debido a esto ha existido una elevada comercialización de este producto. El proyecto tiene como objetivo mejorar la calidad del punto específico requerido de madurez comercial del aguacate, mediante el desarrollo de un algoritmo de programación y Visión Artificial con la ayuda del software libre anaconda con el lenguaje Python. Utilizando la tecnología de visión artificial, las bibliotecas y módulos de Python en donde se desarrolla un algoritmo de clasificación de segmentación por colores. Además, se realiza una interfaz gráfica amigable e intuitiva para mostrar los resultados. Con una cámara de marca Logitech conectada por cable USB A, se hace la adquisición de imagen para la captura de video en tiempo real, prosiguiendo al preprocesamiento con métodos de filtrado y conversión del formato RGB a HSV para luego segmentar en rangos de colores. Finalmente, se obtiene los resultados en la pantalla de la interfaz gráfica, mediante puntos y textualizados con el nombre del tipo de madurez especificada para la clasificación.

Palabras clave: Aguacate tipo Fuerte, Textura, Calidad, Interfaz Gráfica, Segmentación por colores.

ABSTRACT

In the Guayllabamba area, the production and trade of avocados is the main source of income. The Fuerte type avocado variety is the most desired for its flavor and texture, it is demanded by diners in the culinary arts. Due to this, there has been a high commercialization of this product. This present project aims to improve the quality of the required specific point of commercial maturity of the avocado, through the development of a programming algorithm and Computer Vision with the help of the free programming software anaconda and the programming language Python. Using artificial vision technology, Python libraries and modules where a color segmentation classification algorithm is developed, a friendly and intuitive graphical interface is also made to display the results. With a Logitech brand camera connected by USB A cable, image acquisition is done for video capture in real time, continuing with pre-processing with filtering methods and conversion from RGB to HSV format to later segment it into color ranges. Finally, the results are obtained on the screen of the graphical interface, through points and textualized with the name of the type of maturity specified for the classification.

Keywords: Avocado type Strong, Texture, Quality, Graphical Interface, Segmentation by colors.

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Planteamiento del problema

En la zona norte de Pichincha han optado en reformar la agricultura, creando huertos donde se cultivan el aguacate de variedad Fuerte que tiene más acogida local e internacional, esto debido a que los cultivos tradicionales de maíz, tomate de riñón tenían precios inestables y baja economía. Los sembríos más significativos son de aguacate de diferente tipo y otros productos como la naranja, mandarina, durazno, entre otros [1].

Para la zona de Guayllabamba el aguacate es una de sus mayores fuentes económicas. El exquisito sabor y contextura de la variedad de aguacate Fuerte que ha deleitado el paladar de varios comensales ha producido una alta demanda y ha elevado la comercialización del producto [2].

Es por ello, que uno de los problemas identificados, es la falta de calidad del aguacate en estado de madures, este problema es consecuente a los errores del personal calificado durante el proceso de clasificación y al cansancio visual de los mismos. Como resultados se tiene la devolución de aguacate, pérdida económica y perjudicando la imagen del comercializador según entrevista realizada al Sr. Juan Guatemal. Con el desarrollo del algoritmo de visión artificial se podrá identificar el estado de madurez del aguacate de variedad Fuerte, con el fin de garantizar una mejora en la comercialización del producto.

Objetivos

Objetivo general

- Desarrollar un algoritmo de visión artificial para el análisis del estado de madurez del aguacate de variedad Fuerte.

Objetivos específicos

- Determinar las características y requerimientos para la clasificación del aguacate Fuerte según su estado de madurez.
- Diseñar un algoritmo para la identificación del estado de madurez del aguate de variedad Fuerte.
- Implementar una interfaz gráfica que permita observar los datos obtenidos del algoritmo en tiempo real.
- Validar el funcionamiento del algoritmo.

Justificación

El Aguacate ha aumentado poco a poco su popularidad por su infinidad de utilidades en la salud y por ser muy degustado en el arte culinario [3]. Su alta demanda exige una excelente calidad de producto y un proceso rápido de clasificación. El éxito en la propagación de este fruto ha llevado al Instituto Nacional Autónomo de Investigaciones Agropecuarias (INIAP) junto al Ministerio de Agricultura y Ganadería (MAG) a desarrollar proyectos en implementación de cultivos de aguacates y guías para su correcto manejo [2]. Estos proyectos fomentan a futuro, la producción de aguacate de calidad para una mayor comercialización. La implementación de este proyecto contribuirá a la rapidez y eficacia en la clasificación de aguacate, así como, la automatización del proceso manual para disminuir la sobrecarga laboral en los trabajadores.

Alcance

La finalidad del proyecto es la creación de un algoritmo de visión artificial, capaz de sintetizar los criterios de madurez del aguacate de la variedad Fuerte. Para observar los datos obtenidos se implementa una interfaz gráfica y se valida el funcionamiento del algoritmo llevando a cabo sus respectivas pruebas de campo.

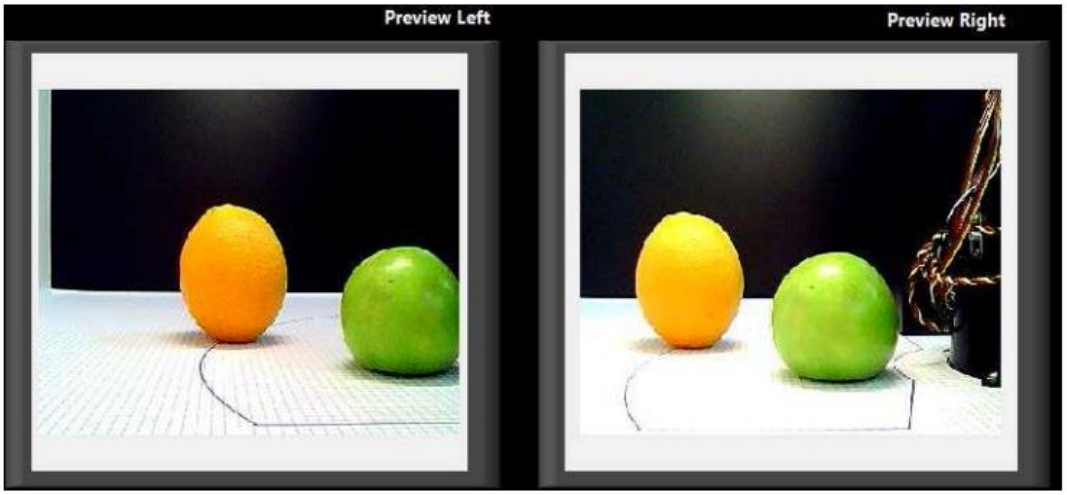
CAPITULO I: MARCO REFERENCIAL

1.1. Antecedentes

En la Universidad Técnica del Norte se realiza la elaboración de un algoritmo en base a una secuencia. Adquisición de imagen y preprocesamiento: mediante el procesamiento de imagen se mejora la calidad del aguacate, cuya imagen se obtiene previamente. El análisis de tamaño en píxeles permite calcular el área del fruto y estimar su tamaño. Los defectos del aguacate influyen en su calidad y se detectan con diferentes análisis de color: RGB, Lab2 y color en defectos. Segmentación: Para clasificar los aguacates según su tamaño y calidad se establecen rangos de clasificación. Y por último se realiza la validación del algoritmo [4].

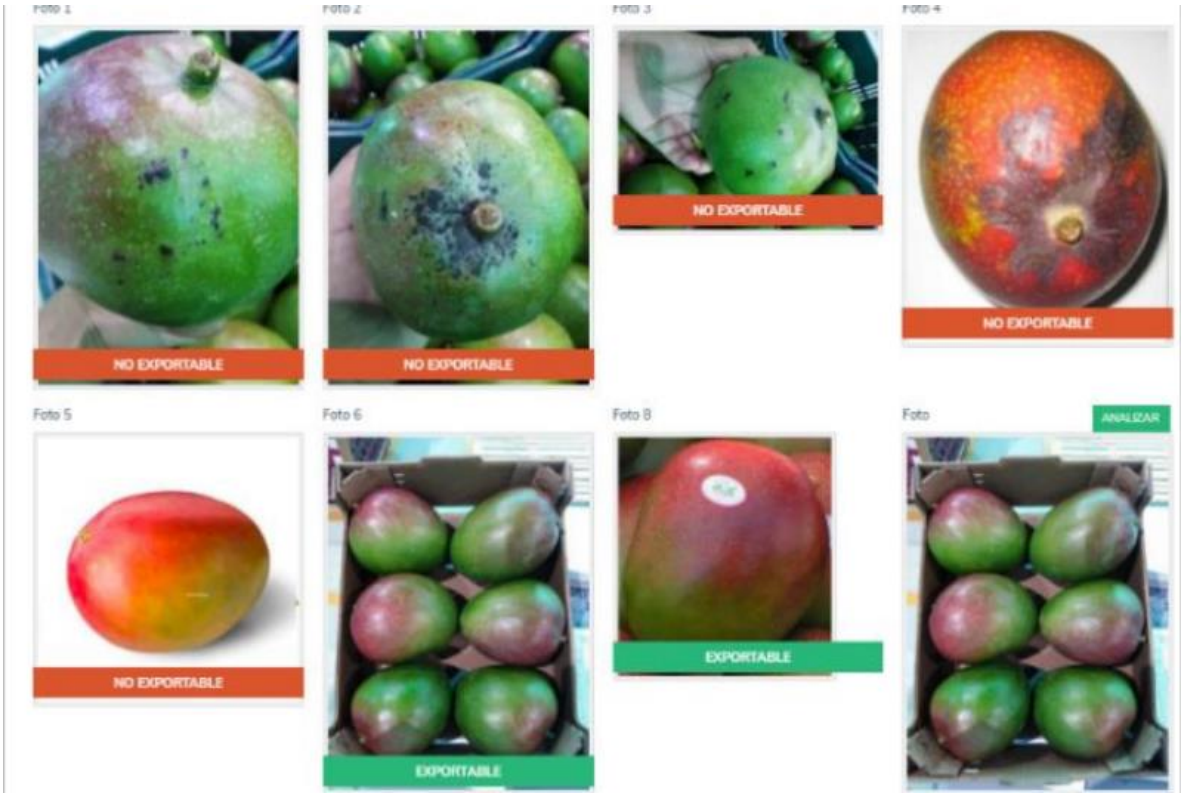
En la investigación realizada [5], se plantea un sistema de visión estereoscópica para clasificar manzanas, limones, naranjas, mandarinas y tomates. Diferencian estas frutas utilizando descriptores propios de la fruta como textura y forma como se ve en la Figura 1. El análisis de color lo realizan con el modelo de color azul, saturación, intensidad (HSI) precisamente los componentes de blue y saturación Con la matriz de coocurrencia de los niveles de nivel de gris (GLCM) extraen información como contraste, homogeneidad, correlación energía y entropía. El sistema reconoce estos patrones mediante una red neuronal. Evaluaron el funcionamiento del algoritmo en 45 frutas de cada denominación, obteniendo un porcentaje de aciertos del 90,2%.

Figura 1 *Clasificación de frutas por su tipo* [5].



En el mango de la variedad Tommy el parámetro primordial para su clasificación es el color, correspondiente a su estado de maduración como se puede apreciar en la Figura 2. En una selección exitosa predominan los colores verde, amarillo y rojo. Inician la adquisición de imagen en un sistema conformado por webcam, pc, banda transportadora y mecanismo de selección. La imagen en modelo red, green y blue (RGB) la segmentan con dilatación y erosión a escala de grises; para eliminar impurezas, obtener descriptores de área, caja, y recortar la imagen. En cada componente de color calculan el promedio de pixeles en porcentaje, este valor junto al área la incluyen en una red de decisión. El resultado de clasificación lo muestran en un cuadro de mensaje con el número de mango, decisión (rechazado o aceptado) y color predominante [6].

Figura 2 *Interfaz para la clasificación de mangos* [6].



En Bucaramanga juntamente con la Universidad Industrial de Santander se ha elaborado un sistema de visión por computador que clasifica aguacates de la variedad HASS tomando en cuenta el estado de maduración. Se propone un sistema basado en conexión Wireless de una Webcam con un ordenador el cual procesan las imágenes obtenidas y envía un mensaje al actuador con el objetivo de ubicar al fruto en el riel correspondiente de una banda transportadora. Antes de la segmentación, para reducir errores, a la imagen adquirida se aplica filtros de Wiener. La separación del fruto del fondo lo realizan mediante el análisis discriminante de Fisher. Obtenido el fruto clasifican en tres clases verde, maduro y muy maduro con el método de agrupamiento K-means. El desempeño del algoritmo es de un 87,85% de precisión [7].

1.2. Fundamentos teóricos

En este apartado se describe la información teórica necesaria para la construcción y diseño del algoritmo, datos de las herramientas de programación pertinentes para este proyecto de visión artificial y los fundamentos de la naturaleza botánica del aguacate.

1.2.1. Aguacate de la variedad fuerte

El origen de este aguacate tuvo lugar en el este y lugares altos del estado mexicano y sitios altos de Guatemala. Este producto fue distribuido e introducido en países de europeos y latinoamericanos por los españoles durante la época colonial. Debido a que buscaban la propagación de la planta de aguacate, inmediatamente buscaron nuevas técnicas de cultivos para que el aguacate soporte diferentes escenarios climáticos. Una de las técnicas más eficaces fue el injerto [8].

El aguacate Fuerte de la familia Persea americana (var. drimifolia) se originó en México, es una especie híbrida entre la raza mexicana y Guatemala. El factor más importante para una producción de calidad de esta especie es el clima y la altitud que está entre los 1500-2500 metros sobre el nivel del mar (msnm), siendo así Guayllabamba la localidad perfecta para su cultivo, ya que está a 1.620msnm [9], [10], [11].

1.2.2. Descripción botánica.

Es un fruto de forma aperada, piel lisa, el color varía dependiendo en el estado de madurez en el que se encuentre, varía desde verde claro hasta verde oscuro, pierde su brillo conforme se acerca a su estado de madurez, su masa está entre los 170 g - 500 g. Se lo puede observar en la Figura 3.

Figura 3 *Aguacate de la variedad Fuerte* [10].

1.2.3. Características.

En este apartado se describen sus características esenciales, esta variedad de aguacate se caracteriza por las siguientes particularidades [10]:

- Forma piriforme
- Semilla de tamaño mediano
- Es de color verde dependiendo su estado de madurez
- Tiene resistencia al transporte
- Buen tamaño

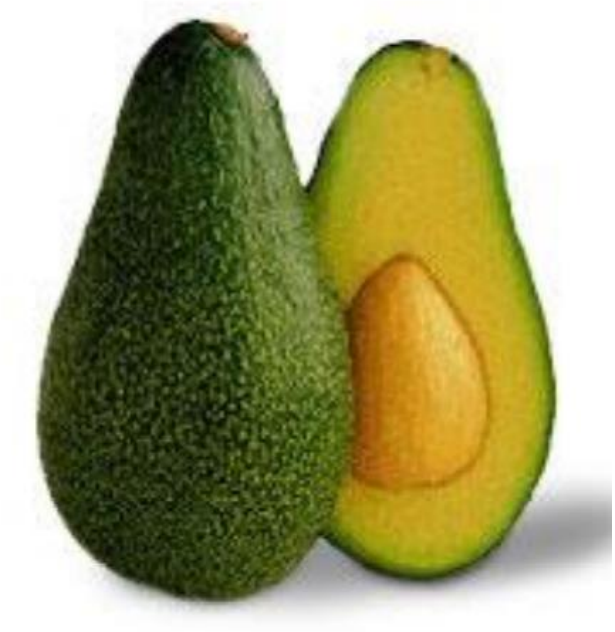
1.2.4. Estados de madurez del aguacate de la variedad Fuerte.

Según las normas INEN 1755 se clasifican en los siguientes estados [12]:

Madures fisiológica: estado en el cual el aguacate puede ser cosechado para seguir con el proceso de maduración, Figura 4.

Madurez comercial: estado en el cual el aguacate a alcanzado los requerimientos para su consumo.

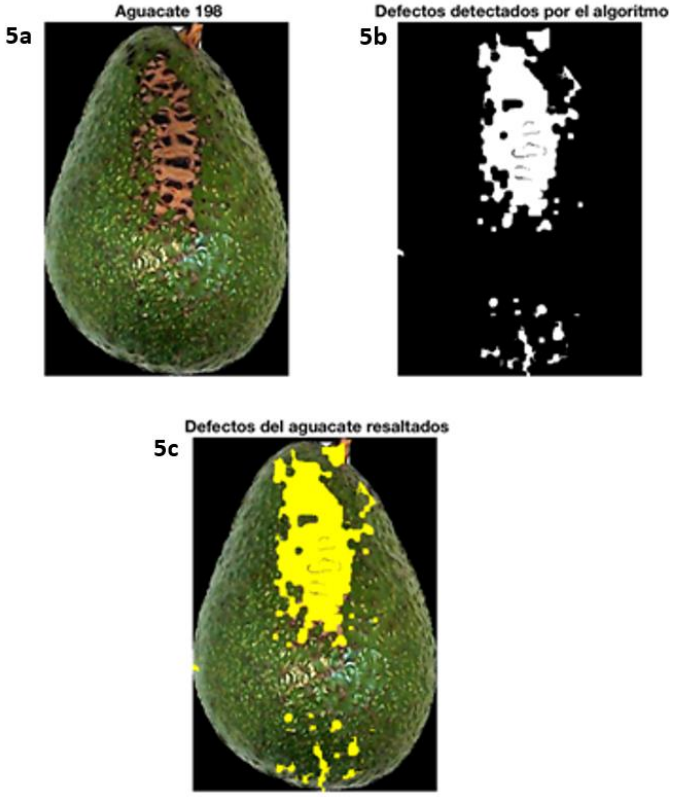
Figura 4 *Aguacate de la variedad fuerte en estado de madures fisiológica* [12].



1.3. Agricultura de precisión

La agricultura de precisión es una de las mejores opciones para optimizar los procesos de producción. Es así como al pasar el tiempo la inteligencia artificial ha tomado riendas en el sector industrial, la visión artificial se ha introducido en la agricultura, para la clasificación de frutos, la detección de enfermedades como se puede observar en la Figura 5. Además de la visión artificial existen diferentes campos de la inteligencia artificial que permiten mejorar procesos en diferentes ámbitos industriales [13].

Figura 5 *Detección de enfermedad roña en un aguacate Hass con visión artificial* [14].



Nota: En la Figura 5b se encuentra la imagen de aguacate original con lesiones de aspecto roñoso y corchoso de color café, en la Figura 5b se muestra la imagen binarisada de la lesión del aguacate, en la Figura 5c se indica la imagen original sobreexpuesta la imagen binarisada con resalto de color amarillo.

1.3.1. Aplicaciones de la agricultura de precisión.

Los sistemas de agricultura de precisión tienen muchas ventajas y facilidades, ya que ofrecen resultados más efectivos que los procesos manuales. La calidad de estos sistemas depende directamente de los sensores, actuadores y otros componentes con los que se han construido [15].

Las aplicaciones de la agricultura de precisión son:

- Detección de enfermedades en hojas y frutos.
- Clasificación por tamaño
- Clasificación por madurez
- Clasificación por especie

- Conteo de frutos.

1.3.2. Ventajas y desventajas de la agricultura de precisión.

Las ventajas de la agricultura de precisión son [13]:

- Aumento de producción
- Abaratar costos en recursos
- Optimizar esfuerzos laborales
- Optimizar el tiempo
- Aumento de calidad de productos
- Evitar desperdicios de recursos y productos

1.4. Visión artificial

Se define a la visión artificial como un proceso óptico de adquisición de información de un entorno, realizado por un sistema inteligente, para una posterior interpretación y análisis de imagen capturado mediante el uso de un ordenador [14].

La visión artificial se usa en diferentes disciplinas, ya sea en el campo de la industria, en la medicina, en la agricultura, entre otros. la V.A también se lo puede aplicar en la agricultura, por ejemplo, la identificación de enfermedades en plantas, aún que, por las condiciones de iluminación y la heterogeneidad de objetos y colores en el entorno, se obtiene como resultado errores en la validación [16].

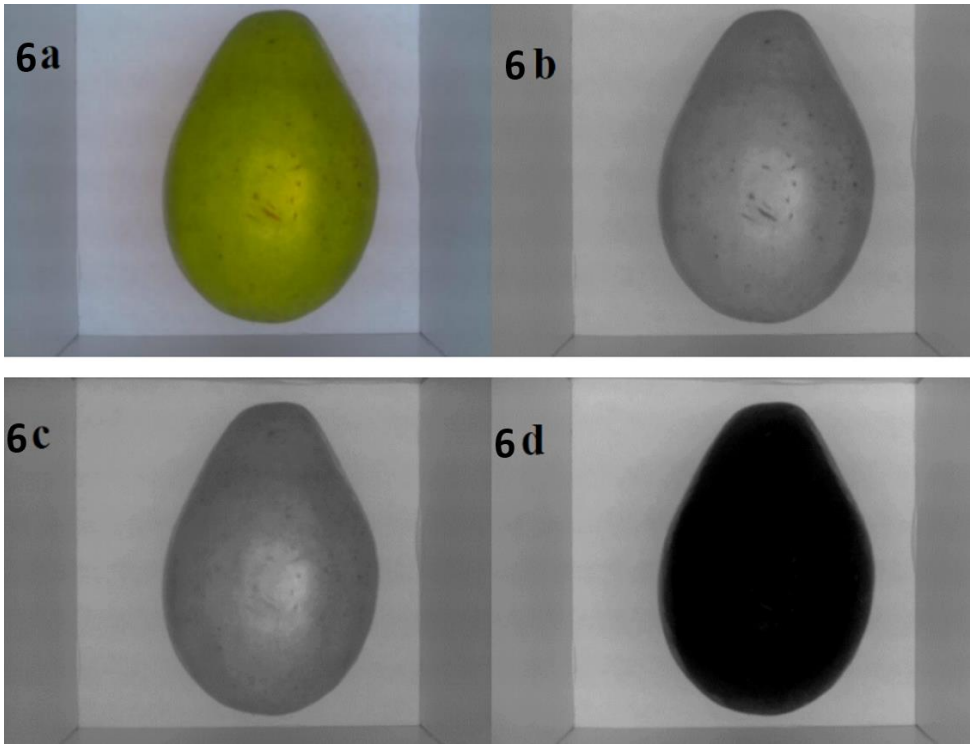
1.4.1. Colores en visión artificial

Los colores que percibe la cámara al igual que el ojo humano son en RGB son más distintivos para la clasificar cualquier objeto a diferencia que clasificarlos por el brillo. Por lo tanto, los productores optan que algunas frutas sean clasificadas por colores, como se muestra en el

ejemplo de procesamiento de imagen de la Figura 6, ya que los consumidores relacionan la calidad de su producto con el color [13].

Para obtener una buena clasificación por colores es necesario tener en cuenta el factor iluminación y el tamaño de pixeles, ya que mientras más pixeles tenga la imagen entonces, se puede decir que, tiene más probabilidad de segmentar el color correctamente [17].

Figura 6 *Muestra de una imagen umbralizada en el rango Green* [18].



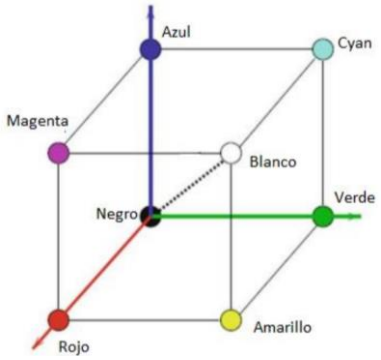
Nota: Se muestra la Figura 6a la imagen del aguacate Hass original, la Figura 6b la imagen en escala de grises, la Figura 6c la imagen en escala de grises con filtrado de imagen gaussiano, en la Figura 6d se observa la imagen con filtrado en totalidad y en escala de grises con un filtrado total en RGB de la imagen original.

1.4.2. RGB y HSV

Las cámaras funcionan con el modelo RGB para la recepción de imágenes, siendo así que en el sistema de coordenadas en el espacio RGB el rojo esté definido en el rango (255,0,0), el color

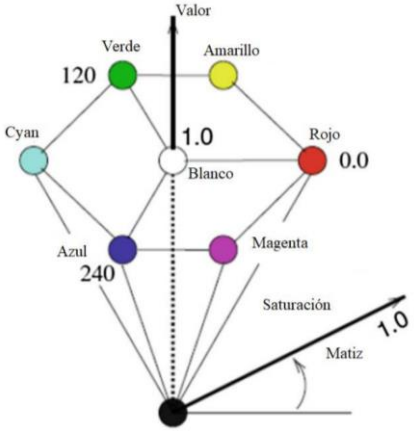
verde por (0,255,0), el azul en (0,0,0) y para colores mixtos (255,255,0) que está entre el color rojo y verde [19] como se observa en la Figura 7.

Figura 7 *Modelo en el espacio RGB* [19].



Sin embargo, la desventaja de usar este método en el espacio de color RGB mostrado en la Figura 9, es poder separar la información de color de brillo. Por la causa anteriormente descrita, por código se cambia al modelo HSV el cual ofrece una amplia gama de colores más amplia a diferencia del modelo RGB como se muestra en la Figura 8, y además es muy intuitivo para el usuario.

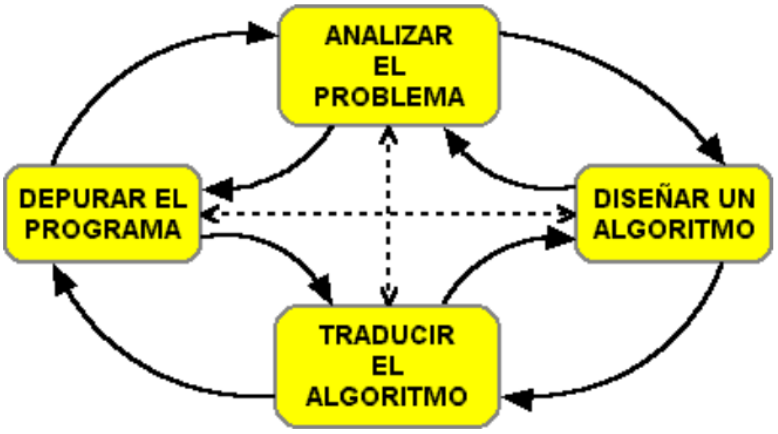
Figura 8 *Espacio del modelo HSV* [20].



1.5.Algoritmo en programación

En informática, los algoritmos son herramientas que permiten describir una secuencia de funciones sin ambigüedad a partir de un orden como se muestra en la Figura 9, que debe llevar a cabo un ordenador con el propósito de obtener un resultado previsto. Obligatoriamente la secuencia de instrucciones de un programa tiene que ser preciso y escrita en un lenguaje entendible para el computador [21].

Figura 9 *Etapas para desarrollar un algoritmo* [22].



1.5.1. Características de los algoritmos

Programar un algoritmo es construir un modelo de proceso con el fin de automatizar un proceso de la vida real por medio de un computador para utilizarlo cuantas veces nos sea útil. Los algoritmos poseen 5 características [23]:

Finitud: tener un número finito de pasos y tener una condición para terminar el programa, aunque existen casos especiales que es necesario tener corriendo el programa.

Proceso definido: estar definido precisamente, libre de ambigüedades.

Datos de entrada: estos datos tienen que servir para el proceso de resultado final.

Información de salida: los resultados del programa.

Efectividad: funcionamiento correcto para lo cual fue diseñado.

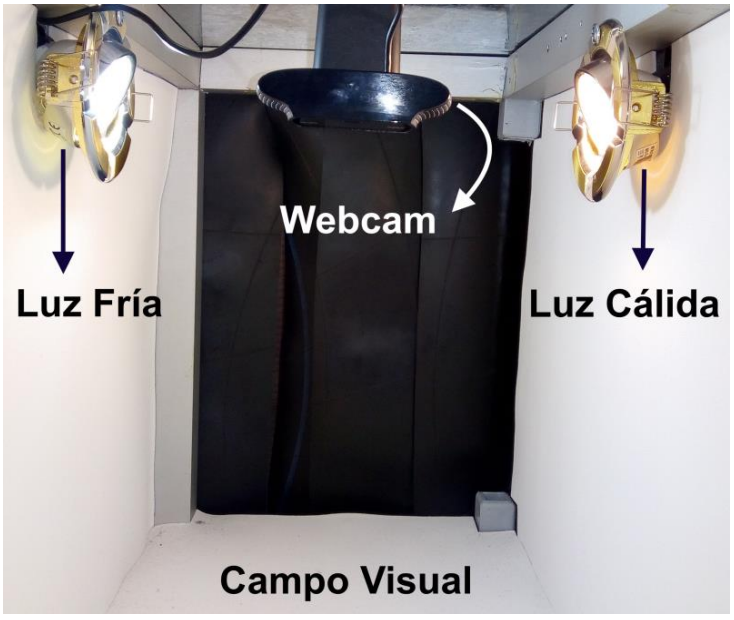
1.6.Etapas de un sistema de visión artificial

En el proceso de realización de un algoritmo de visión artificial, es importante segmentar y entender todas las etapas que lo componen.

1.6.1. Adquisición de imágenes.

Esta etapa es el inicio y a más primordial para que el algoritmo tenga un buen funcionamiento en las etapas siguientes, para una buena adquisición de imagen se tiene factores que se deben configurar bajo parámetros de iluminación controlada como se ve en la Figura 10, el interés de estudio y el objeto de estudio que se quiere segmentar [24].

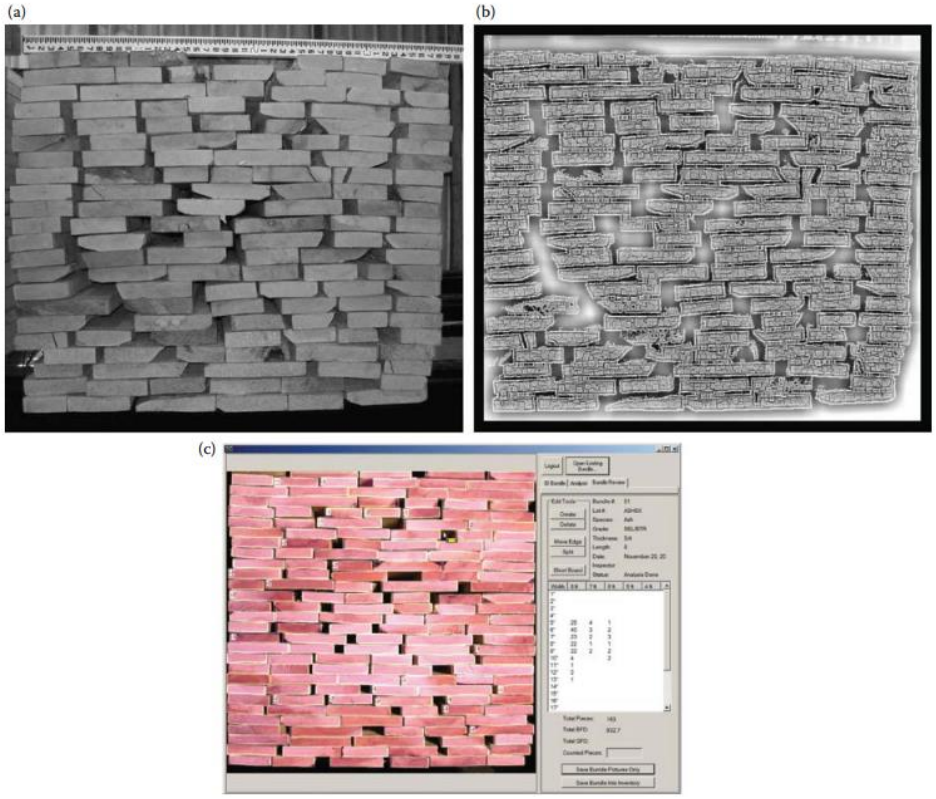
Figura 10 *Ambiente con iluminación controlada* [14].



1.6.2. Procesamiento de imágenes.

Es el análisis y procesamiento de imágenes digitales presentado en la Figura 11, que tienen como objetivo analizar la imagen y actuar sobre ellos, se obtiene las partes de la imagen sujetos a parametrización configurada de acuerdo con la funcionalidad que se pretende realizar, loe ejemplos más visibles en V.A son mejoramiento de imágenes, clasificación por colores, restaura, por tamaño, bordes, etc. [25].

Figura 11 *Imagen digital procesada* [25].



Nota: (a) imagen capturada por el sistema de V.A, (b) imagen durante la segmentación y preprocesamiento, (c) imagen procesada. Fotografía realizada por. Tony Berke, River City Software.

1.6.3. Umbralización.

La umbralización tiene como objetivo separar los pixeles requeridos de una imagen a partir de un rango de colores predispuesto después de la adquisición de imagen como se aprecia en la Figura 12, es uno de los procesos de segmentación fáciles y precisos en visión artificial [18].

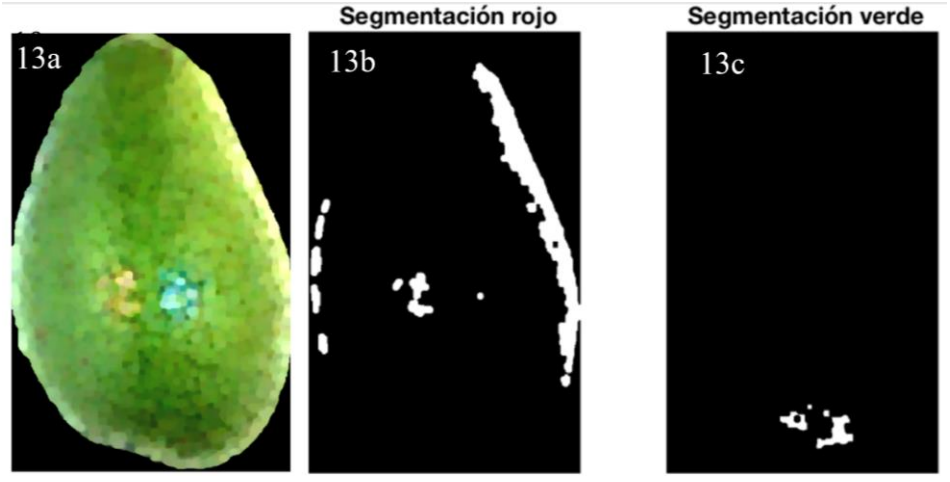
Figura 12 *Umbralización de un plátano en el rango HSV color amarillo* [18].

Nota: En la Figura 12a se muestra la imagen binarisada de la segmentada de un plátano en los rangos de color HSV amarillo, en la Figura 12b se encuentra la imagen original en RGB, en esta se puede apreciar el borde realizado en el área y perímetro de segmentación, así también un texto donde indica el área y perímetro de pixeles del objeto.

1.6.4. Segmentación.

La segmentación consiste in dividir la imagen en distintas partes que la componen como se aprecia en la Figura 13, luego se toma las partes que sean de interés en la imagen. Una forma para segmentar imagen es clasificar los pixeles de una imagen de la misma clase [26].

Figura 13 *Segmentación de una imagen* [27].



Nota: La Figura 13a es la captura de imagen de un aguacate de variedad Hass, en la Figura 13b se encuentra la segmentación del componente rojo que comprende al problema de iluminación, decoloración y lesiones mecánicas de

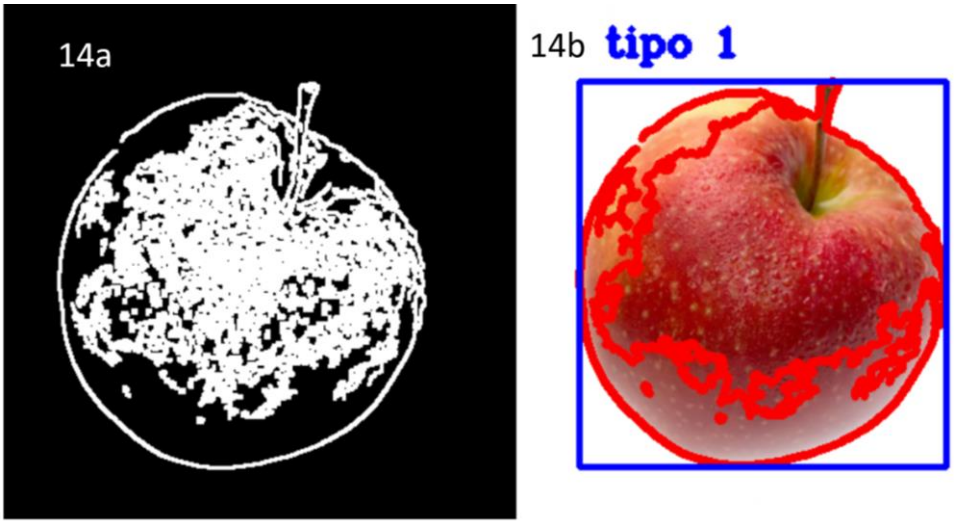
color café y la Figura 13c el componente verde que resalta las manchas de enfermedades y lesiones de colores opacos de la imagen del aguacate a color.

1.6.5. Bordes en Visión Artificial.

Los bordes en visión artificial son líneas que actúan como fronteras para diferenciar un objeto deseado del entorno situado después de la segmentación mostrada en la Figura 14, es así como se puede emplear en la clasificación por colores, por textura y por tamaño.

Contornos precisos en la segmentación dependerá de la calidad de la cámara como sensor de adquisición de imagen, de la iluminación, y de los filtros con los que sea tratada la imagen o video a segmentar.

Figura 14 S*egmentación por colores y detección por bordes.*



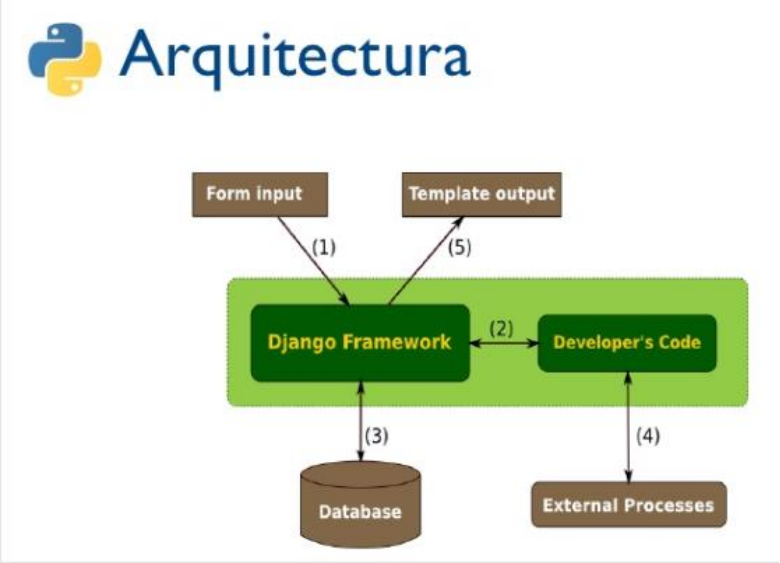
Nota: En la figura 14a se observa la imagen de una manzana aplicada la umbralización en el rango HSV rojo, y en la Figura 14 se muestra los bordes en los contornos de los pixeles umbralizados.

1.7. Python

Python en la actualidad es el lenguaje de programación de código abierto que incluye varias estructuras de datos, y una amplia cantidad de bibliotecas. Proporciona resultados extremadamente deseables y completos, a comparación de otros lenguajes, Python se destaca en un sin número de aplicaciones en diferentes áreas [28].

En la actualidad este lenguaje de programación se utiliza para realizar proyectos comerciales, educativos, de mininvestigación, machine learning, diseño de páginas web, entre otros. Al ser Python un reconocido lenguaje a nivel mundial los colaboradores siguen con actualizaciones con mejoras en las bibliotecas, módulos y demás extensiones que garanticen solucionar los problemas demandados por los usuarios de Python, Figura 15. Además de que Python brinda libros con información para fortalecer el aprendizaje de sus usuarios.

Figura 15 *Arquitectura de Python* [29].



1.7.1. Open CV.

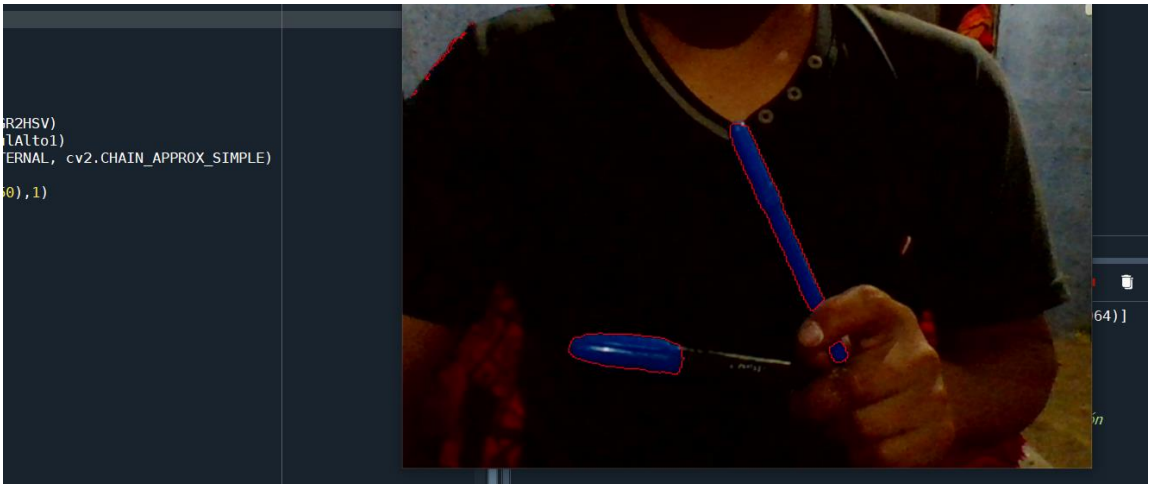
Es una biblioteca de visión artificial de código abierto, enfocada al procesamiento de imágenes, dentro de esta biblioteca existen cientos de algoritmos de V.A, dedicados al reconocimiento de objetos, proyectos como detectores de movimiento de objetos, filtrado de imágenes ente otros [30].

Debido su libre acceso y que los colabores han estado al tanto del crecimiento en tecnología y necesidad de nuevas opciones dentro de esta librería, este se ha dado un reconocimiento mundial,

siendo utilizada en proyectos científicos de investigación, en proyectos comerciales eficientes. Así también por su fácil instalación los diferentes en los sistemas operativos existentes hasta este entonces, como son: Android, Linux, Mac OS y Windows. Además, a la par de cada actualización proveen de manuales y libros, en donde el usuario podrá ver ejemplos de código abierto con los que podrá comprender cada actualización y fomentar su aprendizaje.

En la Figura 16, se puede observar un ejemplo de clasificación por colores realizado en OpenCV este algoritmo fue tomado de los libros de Python. Este algoritmo realiza la umbralización del color azul en este caso, se toma el área umbralizada en pixeles y se dibuja un borde que divide al objeto, poniéndolo en primer plano respecto del fondo.

Figura 16 *Detección de colores en el rango HSV (azul) realizado en OpenCV.*



1.7.2. NumPy.

Una librería de Python que se enfoca en el análisis y el cálculo numérico de datos, especialmente si son de gran cantidad, es NumPy. Esta librería introduce una nueva clase de objetos denominados arrays, que permiten representar conjuntos de datos homogéneos en varias dimensiones y operar con ellos de forma muy eficiente. Los arrays de NumPy tienen la ventaja de que se procesan mucho más rápido (hasta 50 veces más) que las listas predefinidas en Python, lo

que los hace ideales para trabajar con vectores como los mostrados en la Figura 17, y matrices de gran tamaño [31], [32].

Figura 17 *Creación de algoritmo con NumPy* [33].

```
2   import numpy as np
3   cap = cv2.VideoCapture(2)
4   azulBajo1 = np.array([100, 100, 20], np.uint8)
5   azulAlto1 = np.array([140, 255, 255], np.uint8)
6   # redBajo2=np.array([175, 100, 20], np.uint8)
7   # redAlto2=np.array([179, 255, 255], np.uint8)
```

1.7.3. Python Image Library

Es una librería de Python, está enfocado a la edición y exportación de imágenes. Soporta una amplia variedad de formatos de archivos, entre los utilizados en este proyecto están los siguientes: GIF, JPEG y PNG [34]. La imagen observada en la Figura 18, es una adquisición de imagen de video en tiempo real.

Figura 18 *Procesamiento de imagen de video en la interfaz gráfica de TKINTER.*

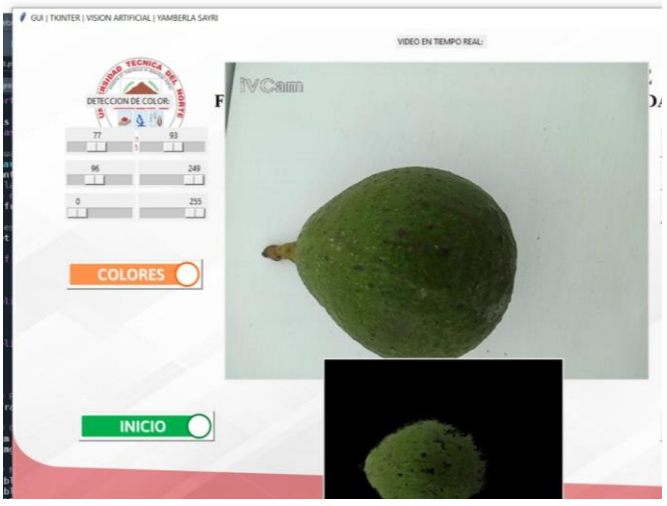
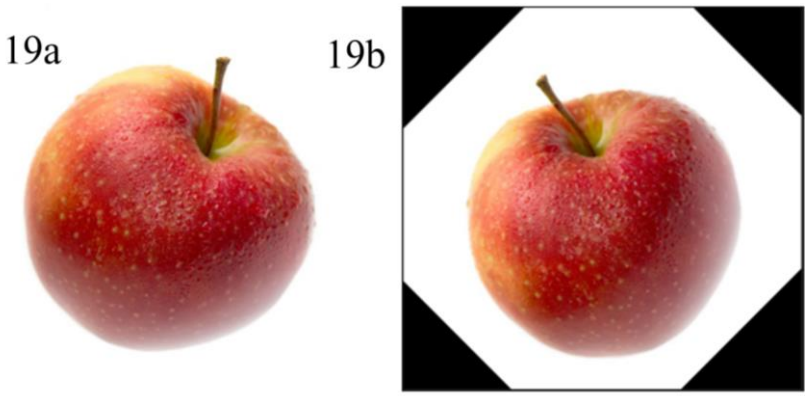


Image Tk: el módulo Image Tk permite crear y modificar objetos BitmapImage y PhotoImage de Tkinter a partir de imágenes PIL como se muestra en la Figura 18 [35].

Image: este módulo proporciona funciones, para cargar archivos de PIL y a partir de estos crear nuevas imágenes de la misma clase como se puede observar en la Figura 19 [36].

Figura 19 *Giro de 45 grados con visor externo con el módulo Image.*



Nota: En la figura 19a se muestra la imagen original de una manzana, la Figura 19b es la imagen girada 45° con un visor externo.

1.7.4. Imutils

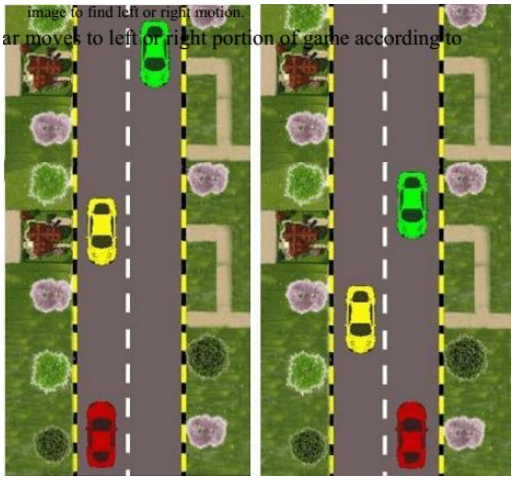
Es un paquete que se basa en OpenCV para facilitar el uso de su interfaz y permitir realizar de forma sencilla operaciones de procesamiento de imágenes como traslación, rotación, zoom. En la Figura 20 [37].

Figura 20 *Traducción de las coordenadas X e Y de la Figura 19a realizada en Imutils.*

1.7.5. Tkinter

Tkinter es la librería que ayuda en la interfaz gráfica de Python (GUI), es la interfaz estándar de Python y la base para otras bibliotecas [38]. Aquí se puede observar la captura de imagen en tiempo real, botones, textos y otros componentes dispuestos por códigos de programación, tal cual la interfaz de un juego en 2D mostrada en la Figura 21 [39].

Figura 21 *GUI utilizando Tkinter* [39].



1.8. Entorno controlado

Para optimizar la aplicación de un sistema de visión artificial, es recomendable garantizar el medio para obtener los mejores resultados como se ven en la Figura 10. A continuación, se describen ciertas características que garantizan el mejor resultado.

1.8.1. Iluminación.

Un sistema de visión artificial depende de varios factores, como son la cámara que se emplea en el proyecto, la distancia en la que está posicionada la cámara y la iluminación controlada [36]. La iluminación es una variable importante para la correcta adquisición de imagen, puesto que ayuda a la cámara a obtener una buena calidad de imagen y posteriormente extraer las más importantes características de la imagen a analizar [40].

1.8.2. Ventajas de la iluminación controlada

La iluminación controlada juega un papel importante en el análisis de datos en Visión Artificial, logrando obtener una eficiencia y minimización de los errores de adquisición de imagen como son: sombras, relieves, brillo [25].

1.8.3. Fuentes de iluminación.

El objetivo de elegir una buena fuente de iluminación es para optimizar la adquisición de la imagen por parte de la cámara y tener una solución del algoritmo de Visión Artificial para la cual fue diseñada, debido a que una mala iluminación afecta en la captura de la imagen y consecuente subiría el error en la validación [24].

Existen diferentes tipos de fuentes de iluminación, los cuales se muestran en la Tabla 1, a continuación:

Tabla 1 *Fuentes de iluminación* [41], [42].

Iluminación	Ventaja	Desventaja
Fluorescentes de alta frecuencia	Bajo precio. Adaptabilidad en forma y color.	No dan luz necesaria No son apropiadas para el proyecto de V. A.
Halógenas	Gran luminosidad	Excesivo calor Precio excesivo Rápido desgaste
Xenón	Mayor luminosidad	Precio alto
Led	Iluminación de calidad Duradera Baja potencia Varias configuraciones	Precio caro

	Disponibles en colores variados	
Láser	Se puede resaltar la tercera dimensión Dirección de iluminación modificable	Alto precio
Incandescentes	Económicas	No se usa como luz estroboscópica por ende negado para V. A.

1.8.4. Técnicas de iluminación.

La clave primordial para que un proyecto de visión artificial tenga un funcionamiento garantizado y preciso es un ambiente con iluminación controlada. Es por eso que se ha creado técnicas y estrategias que cumplan con las demandas de los proyectos de visión artificial, estas técnicas se realizan en sistemas cerrados con equipamiento de calidad, las técnicas existentes se detallan en la Tabla 2.

Tabla 2 *Técnicas de iluminación* [24].

Tipo de Iluminación	Aplicación	Ventajas	Desventajas
Frontal	Diferencia de colores y características. Recomendada para superficies con pocos reflejos.	Elimina sombras, la cámara se puede situar a distancias alejadas entre la cámara y el objeto.	Reflejos sobre superficies reflectantes
Lateral	En la adquisición de rayas, bordes y fisuras.	Resalta relieves hasta pequeñas y las define.	No recomendable en superficies que absorben Luz.

Campo Oscuro	Recomendada para resaltar incrustaciones o grabados en metal.	Buen contraste	No recomendable en superficies que absorben Luz.
Contraste	Recomendada para la adquisición de bordes del objeto.	Captar siluetas e impurezas	No permite reconocer los detalles superficiales.
Axial Difusa	Para superficies planas reflectantes.	Detección de códigos en materiales reflectantes.	No permite el reconocimiento en relieves.
Difusa Tipo Domo	Recomendada para superficies reflectantes.	Eliminación de sombras.	Costo elevado
Láser	Control de profundidad de cortes y objetos.	No es irrelevante la iluminación del entorno.	Baja intensidad lumínica si se utiliza lentes cilíndricas.

CAPITULO II: MARCO METODOLÓGICO

En este capítulo se detalla la metodología que se aplicó para el desarrollo de diseño del algoritmo de visión artificial orientado a la clasificación en postcosecha del aguacate de variedad fuerte. Además, se describe el proceso que valida el cumplimiento de los objetivos específicos planteados en este documento.

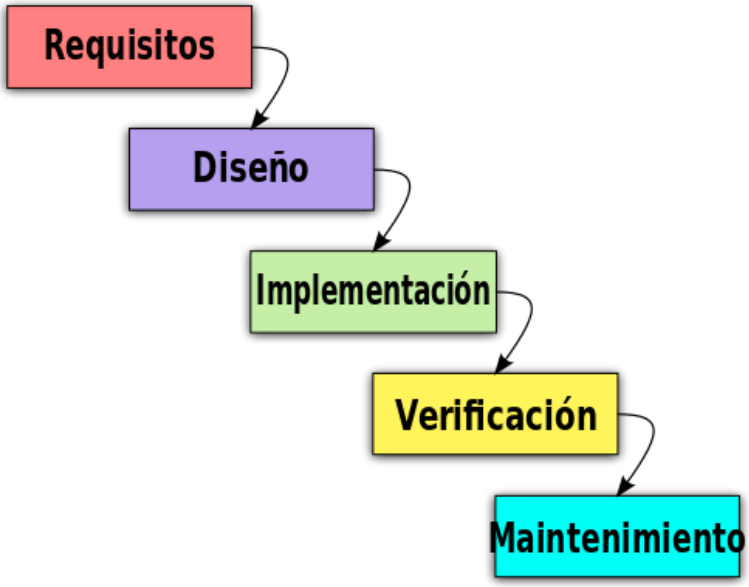
2.1. Modelo de la investigación

Para el desarrollo del presente trabajo de titulación se utilizan técnicas de investigación documental, debido a que se busca información como fuentes de investigación para profundizar el conocimiento y emplear información fundamentada en el diseño del algoritmo de este proyecto [42].

En el desarrollo de este proyecto, para la determinación de requerimientos de madurez de aguacate de variedad fuerte, se realiza la entrevista al usuario, para poder obtener información sobre los rangos de colores y brillo para lo cual el aguacate de variedad fuerte se considera maduro, en este punto se aplica la investigación de campo [43] y para la reseña de cada componente y los procesos se utiliza la investigación descriptiva [24].

Finalmente, para el diseño del algoritmo es necesario recurrir a la metodología tipo cascada como se muestra en la Figura 22, para poseer una orden secuencialmente clara de las etapas que deben ser cumplidas en este proyecto [44].

Figura 22 *Modelo de método cascada* [44].



2.2. Diseño de la investigación

En el diseño de investigación se estructura este proceso en cuatro fases con diversas actividades.

2.2.1. Fase 1: Requerimientos de beneficiario para el diseño del algoritmo.

Se define los requerimientos y características del estado de madurez del aguacate de variedad fuerte con que se va a fundamentar el diseño del algoritmo, esta información va a ser levanta por entrevistas hacia el beneficiario y un técnico en agronomía.

Actividad 1: Entrevista; se realiza la debida entrevista para recolección de información, al Sr. Juan Guatemal interesado de este proyecto y dueño de las plantas de aguacate de la quinta, que sirve como sustento para la solución del problema.

Actividad 2: Análisis de la entrevista; se analizan los requerimientos a los cuales debe estar sujeto el algoritmo.

Actividad 3: Selección de requerimientos; después del análisis de la entrevista, se determinan las variables que se utilizan para la solución del problema, en este caso de acuerdo con la entrevista realizada, se obtiene las siguientes especificaciones y parámetros para la clasificación del Aguacate de variedad Fuerte por estado de madurez.

Actividad 4: Planteamiento del diseño del algoritmo; se plantea realizar un algoritmo capaz de realizar la clasificación en tiempo real y tenga una funcionalidad adecuada sin ambigüedad.

2.2.2. Fase 2: Diseño y construcción del algoritmo.

En la segunda fase se describe el diseño y construcción del algoritmo con los parámetros seleccionados y propuestos en la Fase 1.

Actividad 1: Planteamiento de diseño; se propone el diseño del algoritmo acatando las especificaciones de usuario.

Actividad 2: Selección de librerías y componentes; se opta por las mejores opciones de librerías y módulos.

Actividad 3: Selección de hardware para adquisición de imagen; se selecciona la cámara con mejor adaptabilidad con el proyecto.

Actividad 4: Desarrollo del algoritmo; se desarrolla el algoritmo de visión artificial para la clasificación del producto, utilizando el método de segmentación.

2.2.3. Fase 3: Implementación de la interfaz gráfica.

Actividad 1: Diseño de la interfaz; se diseña la interfaz con especificaciones descritas por el usuario.

Actividad 2: Selección de librerías y componentes; se selecciona librerías que se adapten de mejor manera con Python.

Actividad 3: Desarrollo de la interfaz Gráfica; se desarrolla la interfaz gráfica con la propuesta de la persona que va a hacer uso del algoritmo.

2.2.4. Fase 4: Validar el funcionamiento del algoritmo.

Finalmente, en la Fase 4 se realiza las pruebas de funcionamiento de clasificación con los aguacates de variedad Fuerte en físico.

Actividad 1: Calibración del rango de colores en el formato HSV.

Actividad 2: Corrección de fallas.

Actividad 3: Presentación de interfaz; se hace la presentación del algoritmo con su respectiva interfaz gráfica y el funcionamiento de este.

Actividad 4: Pruebas de funcionamiento; se realiza las pruebas de campo con aguacates de la variedad Fuerte, en donde se observa el correcto funcionamiento.

CAPITULO III: RESULTADOS Y DISCUCIÓN

En este capítulo se detalla el proceso de construcción del algoritmo con su respectiva interfaz hasta cumplir con el objetivo planteado en este documento, y dar a conocer los pasos que se sigue para llegar a los resultados del algoritmo.

3.1. Requerimientos y especificaciones para el diseño del algoritmo

Las especificaciones de para el diseño es importante en este punto, mediante la aplicación de la entrevista al Sr. Juan Guatemal, dueño del sembrío de aguacates de la variedad Fuerte, se recolecta información como sustento para el desarrollo y diseño del algoritmo de visión artificial.

3.1.1. Requerimientos del usuario

Para determinar las especificaciones para el diseño del algoritmo, se realiza una entrevista al interesado en este proyecto, donde se toma los requerimientos relevantes que sugiere el usuario para el desarrollo de este proyecto mostrados en la Tabla 3.

Tabla 3 *Requerimientos de usuario.*

Requerimientos	Descripción
Precisión	Que sea preciso al clasificar los aguacates
Procesamiento	Que procese rápido al clasificar
Facilidad de uso	Que sea fácil de utilizar
Economía	Que se realice en programas de uso libre y material de construcción económico.

3.1.2. Requerimientos técnicos

Una vez escuchado los requerimientos del usuario, se da lugar a los requerimientos que debe tener el algoritmo y la interfaz gráfica, así como también las especificaciones de métodos de iluminación, tipo de cámara y tipo de iluminación que debe tener el prototipo para tener una validación y resultado de clasificación de eficaz y de calidad.

También se define el funcionamiento del algoritmo. Este permitirá la clasificación de aguacates de la variedad fuerte en dos estados, estado de madurez fisiológica y tierno, además tendrá una interfaz gráfica en donde estarán alojados los botones de INICIO y FINALIZAR, luego del inicializado del programa se muestra una pantalla con los resultados textualizados de la clasificación del aguacate.

Tabla 4 *Requerimientos del técnico.*

Requerimientos	Descripción
Procesamiento	Procesamiento rápido
Software de programación	Alto nivel y libre
Precisión	Precisión al clasificar aguacates
Luminosidad	Luminosidad correcta
Interfaz de usuario	Bebe tener una interfaz gráfica
Adquisición de video	Buena cámara
Material de construcción	Madera

3.2. Propuesta de requerimientos

Luego de especificar el funcionamiento del algoritmo se procede a dar las propuestas para la solución, tomando en cuenta el software, hardware y otros elementos existentes en el campo de visión artificial netamente relacionados a lo que son la clasificación de frutos.

3.2.1. Propuesta de requerimientos para la solución del problema

Una vez obtenido los requerimientos de usuario y técnico mostradas en la Tabla 5, es posible observar la alternativa de solución para cumplir con el objetivo propuesto.

Tabla 5 *Propuestas de requerimientos*

Nomenclatura	Requerimientos	Elementos que permite cumplir el requerimiento
R1	De programación de alto nivel	Lenguaje de programación Python
R2	Precisión	Algoritmo de alto nivel
R2	Procesamiento	Numpy y buen procesador.
R3	Adquisición de video	Cámara de video de buena calidad
R4	Luminosidad	Iluminación y ambiente controlado, aro de luces led.
R5	Interfaz de usuario	Tkinter
R6	Material de construcción	Madera

3.2.2. Análisis de los elementos que permiten cumplir el requerimiento

En la Tabla 6 se muestra las ventajas y desventajas de utilizar los componentes que permiten cumplir los requerimientos propuestos para la solución del problema.

Tabla 6 *Descripción de elementos.*

Elemento que permite cumplir el requerimiento	Ventajas	Desventajas

Lenguaje de programación Python	Lenguaje de alto nivel Gratis y de código abierto Módulos y bibliotecas integrados Portabilidad Comunidad fuerte	Consumo de memoria Lentitud
Cámara Samsung A12	Enfoque automático	Tamaño Conexión a PC
Aro luces leds	Ahorro de energía Mayo tiempo útil Dirección de la luz por su forma Cantidad de luz	Precio Temperatura elevada Complicación en la regulación de luz
Interfaz gráfica Tkinter	Preinstalado con Python en la plataforma OpenCV Flexibilidad de uso Documentación completa	Pocos elementos gráficos
Madera	Económico	Propiedades mecánicas bajas

Una vez realizada la entrevista se mencionó sobre las herramientas de programación, las librerías que encontramos para programar en Python, y los elementos físicos que con los cuales se van a solucionar el problema. Se procede a delimitar cuales son los elementos necesarios tomando en cuenta los requerimientos del usuario detallados previamente en la Tabla 3, y los requerimientos

técnicos necesarios para el diseño de un algoritmo de visión artificial que se mostrados previamente en la Tabla 4.

3.3. Propuesta de solución para el desarrollo del algoritmo: software

Realizada la respectiva entrevista y tras haber propuesto una solución a la problemática se procede a utilizar las siguientes herramientas de software de programación para el desarrollo del diseño del algoritmo.

3.3.1. Métodos para la clasificación.

En el campo de la visión artificial existen dos métodos enfocados a la clasificación de objetos por sus características, por su forma, por el tamaño y por el color. Estos métodos permiten diferenciar a un objeto en específico dentro de una escena.

Los métodos para la detección de características de un objeto dentro de una escena son: detección de objetos y tratamiento de imágenes. Dependiendo de la complejidad y costo para el funcionamiento del sistema y ejecución después del desarrollo del algoritmo, se eligió la clasificación por tratamiento de imágenes.

El método de detección de objetos es el más eficiente en la clasificación de objetos en visión artificial, pero debido al su costo del software para el diseño, ya que para este método se necesita un entrenamiento previo a la puesta en marcha del funcionamiento del algoritmo y consecuente a esto es necesario tener un procesador de alta calidad se aplica el método de tratamiento de imágenes, teniendo como objetivo la segmentación.

3.3.2. Lenguaje de programación Python.

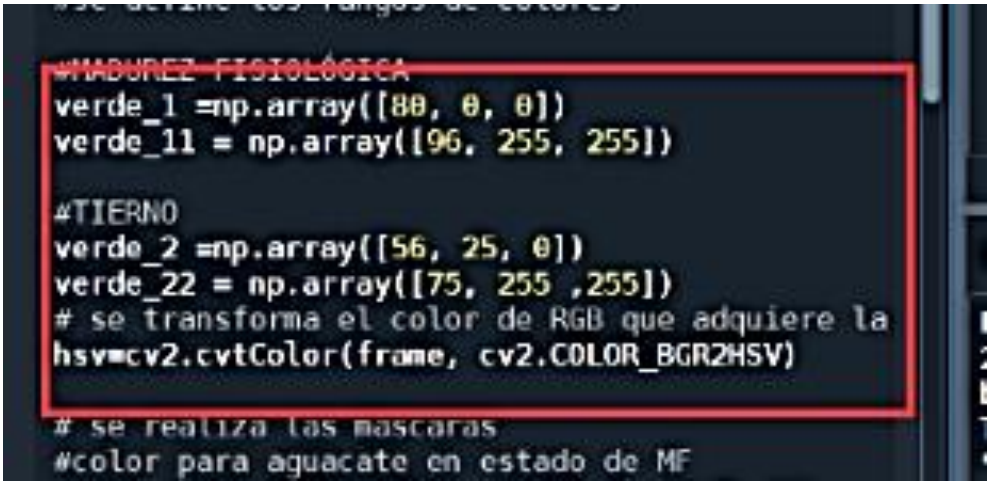
Se propuso los requerimientos mostrados previamente en la Tabla 5, que son los adecuados para desarrollar el diseño y validación del algoritmo de visión artificial. El lenguaje de programación seleccionado es Python, el usuario no tiene presupuesto para pagar un software y

este lenguaje al igual que sus entornos y bibliotecas están más que adaptadas a su estado económico.

Python tienen sinnúmero de librerías y entornos para el desarrollo de código abierto, estás están dentro del entorno de ANACONDA NAVIGATOR, su fácil uso hace que personas con conocimientos básicos puedan programar en estos entornos de programación.

NumPy asiste en el proceso de la información de la imagen de video en matrices, haciendo del algoritmo más rápido. Esta librería está enfocada para visión artificial, para el procesamiento de imagen, ya sea segmentación, modificación de tamaño de imagen, aumento de resolución, filtrado y mejoramiento de imágenes, en la Figura 23 se observa la creación de matrices verde_1, verde_11, verde_2, verde_22 en el rango HSV que servirá para la umbralización de una imagen.

Figura 23 *Creación de matrices en Python.*



El entorno de anaconda es un software multiplataforma, se eligió debido a su costo, a su eficiencia. En este entorno se puede iniciar un proyecto aplicando conocimientos básicos de programación en lenguaje Python.

3.4. Propuesta de solución para el desarrollo del algoritmo: hardware

Para estructurar la sección del hardware del algoritmo se presentan varias áreas y el hardware que cumple con los requerimientos.

3.4.1. Iluminación para el entorno controlado.

El entorno controlado es sin duda una de las variables muy importantes para el funcionamiento eficiente de un proyecto de visión artificial. Las luces led que se muestra en la Figura 24, aumenta la calidad de iluminación, evitando sombras de los mismos objetos a clasificar, facilitando a la adquisición de la imagen, con resaltes en el color y brillo.

Figura 24 *Aro luces led* [45]**.**

3.4.2. Procesador para el Desarrollo del Algoritmo

El ordenador mostrado en la Figura 25, fue empleado en el diseño y programación del algoritmo. Tiene las siguientes especificaciones de dispositivo y de Windows que se muestran en la Tabla 7:

Figura 25 *Laptop Lenovo Legion Y720.*

Tabla 7 *Especificaciones del dispositivo y sistema.*

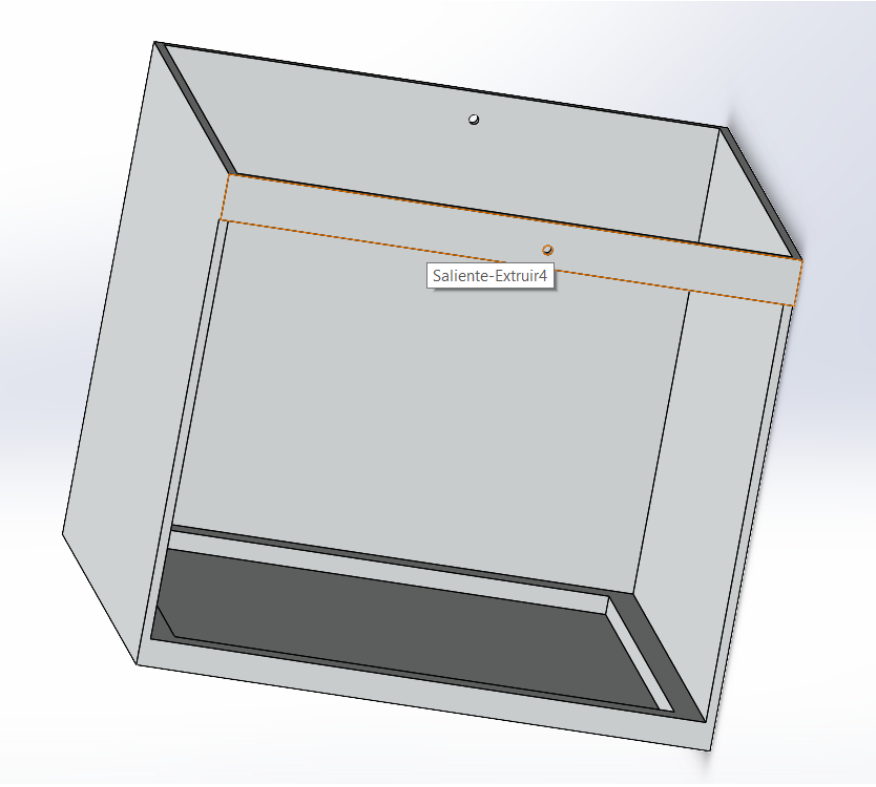
ESPECIFICACIONES DEL DISPOSITIVO (Y720-15lKB)	
Procesador	Intel(R) Core (TM) i7-7700HQ CPU @ 2.80GHz 2.80 GHz
RAM instalada	16.0 GB
Tipo de sistema	Sistema operativo de 64 bits, procesador x64
ESPECIFICACIONES WINDOWS	
Edición	Windows 10 Pro
Versión	21H12

3.4.3. Construcción del ambiente controlado para la verificación del algoritmo

Para la validación y el correcto funcionamiento del algoritmo es necesario tener entorno cerrado con iluminación controlada, además es indispensable una estructura en donde se pueda realizar el montaje de la cámara y la iluminación.

Las dimensiones de la estructura de ambiente controlado son de 27 cm de ancho x 27 cm de largo x 27 cm de altura. Una vez obtenidas estas dimensiones se procede a la construcción y diseño de las piezas en donde se colocará, la cámara y el aro de luces para la iluminación controlada como se muestra en la Figura 26.

Figura 26 *Estructura para el ambiente controlado.*

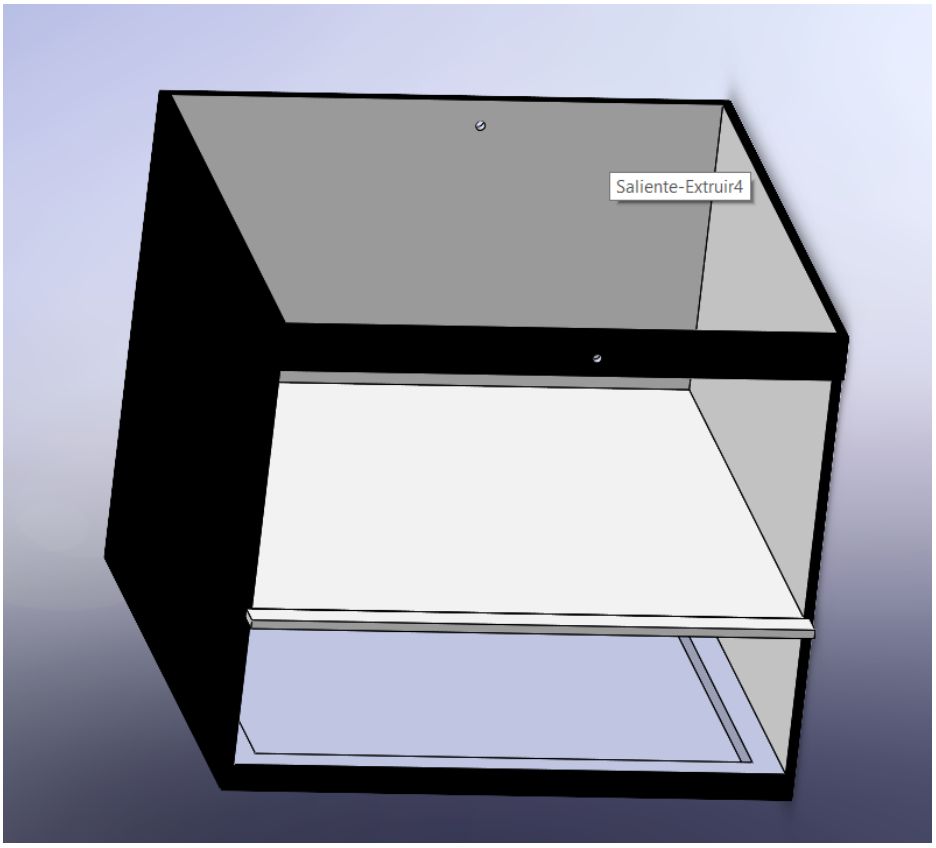


También se realiza los respectivos agujeros en los que se fijará el aro de luces, sobre este aro de luces estará fijada una tira de madera, ensamblada con pernos en el cual sujetar la cámara mediante pernos.

El material es lámina de madera pintada con un color blanco mate como fondo para evitar sombras en la adquisición de la imagen como se muestra en la Figura 28.

La colocación del producto a clasificar será situada en una superficie plana dentro del entorno cerrado, para eso se monta una lámina de madera como una especie de cama, se puede visualizar en la Figura 27.

Figura 27 *Ensamblado de cama.*



Por último, se hace el montaje de la cámara, en la estructura que viene en el paquete del aro de luces LED, como se observa en la Figura 28, con la cual se va a realizar la adquisición de datos.

Figura 28 *Prototipo ensamblado para validación del algoritmo.*

3.5. Funcionamiento

En la contextualización del funcionamiento del algoritmo es útil emplear herramientas que faciliten entender el proceso. El diagrama funcional de la Figura 29, describe el sistema general.

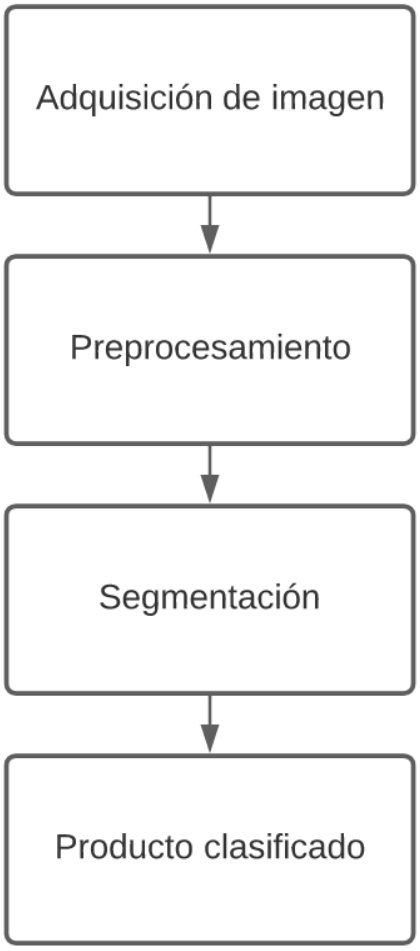
Figura 29 *Diagrama funcional del sistema general.*

Después de especificar la propuesta de requerimientos, se da el funcionamiento del algoritmo.

3.5.1. Funcionamiento del algoritmo.

Para la sección del algoritmo se emplea un flujograma para entender el software del algoritmo como el Figura 30.

Figura 30 *Diagrama de flujo del funcionamiento del algoritmo.*



En cuanto al funcionamiento del algoritmo, después de la adquisición de imagen se da un tratamiento y cambio de formato de RGB a HSV, esto aparece en la interfaz gráfica como un contorno y un punto etiquetado donde se muestra la clasificación del aguacate según el estado de madurez como se observa en la Figura 31.

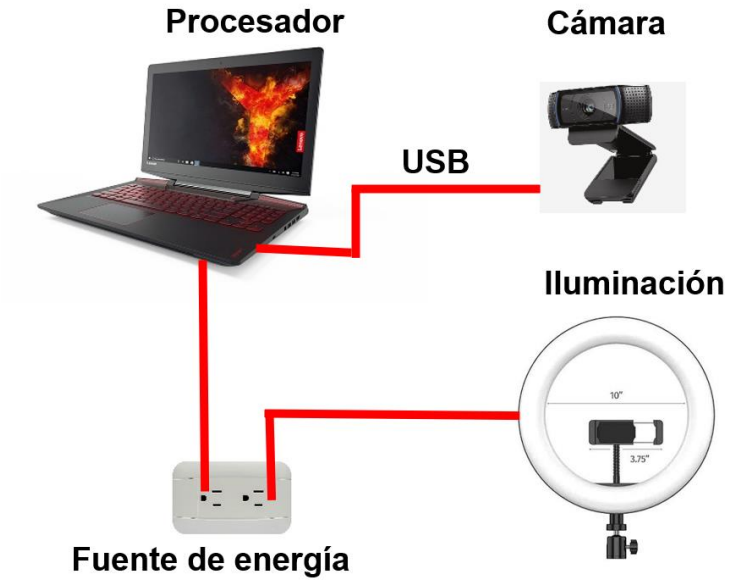
Figura 31 *Interfaz gráfica diseñada en Tkinter.*

3.5.2. Funcionamiento: hardware.

Según estudios anteriores [24],[25], [41 y [42], el entorno donde se realizan las pruebas es crucial para que el algoritmo resuelva adecuadamente el problema planteado. Por lo tanto, para construir el ambiente controlado se considera la información descrita en las secciones 1.4, 1.6 y 1.8.

El funcionamiento del ambiente controlado inicia con la adquisición de la imagen mediante la cámara Logitech situada en el centro del del aro luz considerando las técnicas descritas anteriormente en la sección 1.8, estos componentes están conectadas a una fuente de 12v DC. Una vez realizada la captura de video del objeto, envía está información al ordenador para el funcionamiento del algoritmo. En la Figura 32 se muestra el esquema de conexión de hardware.

Figura 32 *Diagrama de conexión de hardware.*



3.6. Desarrollo del algoritmo

En el desarrollo del algoritmo de visión artificial para obtener el funcionamiento presentado en el diagrama de la Figura 30, se crea varias funciones con instrucciones por bloques, con diferentes finalidades específicas para utilizar a lo largo del código y poder emplear estás funciones como variables globales para repetir tareas dentro de otras funciones, las funciones creadas son las siguientes:

1. **def iniciar():**
2. **def visualizar():**
3. **def finalizar():**

Adquisición de la imagen del aguacate

La adquisición de la imagen de video en tiempo real se inicia en el bloque de funciones **def iniciar():** con la línea de código **cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)** que es un comando de funciones en Open CV.

Para la adquisición de la imagen es relevante y de suma importancia un escenario en condiciones óptimas es por esta razón que se realiza la construcción un prototipo para el control de la iluminación, donde al enviar a correr el algoritmo cumpla con las funciones para la clasificación del aguacate y cumpla como solución para el problema planteado.

En la construcción del prototipo, se toma en consideración las técnicas de iluminación existentes [24] y se toma la iluminación tipo frontal como se muestra en la Figura 33, este tipo de iluminación controlada elimina sombras de la imagen y una buena iluminación, dando como resultado una imagen de calidad, con lo que se logra una buena adquisición de imagen para para segmentación por colores.

Figura 33 *Iluminación frontal* [24].

La implementación y la selección de la cámara mostrada en la Figura 34, se hizo realizando comprobaciones y tomando en cuenta las características necesarias para una adquisición de calidad de imagen en tiempo real, las características de la cámara se describen en la Tabla 8.

Figura 34 *Cámara Logitech* [46].

Tabla 8 *Características cámara Logitech* [46].

Especificaciones Técnicas	
Resolución máxima	1080p/30fps – 720p/30fps
Enfoque	Automático
Tipo de lente	Cristal
Angulo de visión	78°
Apertura	f/2.0
Dimensiones	43.3 mm x 94 mm x 71 mm
Conexión	Puerto USB - A
Requisitos del Sistema	
Compatibilidad	Con versiones de Windows posteriores a Windows 8

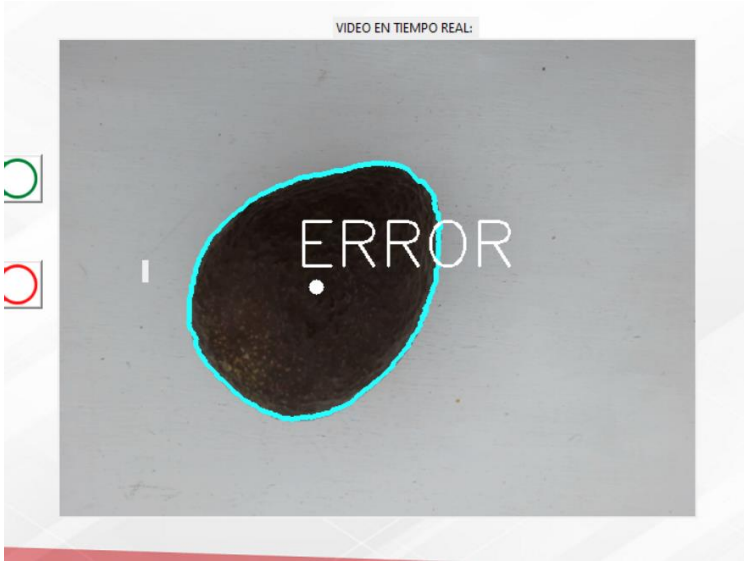
Considerando la elección del tipo de método de iluminación, se construye una cámara cerrada libre de entrada de iluminación del exterior que pueda modificar la adquisición de imagen, con un color homogéneo de fondo, se coloca la cámara en el centro de la fuente de iluminación, a una altura respecto de la base donde se aloja el objeto a clasificar.

3.7. Preprocesamiento de imagen

Una vez que el algoritmo inicie el video en tiempo real, en el bloque de funciones **def visualizar():** se procede a realizar el respectivo preprocesamiento de la imagen.

El algoritmo empieza leyendo cada fotograma, seguido con la línea **frame = imutils.resize(frame, width=600)**, se redimensiona el ancho de pixeles a 600para eliminar pixeles innecesarios del escenario como se ve en la Figura 36, luego se transforma los formatos de colores de RGB que censa la cámara a HSV con la función **hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)**, esto se realiza con el fin de poder expresar los colores para una fácil comprensión y debido a su ambigüedad.

Figura 35 *Redimensionamiento del tamaño de imagen a 600 pixeles.*



3.8. Segmentación de la imagen

Después de procesar la imagen se programa varias líneas de código con los cuales se pueden indicar los resultados de clasificación.

3.8.1. Rangos de colores en HSV

Se define la función **def visualizar ():**, en este bloque se tiene los rangos de colores en los componentes HSV a las cuales se va a clasificar el aguacate de variedad tipo Fuerte, están definidos tomando en cuenta la información del espacio de colores de [20] y realizando varias simulaciones hasta conseguir los siguientes rangos:

1. Rango de color para la segmentación del color de fruta en estado de madurez fisiológica.

   verde_1 =np.array([80, 0, 0])

   verde_11 = np.array([96, 255, 255])

2. Rango de color para la segmentación del color de fruta en estado tierno.

   verde_2 =np.array([56, 25, 0])

   verde_22 = np.array([75, 255 ,255])

Una vez definida los rangos de colores se procede a la umbralización, en este punto del código de programación, se da lugar a la detección de los colores los cuales se parametrizó con por rangos en HSV.

3.8.2. Umbralización

1. Línea de código para la umbralización del aguacate de variedad tipo Fuerte en estado de madurez fisiológica.

   cara1 = cv2.inRange(hsv, verde_1, verde_11)

2. Línea de código para la umbralización del aguacate de variedad tipo Fuerte en estado de madurez fisiológica.

   cara2 = cv2.inRange(hsv, verde_2, verde_22)

3.8.3. Detección de Contornos y Etiqueta de la Segmentación

En este punto se genera los contornos y el etiquetado a partir de la umbralización, estos resultados son visibles en la interfaz gráfica desarrollada como se ve en la Fgura 36. A continuación, se muestra las líneas de código que tienen la función de dibujar los contornos en cada producto clasificado:

1. Contorno para clasificación por estado de madurez fisiológica.

   contorno_1 = cv2.findContours(cara1, cv2.RETR_TREE,

   cv2.CHAIN_APPROX_SIMPLE)

   contorno_1 = imutils.grab_contours(contorno_1)

2. Contorno para clasificación en estado tierno.

   contorno_2 = cv2.findContours(cara2, cv2.RETR_TREE,

   cv2.CHAIN_APPROX_SIMPLE)

contorno_2 = imutils.grab_contours(contorno_2)

Para el etiquetado se realiza un ciclo for, teniendo en cuenta que este aparecerá en el centro del área del contorno, y aparecerá los textos ESTADO DE MADUREZ o TIERNO, según sea el caso de clasificación como se muestra en la Figura 36.

1. Texto para aguacate clasificado en madurez fisiológico

   cv2.putText(frame, "ESTADO FISIOLOGICO",(cx-20, cy-20),cv2.FONT_ITALIC, 2, (255, 255, 255), 2)

2. Texto para aguacate clasificado como tierno

   cv2.putText(frame, "TIERNO",(cx-20, cy-20),cv2.FONT_ITALIC, 2, (255, 255, 255), 2)

Figura 36 *Aguacate de variedad Fuerte clasificado.*

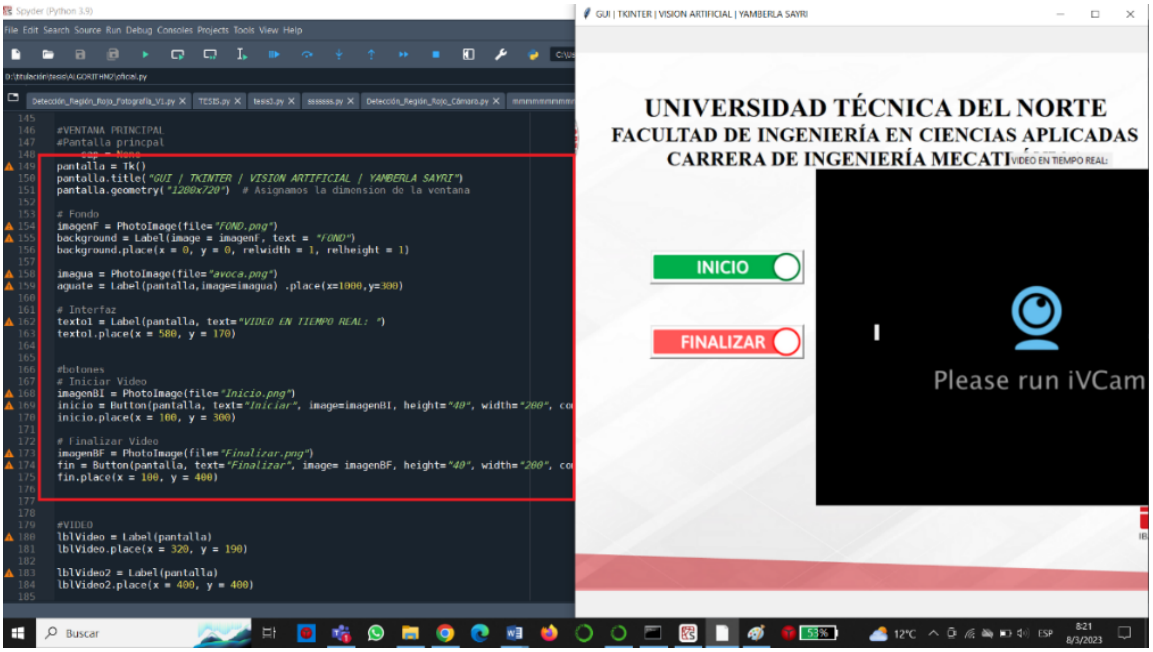
3.9. Creación de la interfaz gráfica

El desarrollo de la interfaz gráfica está realizado en la librería Tkinter del lenguaje de programación Python, este es sin duda una de las mejores opciones para desarrollar aplicaciones

de escritorio, y se recomienda también elegir las librerías dependiendo del proyecto a ejecutar, por el simple hecho de que, algunas interfaces no son compatibles con librerías y formatos.

En este caso el algoritmo de programación consta de varios bloques como funciones para facilitar las funcionalidades de los botones que aparecen en la interfaz gráfica. El diseño de la interfaz gráfica se realiza por código, como se muestra en la Figura 37. En el lenguaje de Python, también existen extensiones de desarrollo por plantillas.

Figura 37 *Creación de interfaz por código.*

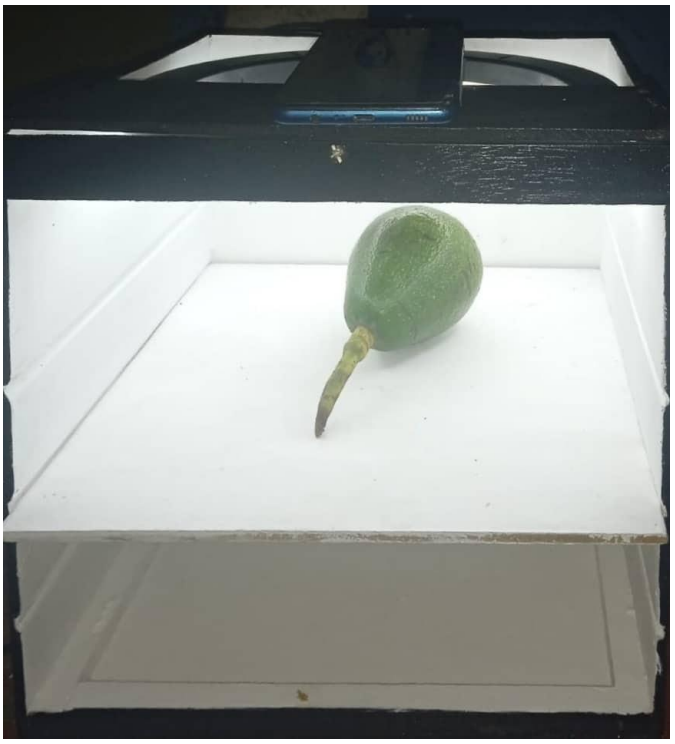


En la Figura 36 se puede observar el botón de **INICIO**, con este inicializamos la ejecución del código de programación, también la pantalla en donde se puede visualizar el resultado de clasificación, luego de la jornada, se puede finalizar el funcionamiento con el botón **FINALIZAR.** En la sección de anexos se puede observar el código de programación.

3.10. Resultados de funcionamiento

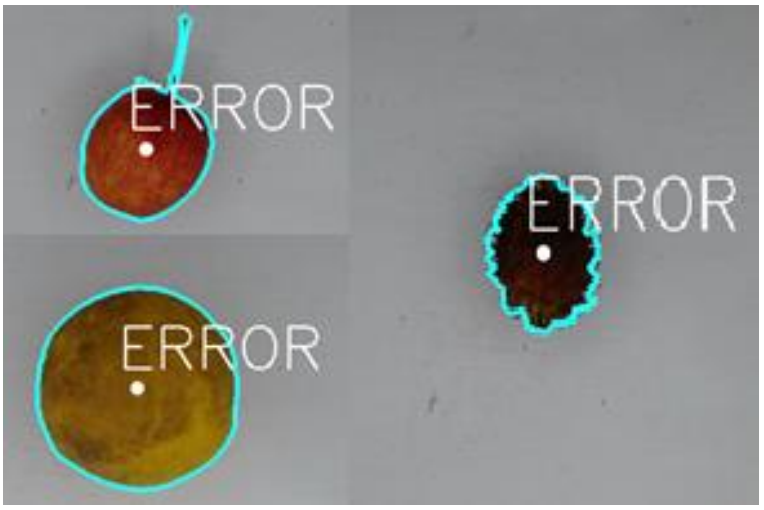
La validación del algoritmo se realiza mediante pruebas con diferentes tipos de muestras de aguacate de variedad de tipo Fuerte, se tomaron 30 y se pusieron cada una de ellas en la base del prototipo como se muestra en la Figura 38, cabe recalcar que el rango de colores para la clasificación se tomó por entrevistas realizada a los encargados y dueño de la Quinta de aguacates quien tiene una trayectoria amplia en la producción de este producto.

Figura 38 *Prueba de funcionamiento con aguacate tierno.*



Es importante que, la clasificación sea netamente de aguacates de la variedad Fuerte. Por ende, se realizó pruebas de funcionamiento, se introdujo dos muestras de diferentes aguacates de la variedad Hass y Fuerte. También se introdujo otros frutos como los mostrados en la Figura 39, tomando la tabulación que se muestra en la Tabla 9, se observa que el algoritmo segmenta únicamente el fruto deseado.

Figura 39 *Prueba de validación de algoritmo con otras frutas.*



No obstante, el algoritmo no toma en cuenta el color del aguacate Hass como se muestra en la Figura 40, es decir que el rango de colores optado es el adecuado. Además nos muestra las etiquetas de ESTADO FISIOLÓGICO y TIERNO. Mediante esta prueba se verifica o se valida que el funcionamiento del algoritmo está en su correcto funcionamiento.

Figura 40 *Validación del algoritmo con aguacate de la variedad Hass.*

Tabla 9 *Clasificación de muestras de aguacates de diferente tipo.*

N°	AGUACATE TIPO	PRODUCTO SEGMENTADO
1	Fuerte	Si
2	Hass	No
3	Hass	No
4	Fuerte	Si
5	Fuerte	Si
6	Fuerte	Si
7	Fuerte	Si
8	Hass	No
9	Fuerte	Si
10	Fuerte	Si
11	Hass	No
12	Fuerte	Si
13	Fuerte	Si
14	Fuerte	Si
15	Fuerte	Si
16	Fuerte	Si
17	Fuerte	Si
18	Hass	No
19	Fuerte	Si
20	Fuerte	Si
21	Fuerte	Si
22	Hass	No
23	Fuerte	Si
24	Fuerte	Si
25	Fuerte	Si
26	Hass	No
27	Fuerte	Si
28	Fuerte	Si
29	Fuerte	Si
30	Fuerte	Si

La prueba final o validación del algoritmo, el procedimiento consiste en iniciar el funcionamiento del algoritmo, introducir muestras de aguacate de la variedad Fuerte sobre la cama

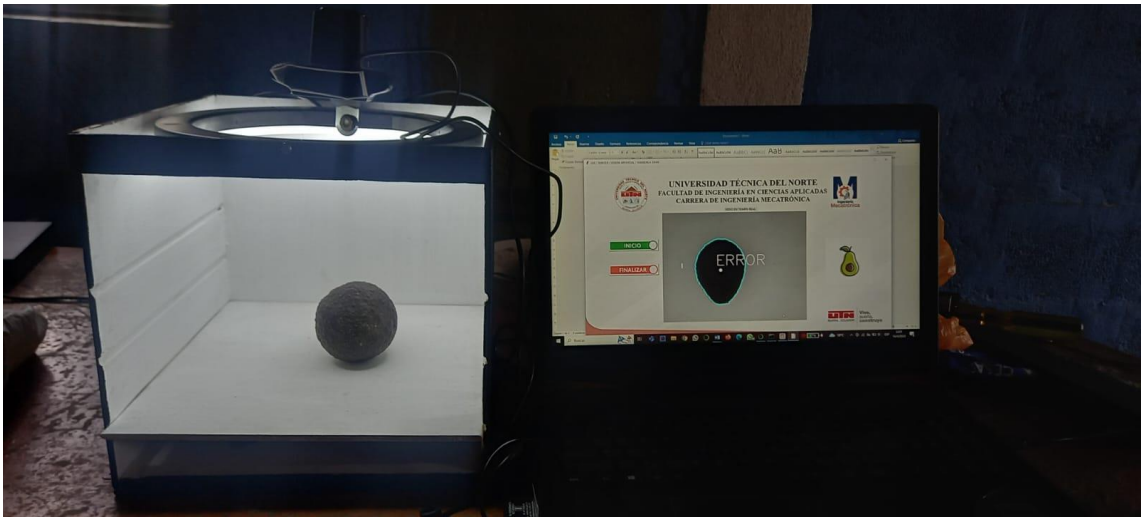
situada en el prototipo, este producto fue clasificado anteriormente con criterios del estado de madurez fisiológica del productor, cabe destacar que el personal en postcosecha tiene más de 10 años de experiencia en producción de aguacates de este y diferentes tipos.

Tabla 10 *Tabulación de producto clasificado.*

N°	AGUACATE TIPO	CLASIFICCIÓN POR EL ALGORITMO	CLASIFICACIÓN POR PRODUCTOR
1	Fuerte	Madurez fisiológica	Tierno
2	Fuerte	Madurez fisiológica	Madurez fisiológica
3	Fuerte	Tierno	Madurez fisiológica
4	Fuerte	Madurez fisiológica	Madurez fisiológica
5	Fuerte	Tierno	Tierno
6	Fuerte	Tierno	Tierno
7	Fuerte	Madurez fisiológica	Madurez fisiológica
8	Fuerte	Madurez fisiológica	Madurez fisiológica
9	Fuerte	Madurez fisiológica	Madurez fisiológica
10	Fuerte	Madurez fisiológica	Madurez fisiológica
11	Fuerte	Madurez fisiológica	Madurez fisiológica
12	Fuerte	Madurez fisiológica	Madurez fisiológica
13	Fuerte	Madurez fisiológica	Madurez fisiológica
14	Fuerte	Madurez fisiológica	Madurez fisiológica
15	Fuerte	Madurez fisiológica	Madurez fisiológica
16	Fuerte	Madurez fisiológica	Madurez fisiológica
17	Fuerte	Madurez fisiológica	Madurez fisiológica
18	Fuerte	Madurez fisiológica	Madurez fisiológica
19	Fuerte	Madurez fisiológica	Madurez fisiológica
20	Fuerte	Madurez fisiológica	Madurez fisiológica
21	Fuerte	Madurez fisiológica	Madurez fisiológica
22	Fuerte	Madurez fisiológica	Madurez fisiológica
23	Fuerte	Madurez fisiológica	Madurez fisiológica
24	Fuerte	Tierno	Tierno
25	Fuerte	Madurez fisiológica	Madurez fisiológica
26	Fuerte	Madurez fisiológica	Madurez fisiológica
27	Fuerte	Madurez fisiológica	Madurez fisiológica
28	Fuerte	Madurez fisiológica	Madurez fisiológica
29	Fuerte	Madurez fisiológica	Madurez fisiológica
30	Fuerte	Madurez fisiológica	Madurez fisiológica

En la prueba de funcionabilidad realizada, se pudo demostrar que los rangos establecidos para la segmentación por colores están próximos a la exactitud, además se realizó diversas pruebas de funcionamiento adicionales para demostrar que un algoritmo de visión artificial es más rápido y eficaz que la clasificación por percepción de la vista humana.

Figura 41 *Prototipo funcional.*



La eficiencia que tiene el algoritmo se muestra tabulado en la Tabla 11, donde se observa claramente que de 30 aguacates de variedad tipo Fuerte, el error en la clasificación es de 1 unidad de producto y el error porcentual es de 3.33 como se indica en la Tabla 12.

Tabla 11 *Eficiencia de clasificación con el algoritmo.*

Muestra	30
Coincidencias	29
Error	1
Eficiencia de clasificación con el algoritmo	96.66%

Tabla 12 Porcentaje de *error*.

Muestra	30
Valor calculado	29
Valor real	30
Error	3.33%

En los resultados de este trabajo se puede apreciar que el proceso o secuencia que debe seguir un algoritmo de programación para clasificación por colores es la segmentación, resultado que coincide con la investigación documentada de Pasuy Quevedo[4]. Por otro lado, para tener fundamentada la calidad del aguacate por su color se toma en cuenta la información documentada de las normas INEN1755 de la página 2 hasta la 11 en las secciones 4.2, 4.3 y 4.4 [47], que concuerdan con la investigación sobre calidad de frutos frescos de Pasuy Quevedo del aguacate de variedad Hass. Además, la construcción de prototipado para la iluminación de ambiente controlado mencionado en la sección 1.8 y 3.4, que permite configurar los factores de iluminación se asemejan al modelo de Sherman De La Torre [48].

El software utilizado se acerca a las necesidades, requerimientos e indicaciones del beneficiado. Por lo dicho se sugiere Python como lenguaje de programación haciendo énfasis el alcance económico. Este es uno de los lenguajes universales de programación enfocados a la visión artificial y en los resultados se puede observar claramente que tiene una eficacia mayor al 90% como se muestra en la Tabla 11 previamente, con este resultado se puede constatar que la selección fue precisa y que los resultados de calidad obtenidas son semejantes a los resultados de Pasuy[4] que sobrepasa el 90% de eficiencia.

Adicionalmente se utiliza una cámara Logitech, para tomar capturas de video con enfoque automático, dando como resultado datos con más precisión para el proceso de segmentación.

CONCLUSIONES

En el Ecuador existe la norma técnica INEN 1755 como requisitos de calidad para los aguacates de variedad tipo Fuerte y Hass. Sin embargo, hay la inexistencia de normas técnicas que establezcan requisitos visuales para la clasificación del producto en estudio. Por ende, se realizar la entrevista al personal encargado de la clasificación del producto en postcosecha, tomando muestras de aguacates para hacer una comparativa del algoritmo versus el aguacate clasificado por la percepción de la vista humana y conocimiento empírico.

Los rangos cromáticos de los colores en HSV son extensos, este formato es uno de los indicados para realizar este tipo de algoritmos para un problema de este tipo. El algoritmo diseñado considera el color del aguacate de Variedad tipo Fuerte como parámetro de clasificación, el proceso para el diseño es el siguiente: Adquisición, preprocesamiento de imagen, segmentación y contornos.

Se diseñó una interfaz gráfica en donde se observa el resultado de clasificación de los aguacates de variedad Fuerte. Para ello se declaró variables como funciones que permiten interactuar con los botones INCIALIZAR y FINALIZAR que se encuentra en la ventana principal.

La validación del algoritmo se realizó en un prototipo con iluminación controlada, se toma aguacates clasificados por el personal calificado al azar, el resultado de eficiencia es del 96.66%, de 30 aguacates 1 fue clasificado erróneamente.

RECOMENDACIONES

Se recomienda emplear una cámara de alta gama para para adquisición de imagen de mayor calidad y buenos resultados a fin de obtener un procesamiento de imagen eficaz.

Para la creación de interfaces gráficas en visión artificial y si tiene como objetivo adaptarse a la clasificación por colores al espacio HSV, se recomienda utilizar módulos y bibliotecas que compatibles con el mismo.

La visión artificial requiere de un entorno o un dispositivo que facilite la captura de datos de muchos ejemplares bajo las mismas condiciones, lo que permite crear una base de datos específica para cada propósito.

REFERENCIAS

[1] W. Benalcazar, “Los fruticultores de imbabura apuntan a la exportación,” *El Comercio*, 2018.

[2] Ministerio de Agricultura, “Agricultores de Pichincha se capacitan en manejo de cultivo de aguacate,” 2017.

[3] CORPEI, “Perfil de aguacate,” 2019.

[4] P. Pasuy, “Clasificación de aguacates basado en visión por Computador,” 2019. Accessed: Apr. 30, 2023. [Online]. Available: http://repositorio.utn.edu.ec/handle/123456789/9364

[5] J. Medina and M. Mono, “Control de un brazo robótico para clasificar objetos sólidos con formas definidas utilizando cisión estereoscópica,” 2012.

[6] A. Romero, A. Marín, and J. Jiménez, “Sistema de clasificación por visión artificial de mangos tipo Tommy,” 2014.

[7] P. Contreras, C. Peña, and C. Riaño, “Módulo robótico para la clasificación de lulos (Solanum Quitoense) implementando visión artificial,” *INGE CUC*, vol. 10, no. 1, pp. 51–62, Jul. 2014, doi: 10.1/JQUERY.MIN.JS.

[8] A. Barrientos and L. López, “Historia y genética del aguacate,” 2000.

[9] E. Lavaire, *Manual Técnico del cultivo de Agucate de Honduras*. 2013.

[10] I. Ferro, “Ensayo Cultivo de Aguacate Ibague ,” 2001.

[11] El Comercio, “El aguacate tiene diferentes formas y sabores,” 2011.

[12] Instituto Ecuatoriano de Normalización, “FRUTAS FRESCAS. AGUACATE. REQUISITOS.,” 2009.

[13] F. Millán and Z. Ostojich, “Predicción mediante redes neuronales artificiales de la transferencia de masa en frutas osmóticamente deshidratadas,” *Interciencia: Revista de*

ciencia y tecnología de América, ISSN 0378-1844, Vol. 31, Nº. 3, 2006, págs. 206-210, vol. 31, no. 3, pp. 206–210, 2006.

[14] H. Paz and J. Magaly, "Desarrollo de un sistema de visión artificial para la detección de aglomeración de personas en un semáforo.," 2016.

[15] A. Romero, A. Marín, and J. Jiménez, "Sistema de clasificación por visión artificial de mangos tipo Tommy Classification system for artificial vision type Tommy mango," 2014, Accessed: Apr. 30, 2023. [Online]. Available: https://www.redalyc.org/pdf/5537/553756867002.pdf

[16] R. S. Zandonadi, F. A. C. Pinto, D. G. Sena, D. M. Queiroz, P. A. Viana, and E. C. Mantovani, "Identification of Lesser Cornstalk Borer-attacked Maize Plants using Infrared Images," *Biosyst Eng*, vol. 91, no. 4, pp. 433–439, Aug. 2005, doi: 10.1016/J.BIOSYSTEMSENG.2005.05.002.

[17] S. Osejos, "Desarrollo de un sistema para detectar los colores en tejidos con hilo utilizando visión artificial," 2022.

[18] M. Escobar and J. Castaño, "DETERMINACIÓN DEL ESTADO DE MADUREZ DEL AGUACATE MEDIANTE PROCESAMIENTO DE IMÁGENES CON LA RASPBERRY PI," 2018.

[19] U. O. Dorj, M. Lee, and S. seok Yun, "An yield estimation in citrus orchards via fruit detection and counting using image processing," *Comput Electron Agric*, vol. 140, pp. 103–112, Aug. 2017, doi: 10.1016/J.COMPAG.2017.05.019.

[20] F. R. Al-Osaimi, M. Bennamoun, and A. Mian, "Illumination normalization of facial images by reversing the process of image formation," *Mach Vis Appl*, vol. 22, no. 6, pp. 899–911, Nov. 2011, doi: 10.1007/S00138-010-0309-5/METRICS.

[21] S. Pineda, *Lógica de programación y algoritmos. Libro Guía*. 2006.

[22] J. López, *ALGORITMOS Y PROGRAMACIÓN GUÍA PARA DOCENTES*. 2009.

[23] G. Ayala, *Algoritmos y programación*. Universidad de las Américas Puebla - UDLAP, 2018.

[24] S. De la Torre, "Sistema de clasificación de huevos mediante un algoritmo de visión artificial," 2022.

[25] S. Umbaugh, *Digital Image Processing and Analysis: Applications with MATLAB*. 2018.

[26] V. Tyagi, *Understanding digital image processing*. 2018.

[27] P. Pasuy, "Clasificación de aguacates basado en visión por Computador," 2019.

[28] A. Kumar and S. P. Panda, "A Survey: How Python Pitches in IT-World," *Proceedings of the International Conference on Machine Learning, Big Data, Cloud and Parallel Computing: Trends, Prespectives and Prospects, COMITCon 2019*, pp. 248–251, Feb. 2019, doi: 10.1109/COMITCON.2019.8862251.

[29] Santander Universidades, "Python: qué es y por qué deberías aprender a utilizarlo," 2021. https://www.becas-santander.com/es/blog/python-que-es.html (accessed May 01, 2023).

[30] J. Nolasco, *Python Aplicaciones practicas*. 2011.

[31] NumPy, "NumPy user guide ," 2023. https://numpy.org/doc/stable/user/ (accessed May 01, 2023).

[32] A. Pawlik, J. Segal, H. Sharp, and M. Petre, "Crowdsourcing scientific software documentation: A case study of the NumPy documentation project," *Comput Sci Eng*, vol. 17, no. 1, pp. 28–36, Jan. 2015, doi: 10.1109/MCSE.2014.93.

[33] NumPy, "NumPy 1.23.0 Release Notes," 2023. https://numpy.org/devdocs/release/1.23.0-notes.html (accessed May 01, 2023).

[34] Pillow, “9.3.0 - Pillow (PIL Fork) 9.5.0 documentation,” 2023. https://pillow.readthedocs.io/en/stable/releasenotes/9.3.0.html (accessed May 01, 2023).

[35] D. Tivane, “tk-async-image ,” 2023. https://pypi.org/project/tk-async-image/ (accessed May 01, 2023).

[36] D. Acuña, “Visión artificial aplicada a la detección e identificación de personas en tiempo real,” Quito, 2019., 2019.

[37] S. K. Shammi, S. Sultana, M. S. Islam, and A. Chakrabarty, “Low latency image processing of transportation system using parallel processing co-incident multithreading (PPcM),” *2018 Joint 7th International Conference on Informatics, Electronics and Vision and 2nd International Conference on Imaging, Vision and Pattern Recognition, ICIEV-IVPR 2018*, pp. 363–368, Feb. 2019, doi: 10.1109/ICIEV.2018.8640957.

[38] G. Anitha, P. Roy, and S. Nashita, “E-SEEK - RFID based Detection System for Localizing the Misplaced Objects,” *Proceedings of the 6th International Conference on Communication and Electronics Systems, ICCES 2021*, pp. 989–993, Jul. 2021, doi: 10.1109/ICCES51350.2021.9488933.

[39] S. R. Hussain, K. R. Naidu, C. R. S. Lokesh, P. Vamsikrishna, and G. Rohan, “2D-game development using Raspberry Pi,” *2016 International Conference on Information Communication and Embedded Systems, ICICES 2016*, Jul. 2016, doi: 10.1109/ICICES.2016.7518858.

[40] S. Cubero, “Diseño e implementación de nuevas tecnologías basadas en visión artificial para la inspección no destructiva de la calidad de fruta en campo y mínimamente procesada,” Universitat Politècnica de València, Valencia (Spain), 2012. doi: 10.4995/THESIS/10251/15999.

[41] G. Lozano and J. Rodriguez, “Diseño de un sistema de visión artificial para la revisión del nivel de llenado de bebidas embotelladas.,” Universidad Autónoma del Caribe, 2015.

[42] Cooperación Alemana, “Eficiencia Energética en la Iluminación,” 2023.

[43] J. Bermeo, “GESTIÓN Y DESARROLLO TURÍSTICO EN LA COMUNIDAD SAN FRANCISCO DE CUNUGUACHAY, PARROQUIA SANTIAGO DE CALPI, PROVINCIA DE CHIMBORAZO,” 2011.

[44] C. Prieto, “Adaptación de las Metodologías Tradicionales Cascada y Espiral para la Inclusión de Evaluación Inicial de Usabilidad en el Desarrollo de Productos de Software en México.,” 2015.

[45] MaxiTec, “Aro de luz led profesional de 25.4 cm - SLI-VL120 - MaxiTec,” 2023. https://www.maxitec.com.ec/slide-aro-de-luz-led-profesional-de-254-cm-sli-vl120/p (accessed May 01, 2023).

[46] LOGITECH, “Cámara web HD PRO Logitech C920.” https://www.logitech.com/es-es/products/webcams/c920-pro-hd-webcam.960-001055.html (accessed Apr. 30, 2023).

[47] Instituto Ecuatoriano de Normalización, “FRUTAS FRESCAS. AGUACATE. REQUISITOS.,” 2009.

[48] S. De la Torre, “Sistema de clasificación de huevos mediante un algoritmo de visión artificial,” 2022. Accessed: Apr. 30, 2023. [Online]. Available: http://repositorio.utn.edu.ec/handle/123456789/13160

ANEXOS

Anexo 1 Entrevista

Nombre de entrevistado: Juan Guatemal (propietario de sembrío de agucate)

Lugar: Pichincha, Quito, Guayllabamba, Doña Anna

Tema para tratar: calidad del aguacate de tipo Fuerte para la comercialización.

¿Por qué busca la calidad del aguacate de variedad Fuerte para la comercialización?

Juan Guatemal: los ingresos económicos en este negocio, depende de la calidad de aguacate para comercializar, una vez cosechado el aguacate se procede a la clasificación por calidad, por el tamaño, por el estado de madurez, por los daños mecánicos ocasionados durante la cosecha.

¿Tiene conocimiento sobre la agricultura de precisión?

Juan Guatemal: tome un curso sobre agricultura de precisión e inteligencia artificial con estudiantes de la Universidad Técnica del Norte, eran estudiantes que hacían vinculación con la comunidad. Lo que aprendí en el curso fue sobre como la tecnología a tomado lugar en la agricultura, que se pueden realizar controles de calidad de suelos, de enfermedades en las cosechas para tener como resultado en excelente producto para la comercialización. Luego de haber sabido toda esta información quería solucionar un problema existente en mi negocio, mi problema es la devolución de los aguacates de la variedad tipo Fuerte, debido a la clasificación en postcosecha por así decirlo, esto ocurre por el cansancio físico de mis trabajadores, y además de eso cansancio visual ocasionados por la iluminación del sol, además que la clasificación se realiza en la noche y no se tiene la misma iluminación del día. Pero veo que para implementar un sistema de agricultura de precisión es muy cara y económicamente no he podido estabilizarme de los problemas económicos causado por la pandemia del COVID 19 y los paros nacionales.

El propósito de esta entrevista es para proponerle el diseño de un algoritmo de visión artificial para la clasificación de su producto de comercialización, ¿estaría interesado en esta idea?

Juan Guatemal: como anteriormente lo dije, estoy interesado en la agricultura de presión y me interesa su propuesta. Estoy gustoso en aplicar algo tecnológico para aumentar la calidad de mis productos.

¿Qué requerimientos debe tener el algoritmo?

Juan Guatemal: el algoritmo, como anteriormente le dije, no debería ser costoso, también es muy necesario la rapidez y la precisión para clasificar, tiene que ser funcional y fácil de utilizar.

Anexo 2 Algoritmo para la clasificación de aguacates de variedad tipo Fuerte.

```
# Importamos librerias
# En esta linea se importa toda la librerÃ-a de tkinter para construir la GUI
from tkinter import *

# Importamos la librerÃ-a de PIL, permite interactuar el video
# en tiempo real con la GUI, para eso tambiÃ©n se necesita los mÃ³dulos Image e Image Tk
from PIL import Image, ImageTk

#librerÃ-a de OpenCv
import cv2

# ImportaciÃ³n de la librerÃ-a para llamar a la interfaz de Open Cv de manera concisa
import imutils

# Se importa la librerÃ-a numpy para el analisis de datos.
import numpy as np   #

#NOTA:HSV
```

```
#HUE: en este canal codifica las tonalidades de color: rojo, amarillo, verde, azul
#SATURACIÃ“N: en este canal se codifica la pureza que tiene el color.
#VALUE: en este canal se codifica la luminosidad del color, asÃ- entre menos sea mÃ¡s scuro es el color.

# FUNCIÃ“N PARA INICIAR EL VIDEO EN TIEMPO REAL
def iniciar():
    # Se crea una variable global porque se va a utilizar en otras funciones como variable.
    global cap
    # Para capturar el video en tiempo real, en este caso se cambia e 0 dependiendo de las camaras añadidas a los puertos usb del computador
    cap = cv2.VideoCapture(3, cv2.CAP_DSHOW)
    # Nos ayuda a ver cada fotograma en la interfaz gráfica.
    visualizar()

def visualizar(): # Creación de la funcion vizualisar.
    global cap #,frame, ret, hsv, verde_1, verde_11, verde_2, verde_22, contorno_1, contorno_2, cara1, cara2
#se abre el video en tiempo real y que siga abierto hasta la función funalizar
    if cap is not None: # Si se ha iniciado Cap procedemos a leer la captura.
        ret, frame = cap.read()
        if ret == True:
            frame = imutils.resize(frame, width=600)
            frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

            #se define los rangos de colores

            #MADUREZ FISIOLÓGICA
            verde_1 =np.array([80, 0, 0])
            verde_11 = np.array([96, 255, 255])

            #TIERNO
            verde_2 =np.array([56, 25, 0])
            verde_22 = np.array([75, 255 ,255])

            #MADUREZ FISIOLÓGICA
            err_1 =np.array([96, 25, 0])
```

```
err_11 = np.array([179, 255, 255])

#ERROR
err_2 =np.array([0, 25, 0])
err_22 = np.array([56, 255 ,255])

# se transforma el color de RGB que adquiere la camara a HSV
hsv=cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

# se realiza las mascaras
#color para aguacate en estado de MF
cara1 = cv2.inRange(hsv, verde_1, verde_11)

#color para aguacate tierno
cara2 = cv2.inRange(hsv, verde_2, verde_22)

#ERROR AGUACATES
cara3 = cv2.inRange(hsv, err_1, err_11)

#color para aguacate tierno
cara4 = cv2.inRange(hsv, err_2, err_22)

contorno_3 = cv2.findContours(cara3, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
contorno_3 = imutils.grab_contours(contorno_3)

contorno_4 = cv2.findContours(cara4, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
contorno_4 = imutils.grab_contours(contorno_4)

#se define los contornos de los objetosa clasificar
contorno_1 = cv2.findContours(cara1, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
contorno_1 = imutils.grab_contours(contorno_1)

contorno_2 = cv2.findContours(cara2, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
contorno_2 = imutils.grab_contours(contorno_2)
```

se realiza un ciclo for para obcervar los contornos de acuerdo a sus areas en pixeles para quitar pequeños ruidos.

```
        for c in contorno_1:
            area1 = cv2.contourArea(c)
            # para quitar los ruidos de bordes se utiliza la secuencia if siguiente
            if area1>10000:
                cv2.drawContours(frame, [c], -1, (38, 255, 255), 3)

                #se la pocicion del objeto para el etiquetado de la imagen procesada
                M = cv2.moments(c)

                # se define las coordenadas en X
                cx = int(M["m10"]/M["m00"])

                # se define las coordenadas en Y
                cy = int(M["m01"]/M["m00"])

               # se define un circulo que este en el centro del frame
                cv2.circle(frame, (cx, cy), 7, (255, 255, 255), -1)

                #se define qel texto que se desea poner en el aguacate clasificado
                cv2.putText(frame, "ESTADO FISIOLOGICO",(cx-20, cy-20),cv2.FONT_ITALIC, 2,
(255, 255, 255), 2)

        for c in contorno_2:
            area1 = cv2.contourArea(c)
            # para quitar los ruidos de bordes se utiliza la secuencia if siguiente
            if area1>5000:
                cv2.drawContours(frame, [c], -1, (38, 255, 255), 3)

                #se la pocicion del objeto para el etiquetado de la imagen procesada
                M = cv2.moments(c)

                # se define las coordenadas en X
                cx = int(M["m10"]/M["m00"])

                # se define las coordenadas en Y
                cy = int(M["m01"]/M["m00"])
```

```
            # se define un circulo que este en el centro del frame
            cv2.circle(frame, (cx, cy), 7, (255, 255, 255), -1)

            #se define qel texto que se desea poner en el aguacate clasificado
            cv2.putText(frame, "TIERNO",(cx-20, cy-20),cv2.FONT_ITALIC, 2, (255, 255, 255),
2)

        for c in contorno_3:
            area1 = cv2.contourArea(c)
            # para quitar los ruidos de bordes se utiliza la secuencia if siguiente
            if area1>5000:
                cv2.drawContours(frame, [c], -1, (38, 255, 255), 3)

                #se la pocicion del objeto para el etiquetado de la imagen procesada
                M = cv2.moments(c)

                # se define las coordenadas en X
                cx = int(M["m10"]/M["m00"])

                # se define las coordenadas en Y
                cy = int(M["m01"]/M["m00"])

                # se define un circulo que este en el centro del frame
                cv2.circle(frame, (cx, cy), 7, (255, 255, 255), -1)

                #se define qel texto que se desea poner en el aguacate clasificado
                cv2.putText(frame, "ERROR",(cx-20, cy-20),cv2.FONT_ITALIC, 2, (255, 255, 255),
2)

        for c in contorno_4:
            area1 = cv2.contourArea(c)
            # para quitar los ruidos de bordes se utiliza la secuencia if siguiente
            if area1>5000:
                cv2.drawContours(frame, [c], -1, (38, 255, 255), 3)

                #se la pocicion del objeto para el etiquetado de la imagen procesada
                M = cv2.moments(c)
```

```
                # se define las coordenadas en X
                cx = int(M["m10"]/M["m00"])

                # se define las coordenadas en Y
                cy = int(M["m01"]/M["m00"])

               # se define un circulo que este en el centro del frame
                cv2.circle(frame, (cx, cy), 7, (255, 255, 255), -1)

                #se define qel texto que se desea poner en el aguacate clasificado
                cv2.putText(frame, "ERROR",(cx-20, cy-20),cv2.FONT_ITALIC, 2, (255, 255, 255),
2)

            # Incorporamos el frame a la GUI
            im = Image.fromarray(frame)
            # cambiamos la imagen al formato de Tk para que aparezaca en la GUI
            img = ImageTk.PhotoImage(image=im)

            # se ubica img en la GUI
            lblVideo.configure(image=img)
            # Se agrega img en la  GUI, es decir se agrega los fotogramas para que no sean
borrados
            lblVideo.image = img
            # Se lee constantemente los fotogramas de una manera secuencial
            # con after se llama a visualizar cada 10ms.
            lblVideo.after(10, visualizar)

            # Cuando se haya finalizado la funcion finalizar, se apagará la camara.

        else:
            lblVideo.image = ""
            cap.release()

# FUNCIÓN FINALIZAR
def finalizar():
    global cap
```

```
    cap.release()

#VENTANA PRINCIPAL
#Pantalla princpal
    cap = None
pantalla = Tk()
pantalla.title("GUI | TKINTER | VISION ARTIFICIAL | YAMBERLA SAYRI")
pantalla.geometry("1280x720")  # Asignamos la dimension de la ventana

# Fondo
imagenF = PhotoImage(file="FOND.png")
background = Label(image = imagenF, text = "FOND")
background.place(x = 0, y = 0, relwidth = 1, relheight = 1)

imagua = PhotoImage(file="avoca.png")
aguate = Label(pantalla,image=imagua) .place(x=1000,y=300)

# Interfaz
texto1 = Label(pantalla, text="VIDEO EN TIEMPO REAL: ")
texto1.place(x = 580, y = 170)

#botones
# Iniciar Video
imagenBI = PhotoImage(file="Inicio.png")
inicio = Button(pantalla, text="Iniciar", image=imagenBI, height="40", width="200",
command=iniciar)
inicio.place(x = 100, y = 300)

# Finalizar Video
imagenBF = PhotoImage(file="Finalizar.png")
fin = Button(pantalla, text="Finalizar", image= imagenBF, height="40", width="200",
command=finalizar)
fin.place(x = 100, y = 400)

#VIDEO
lblVideo = Label(pantalla)
lblVideo.place(x = 320, y = 190)
```

```
lblVideo2 = Label(pantalla)
lblVideo2.place(x = 400, y = 400)

pantalla.mainloop()
```