UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

DETECCIÓN Y CLASIFICACIÓN AUTOMÁTICA DE COLORES DE FRUTAS PARA LA APLICACIÓN MÓVIL QCFORMS

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERA CIVIL EN COMPUTACIÓN

ROSARIO DEL PILAR MOLINA FERREIRO

PROFESOR GUÍA:
BENJAMÍN BUSTOS CÁRDENAS

MIEMBROS DE LA COMISIÓN:
JUAN MANUEL BARRIOS NÚÑEZ
ANDRÉS ABELIUK KIMELMAN

SANTIAGO DE CHILE
2023

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE INGENIERA CIVIL EN COMPUTACIÓN
POR: ROSARIO DEL PILAR MOLINA FERREIRO
FECHA: 2023
PROF. GUÍA: BENJAMÍN BUSTOS CÁRDENAS

DETECCIÓN Y CLASIFICACIÓN AUTOMÁTICA DE COLORES DE FRUTAS PARA LA APLICACIÓN MÓVIL QCFORMS

El control de calidad es crucial en la cadena de suministro de frutas frescas, y todas las frutas y verduras son evaluadas para asegurarse de que cumplan con los requisitos del consumidor. El color es una de las características más importantes evaluadas por los consumidores al comprar frutas y verduras. El control de calidad de frutas a granel es importante para determinar el tipo de procesamiento y embalaje que se va a realizar para su comercialización. Actualmente, la clasificación de color se realiza manualmente durante las inspecciones de control de calidad, lo que puede llevar a inconsistencias y subjetividad debido a las diferentes condiciones de iluminación y percepciones de cada inspector. La clasificación de color de limones a granel en bins es un ejemplo de una instancia que requiere la clasificación manual uno a uno de cada limón.

QCForms es una empresa chilena desarrolladora de software para el control de calidad de frutas. QCforms cuenta con una plataforma y aplicación móvil para la inspección de frutas. Parte de la inspección de calidad de frutas consiste en la clasificación de color, actualmente esta clasificación se realiza por inspección visual y el resultado se ingresa de manera manual en la aplicación.

En el presente trabajo se desarrolló una nueva funcionalidad para la aplicación móvil de QCForms, que consiste en la clasificación automática de color de limones por cada bin a partir del procesamiento de una foto de este y una foto de la plantilla patrón de colores para la clasificación. Esta nueva funcionalidad permitirá reducir significativamente los tiempos que tarda la inspección, así como también eliminar la subjetividad en la clasificación de color.

Específicamente se desarrolló un método para la detección de limones, la detección de una plantilla de color y la clasificación de los limones de acuerdo con dicha plantilla. Esta nueva funcionalidad se ejecuta de manera local en el dispositivo móvil y tiene un tiempo de ejecución menor a 5 segundos.

También se realizó una prueba en terreno de la aplicación desarrollada donde se obtuvieron resultados concretos y funcionales además de una retroalimentación respecto a cuales aspectos se deben definir (inspección en condiciones controladas) y cuales aspectos mejorar en la funcionalidad desarrollada.

En la evaluación del algoritmo con los usuarios este mostró errores en la detección de los patrones de color, lo cuál debe ser corregido. Además se encontraron dificultades para evaluar el algoritmo de forma objetiva utilizando métricas comunes de evaluación, por lo cual se proponen modificaciones para el trabajo futuro.

A mi familia y amigas, por su constante apoyo y ayuda para seguir adelante.

Tabla de Contenido

Capítulo 1

Introducción

La fruta fresca representa una gran parte del consumo mundial de alimentos. A la hora de comprar frutas y verduras en un supermercado, verdulería o en un mercado, los consumidores evalúan ciertas características de las frutas y verduras que consumen, como el color, el tamaño, la consistencia y, por supuesto, el sabor. Para que esta práctica sea satisfactoria, todas las frutas y verduras pasan por un control de calidad, el cual evalúa las características de la fruta y permite asegurar que efectivamente se cumplirán los requisitos que el consumidor espera.

El control de calidad es esencial en la cadena de suministro mundial de fruta fresca. Pese a su amplio uso e importancia, los métodos de control de calidad siguen siendo en su mayoría manuales, subjetivos e ineficientes. Un ejemplo es la clasificación del color de las frutas, la cuales es realizada por inspección visual. Sin embargo, la adaptación del ojo humano a ligeros cambios de color, el efecto del fondo sobre el color percibido y la intensidad del color son los principales inconvenientes de la inspección visual.

Para eliminar las inconsistencias en la clasificación de color durante el proceso de inspección de calidad, se propone utilizar una tecnología de visión artificial basada en el análisis de imágenes. La visión mediante sistemas computarizados proporciona una alternativa automatizada para una técnica de análisis no destructiva, de calidad precisa, rápida y objetiva para realizar una clasificación de colores de fruta de acuerdo con estándares, parámetros y patrones predefinidos.

La propuesta es trabajar con QCforms, una plataforma y aplicación móvil para la inspección de frutas, en la cual actualmente la clasificación de color de los frutos inspeccionados se realiza de forma manual. La propuesta es implementar una nueva funcionalidad a la aplicación que permita a los inspectores tomar una foto de los productos y obtener de forma automática la clasificación y distribución de colores de estos. Implementar esta nueva funcionalidad implica significativas restricciones tanto en hardware como software, las cuales serán descritas en este informe.

1.1. Identificación y formulación del problema

Actualmente la clasificación de color en inspecciones de control de calidad se realiza manualmente, lo que puede llevar a inconsistencias dado las diferentes condiciones de iluminación

así como también la subjetividad que surge por la percepción de cada inspector. Es por esto que es necesario eliminar las inconsistencias en la clasificación de color. Una de las formas de realizar esto es mediate un software de visión artificial basada en el análisis de imágenes.

Una de las inspecciones de frutas de mayor dificultad en la clasificación de color es la clasificación de limones en bins, el cual se puede ver en la Figura 1.1. Los Bins son contenedores utilizados para la cosecha de limones y en la actualidad se utilizan principalmente de material plástico ya que les proporciona una mayor vida útil. Un Bin de plástico lleno permite contener entre 340 kg a 380 kg de limones, y en promedio (dependiendo del calibre del limón) contienen aproximadamente unos 2.600 limones. El tamaño estándar de los bins plásticos utilizados, es de base cuadrada de 1220 mm por una altura de 775 mm.

En la actualidad el tamaño de muestra para control de calidad de recepción de limones (control manual), es de 100 frutos, los cuales son seleccionados aleatoriamente desde un bin y luego deben ser clasificados individualmente. La fruta es recibida diariamente desde los lugares de cosecha en camiones o remolques correspondiente a un lote, cada lote es de la misma variedad, del mismo productor y lugar de cosecha. Cada lote puede estar compuesto entre 10 a 80 bins con un promedio de 32 bins por camión o remolque.

En inspecciones de otros frutos la cantidad de frutos a inspeccionar es considerablemente menor. Es por esto que el trabajo actual se enfocará en la clasificación de colores de limones en estos bins.

En este trabajo las imágenes con las que se trabaja son de bins cuyos limones corresponde a la variedad Eureka. El periodo de cosecha transcurre desde mayo hasta agosto, aunque entre mayo y junio se concentra la mayor parte de la producción. El tamaño de los limones varía entre 45 y 90 mm (diámetro ecuatorial) y estos se clasifican por calibre (que equivale al total de frutos por caja de 17.2 kg) y varía entre 75 y 235 kg. El peso de cada limón fluctúa entre los 100 y 200 gr. Para los efectos del control de calidad para todas las variedades de limones se utilizan los mismos parámetros de medición de tamaño y color.

Figura 1.1: Bin de limones

1.1.1. Restricciones de software

La solución al problema de clasificación de colores de limones se integrará a la actual aplicación de QCForms, la cual es usada en dispositivos móviles, mayormente tablets, cuya RAM en muchos casos no supera los 4GB.

Otra restricción que existe es el acceso a Internet. La inspección de calidad de frutas muchas veces se lleva a cabo en frigoríficos donde no hay señal para conectarse a la web, por lo que la solución debe realizarse de forma local en el dispositivo móvil.

1.2. Objetivos

1.2.1. Objetivo general

El objetivo general de este trabajo de memoria es incluir en la aplicación móvil de QCforms una funcionalidad donde el inspector pueda tomar una foto a un bin de limones para obtener la distribución de colores según el patrón de clasificación respectivo. En la Figura 1.2 se puede ver un ejemplo de patrón de clasificación de colores.

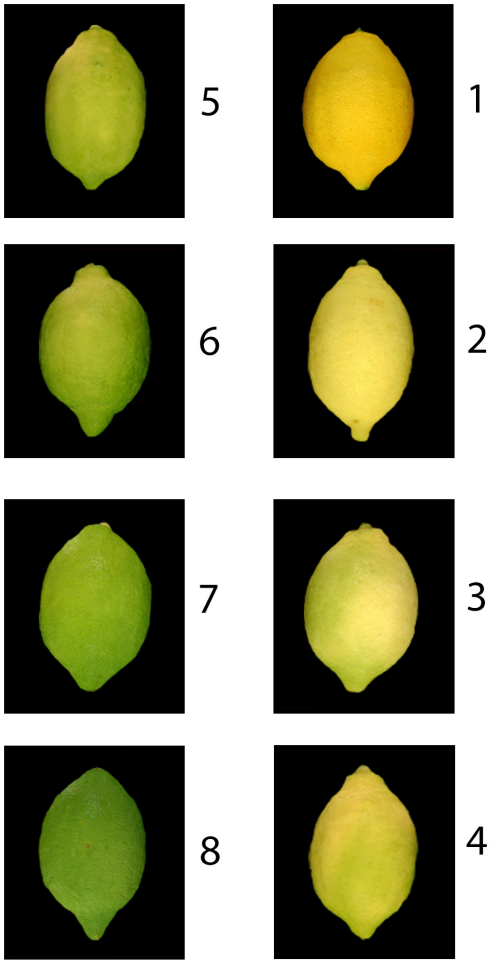


Figura 1.2: Carta de colores para limones

1.2.2. Objetivos específicos

Específicamente se trabajará en crear una solución a la clasificación de colores en el proceso de inspección de calidad, que elimine la subjetividad e inconsistencia en la evaluación, producto de que la clasificación se realiza mediante inspección visual. Se buscará una solución a la clasificación de colores en bins de limones. Además, se busca implementar una solución que disminuya el tiempo que toma realizar esta parte de la inspección de calidad.

1. Implementar un método de detección de colores en un bin de limones
   a) Implementar un método para la segmentación de limones
   b) Implementar un método para el reconocimiento de la plantilla de colores
   c) Implementar un método para la clasificación de colores de limones de acuerdo a la plantilla de colores
2. Cumplir con las restricciones de tiempo y acceso a Internet
3. Integrar el método de detección de colores a la plantilla de QCForms
4. Evaluar los algoritmos de detección y clasificación

5. Evaluar la usabilidad de la aplicación

1.3. Organización del trabajo de memoria

En el siguiente capítulo, Marco Teórico y Estado del Arte, se presentan los conceptos importantes para el presente trabajo, luego en el capítulo 3 se detalla la metodología de trabajo a seguir, específicamente la solución propuesta y cómo esta será evaluada.

En el capítulo 4 se exponen los algoritmos desarrollados junto con la integración de estos a la aplicación móvil.

En el capítulo 5 se exponen los resultados de los algoritmos además de los resultados de la evaluación de usabilidad.

Finalmente en el capítulo 6 se procede a concluir, analizando los resultados obtenidos.

Capítulo 2

Marco Teórico y Estado del Arte

2.1. Situación actual

Actualmente la mayor parte de las inspecciones de calidad se realizan de forma manual, desde la toma de medidas de tamaño, peso, identificación de defectos y la clasificación de color, entre otros.

2.2. Control de calidad de frutas

El control de calidad es una evaluación realizada a diferentes niveles y escalas a frutas y verduras para determinar si cumplen adecuadamente con las características esperadas para llegar a un consumidor final. El control de calidad se realiza en el campo antes y después de la cosecha, en las plantas de proceso donde se embala la fruta antes de la exportación o distribución, en los puertos de llegada de la fruta exportada y en los frigoríficos de distribución hacia los mercados. Definir los parámetros de calidad de las frutas y su medición para la aceptación es complejo. Los métodos de control de calidad no están regulados y están sujetos a diferentes estándares de inspección para diferentes compradores, empresas productoras, exportadoras y vendedoras. Madurez, color, uniformidad, tamaño y cuantificación de defectos son algunas de las componentes que definen la calidad. La evaluación de estas medidas puede ser subjetiva, es decir, se evalúa manualmente según el criterio del inspector, u objetiva, cuando son medidas por un instrumento.

El color de la superficie de los frutos es el primer elemento que observa el consumidor y tiene una gran influencia en la selección que realiza. El color de la fruta depende de varios factores, incluyendo la exposición solar, la temperatura, la humedad, la absorción de nutrientes y los cambios bioquímicos que ocurren durante el crecimiento, la maduración, el manejo y procesamiento posterior a la cosecha. Es por esto que el color de los alimentos es un excelente indicador de su calidad, ya que refleja estas condiciones y por lo tanto es uno de los atributos de calidad del producto más importantes en el control de calidad.

El color en la mayoría de las frutas tiene una relación con el estado de madurez, y con una adecuada clasificación de color se pueden controlar las condiciones adecuadas de maduración y almacenamiento, permitiendo a los productores de frutas enviar productos que alcanzarán su punto máximo de madurez en el momento óptimo.

Además, en muchas especies de fruta el color es un factor muy importante para determinar el mercado en el cual se comercializará, y también puede representar una diferencia importante en el valor final de venta, ya que el precio de la fruta está cada vez más ligado a la calidad del producto final y en ello el color es un factor fundamental para determinar la apariencia de la fruta, además de inferir a través de su clasificación el nivel de maduración en que se encuentra.

2.3. Instancias de medición de color de frutas

Se distinguen al menos cuatro instancias en las cuales se efectúan clasificaciones de colores de fruta.

2.3.1. Control en el proceso de crecimiento

El control de color de fruta durante el proceso de crecimiento proporciona una herramienta de monitoreo del proceso de crecimiento. Mediante el análisis de la evolución de colores durante el proceso de crecimiento, se puede determinar el estado de maduración óptima para su cosecha.

Esta instancia se utiliza principalmente para ciertos frutos en los cuales está demostrado que hay una correlación directa en niveles de sólidos solubles medidos en grados brix y el color del fruto.

En la actualidad el control de calidad y en específico el control de color de fruta en huerto, se hace en las cercanías del periodo de cosecha de la fruta con el objetivo de determinar el punto óptimo en el cual efectuar la recolección. Debido a que la clasificación de colores se hace en terreno, esta es un registro manual subjetivo por parte del agrónomo-inspector encargado del proceso. En algunos casos se suelen tomar fotografías de los frutos en el árbol, pero tan solo como una forma de dejar una evidencia, sin mayor procesamiento de esta.

2.3.2. Control de color de fruta cosechada

El control de color de la fruta recién cosechada es probablemente una de las instancias más importantes del control de calidad de fruta. Sus resultados permiten determinar el futuro comercial que se obtendrá para esta fruta, ya que es de suma importancia para determinar el tipo de procesamiento previo al embalaje y el tipo de embalaje específico que se utilizará dependiendo del mercado objetivo al cual se destinará la fruta.

2.3.3. Control de color en el proceso de embalaje

Para la gran mayoría de la fruta, esta es la instancia en la que están más desarrolladas el control y selección de fruta por color. En la actualidad todas las frutas que pasan por una línea automatizada de proceso y embalaje son analizadas mediante el uso de fotografía ya sea para su clasificación por color o para determinar algunos defectos presentes.

Es necesario hacer presente que también hay algunas especies de fruta que no pasan por una línea automatizada y su embalaje es completamente manual. Entre estas frutas se encuentran las uvas, piñas, melones, sandías y plátanos.

2.3.4. Control de Color de fruta embalada

Este tipo de control de calidad y la clasificación de color de la fruta se hace en instancias posteriores al proceso de embalaje, tales como control de calidad de fruta previo al embarque, control de calidad en la recepción en destino de la fruta y control de calidad de "fruta testigo".

2.4. Sobre el uso de fotografía en la clasificación de fruta

El uso de las cámaras fotográficas en la clasificación de frutas tiene varios años, ya en 1986 la empresa TNT srl (Tomorrow New Technology) aplico por primera vez sistemas de cámaras para la selección del calibre y color en la fruta. Estos sistemas estaban integrados directamente en las calibradoras electrónicas y permitían una selección automática de acuerdo al color y tamaño de cada fruto.

Empresas como Unitec (Italia) y MAF (Francia) dominaron el mercado de los calibradores electrónicos durante un buen tiempo, los cuales eran utilizadas principalmente en manzanas y carozos.

Las cámaras fotográficas se utilizan para capturar imágenes de las frutas a medida que se mueven a través de la línea de producción de fruta fresca. El número de cámaras utilizadas en una línea de producción de frutas puede variar dependiendo del tamaño y la complejidad de la línea. En general, las cámaras se instalan en una parte específica de la línea llamada "estación de visión". La estación de visión es un componente clave de los sistemas de clasificación y selección electrónicos, y suele incluir una o varias cámaras de alta resolución que están dispuestas para capturar imágenes de las frutas desde diferentes ángulos. Las cámaras están diseñadas para funcionar en tiempo real y en alta definición. Las estaciones de visión también pueden incluir iluminación especializada para asegurar que las imágenes sean claras y nítidas, y pueden ser ajustadas para adaptarse a diferentes tipos de frutas. Además, pueden contar con sistemas de limpieza para mantener las cámaras limpias y en óptimas condiciones de funcionamiento.

La información recopilada por las cámaras se procesa en tiempo real mediante software es-

pecializado que utiliza algoritmos de visión por computadora y machine learning automático para detectar y clasificar las frutas en función de varios criterios, como tamaño, color, forma, defectos, entre otros. Este procesamiento en tiempo real se realiza usando sistemas informáticos especializados de alta capacidad que puede procesar grandes cantidades de datos en tiempo real. Una vez que se analizan las imágenes, el sistema clasifica la fruta en diferentes categorías y envía el resultado en segundos al calibrador electrónico que las separa automáticamente. El tamaño y color determinan el punto de salida (clasificación) en la línea y la forma y defectos determinan si es apta para destino exportación, mercado interno o desecho.

El paso por estas líneas de calibración electrónica tiene un costo importante, lo ideal es que un gran porcentaje de la fruta que se procesa en ellas tenga como destino la exportación (porcentaje de exportación del lote), no resulta rentable procesar un lote con bajo porcentaje de exportación por estas sofisticadas líneas, y por eso es muy importante estimar con anterioridad al proceso, cuál será el rendimiento de exportación del lote recibido. Es ahí donde adquiere importancia el control de calidad al momento de la recepción de la fruta, ya que es el que determina con anticipación una estimación del rendimiento de exportación del lote.

Las calibradoras electrónicas también han incorporado sofisticados softwares que utilizan machine learning para la detección de imperfecciones en cada fruto y algún grado de clasificación de defectos.

Hay varias empresas de software que han desarrollado este tipo de sistemas para la clasificación de frutas (fruit grading) tales como Ellips (www.ellips.com), Ingivision (ingivision.com), Futura (futura-grading.com) pero todas ellas dedicadas principalmente al uso en clasificadores electrónicos (fruit sorter) en líneas de proceso.

Además de estas grandes operaciones que involucran varias cámaras existen algunas empresas que han desarrollado sistemas de análisis de frutas en aplicaciones móviles para el control de calidad.

Clarifruit es una empresa de origen israelí, que ha avanzado bastante en el desarrollo de una aplicación que utiliza fotografía para clasificar colores y frutas. Principalmente se utiliza en el campo y es utilizada para el control del proceso de maduración y desarrollo de la fruta. En la actualidad tiene muchos avances la aplicación para usarla en uva de mesa en la cual fotografían los racimos y el sistema les analiza el color y tamaño de bayas. No tienen desarrollos que utilicen fotografías de frutas a granel, siempre es en base a fotografía de frutos o racimos separados.

Otra empresa es Hectre, que ha desarrollado una aplicación enfocada en la detección de calibre de frutas en base a fotos tomadas desde un dispositivo móvil.

2.4.1. Aprendizaje profundo en el procesamiento de fotografías para frutas

Si bien existen diversas técnicas como potenciales soluciones, entre ellas aprendizaje profundo, las técnicas basadas en Deep learning no fueron agregadas por dos razones. En primera

instancia es porque estas requieren una base de datos con una gran cantidad de imágenes, lo cual no se tiene en este caso. Segundo es que las publicaciones relacionadas a la fruta con Deep learning se enfocan detección y clasificación de frutas (determinar qué tipo de fruta es), además de identificación de defectos [3], [4].

2.5. Plataforma QCForms para el control de calidad de frutas

QCforms [1] es una empresa chilena desarrolladora de software para el control de calidad de frutas. El software desarrollado por QCforms es utilizado en 21 países entre Norte América, Centro América, Sudamérica y Europa. El software desarrollado por QCforms permite digitalizar el proceso de control de calidad, automatizar la recolección de datos y entregar a los clientes las herramientas para tomar decisiones basadas en la calidad de los productos en tiempo real y realizar un análisis evolutivo de los datos recolectados.

QCforms tiene dos componentes, una plataforma web y una aplicación móvil. La plataforma web permite el manejo y análisis de los datos recopilados, para el uso de los administradores y operadores de gestión. La aplicación móvil se usa para la recopilación de datos durante la inspección de calidad, la App es usada por los inspectores para evaluar los criterios de calidad establecidos e ingresar los datos en la aplicación. Esta aplicación es configurada de acuerdo con la necesidad de cada cliente, donde pueden acceder a un formulario con los campos de medición y registros de calidad y condición que se necesitan. Uno de los campos más determinantes en la evaluación de la apariencia y calidad de la fruta es el color.

La gran mayoría de las especies de frutas tienen plantillas de clasificación de colores, algunas son de uso general según normas internacionales y otras se han creado en Chile para una estandarización en la clasificación de la fruta de exportación.

El uso de plantillas de clasificación de colores es parte del proceso de capacitación de los inspectores de calidad de fruta, de tal manera que puedan, mediante comparación con la plantilla, hacer la clasificación del color individual de cada uno de los frutos de muestra. Esta asignación es realizada de forma manual de acuerdo a la inspección visual.

Algunas plantillas de clasificación de colores de fruta se hacen para la especie en general, pero existen además plantillas específicas para algunas variedades de fruta que toman en cuenta el color patrón de la variedad y consideran las posibles variaciones de acuerdo con el nivel de maduración (ej. paltas, cítricos).

En las Figuras 2.1 y 2.2 se ve la interfaz de la aplicación móvil de QCForms. Al ingresar a la aplicación se despliega un dashboard, Figura 2.1.a con información sobre el contenido de la base de datos interna y otra información relevante al usuario.

Para revisar e ingresar información primero se selecciona el menú principal, Figura 2.1.b, y con esto se accede al despliegue de información sobre días de inspección y los embarques ingresados existentes, Figura 2.1.c. Al seleccionar uno de estos embarques se visualiza la información en detalle del embarque junto con la lista de lotes asociados a este, Figura 2.1.d.

Un lote corresponde a una separación dentro del embarque que identifica el tipo de producto.

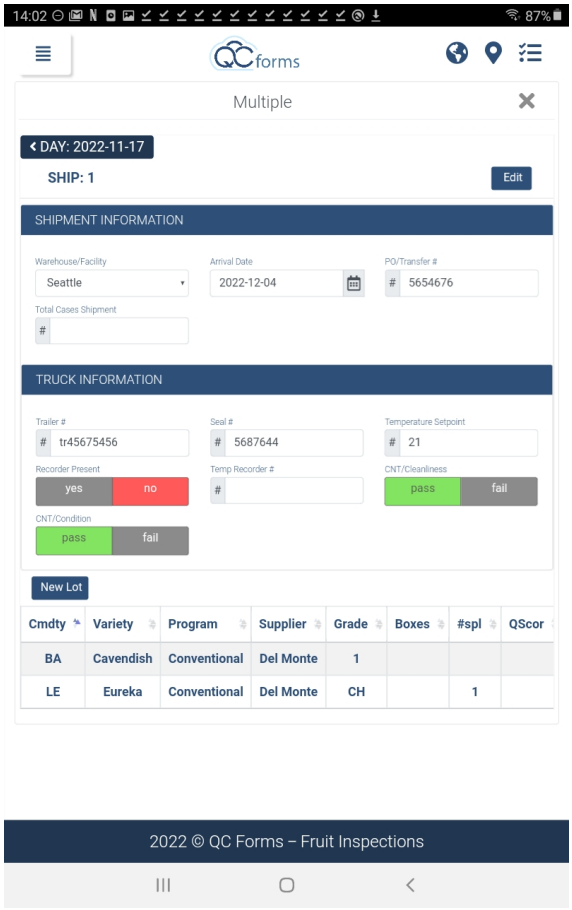
Al seleccionar un lote, o crear uno nuevo, se despliega el formulario con la información en detalle de este junto con las muestras de inspección asociadas, Figura 2.2.a. Finalmente al seleccionar una muestra de inspección o crear una nueva, se llega al formulario de la muestra desplegado en las Figuras 2.2.b, 2.2.c y 2.2.d.

El formulario de la muestra de inspección contiene distintas secciones de acuerdo al tipo de información a ingresar, entre ellas está la clasificación de color de los frutos, lo cual se puede observar en las Figuras 2.2.b, 2.2.c y 2.2.d. En esta sección existen campos para ingresar el número de frutos en cada color de clasificación. Esto se puede realizar escribiendo el número en el teclado del dispositivo móvil o utilizando los botones + y -.

(a) Dashboard

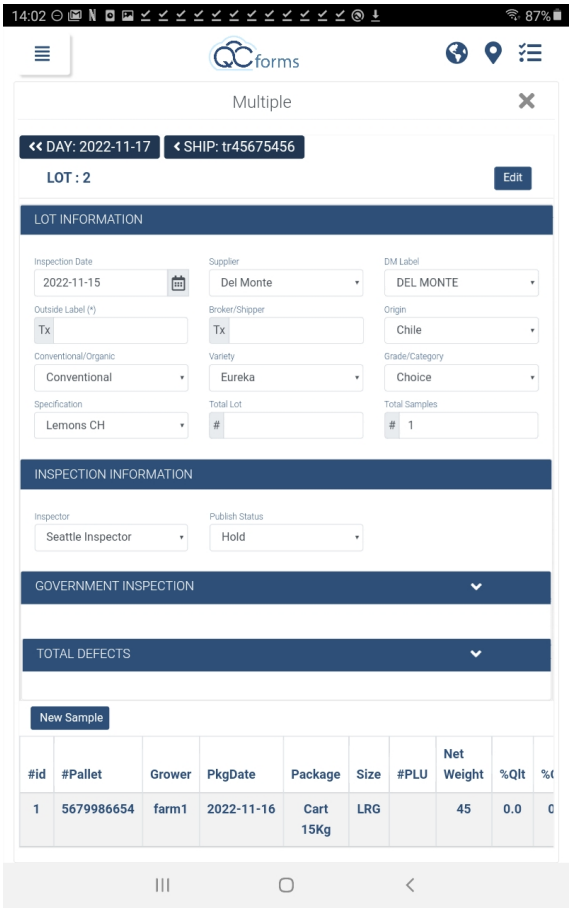
(b) Menu principal

(c) Fecha y listado de embarques

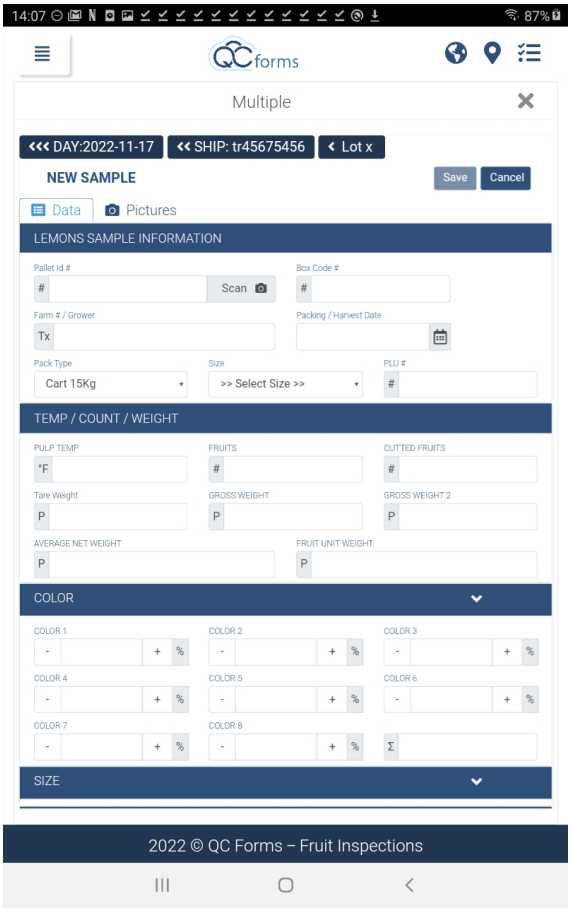


(d) Embarque y listado de lotes

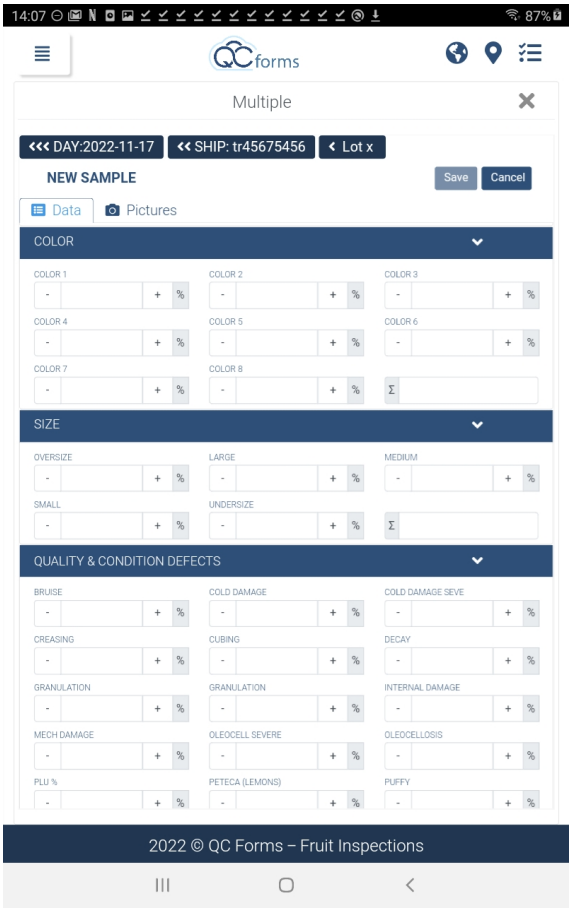
Figura 2.1: App QCForms



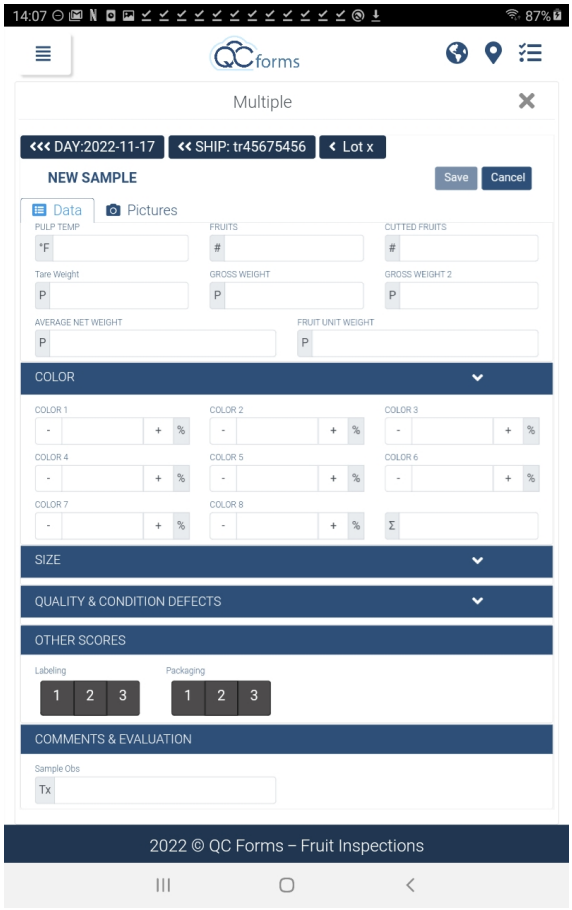
(a) Lote y listado de muestras



(b) Formulario muestra



(c) Formulario muestra



(d) Formulario muestra

Figura 2.2: App QCForms

2.6. Algoritmos de detección

La detección de objetos ha tenido un rápido cambio en el campo de la visión artificial. Su participación en la combinación de clasificación de objetos y localización de objetos lo convierte en uno de los temas más desafiantes en el dominio de la visión artificial. La detección y el reconocimiento de objetos están estrictamente conectados, hasta cierto punto, los dominios de los dos pueden verse como una clasificación de patrones. La detección de objetos se refiere a responder a la pregunta sobre si un tipo de objeto determinado está presente en las imágenes. A veces, su apariencia y posición actuales también son de interés. Por otro lado, el objetivo del reconocimiento de objetos es identificar su tipo particular [2].

Un ejemplo de algoritmo de detección de objetos es "template matching". Este método utiliza una imagen de referencia conocida como plantilla y la desliza por toda la imagen de origen un píxel a la vez para determinar los objetos más similares. Producirá otra imagen o matriz donde los valores de píxel corresponden a la similitud de nuestra plantilla con la imagen de origen. Por lo tanto, cuando intentamos ver la imagen de salida, los valores de píxel de los objetos coincidentes alcanzarán su punto máximo o se resaltarán.

Varios algoritmos se basan en una combinación de pasos de pre-procesamiento, aplicación de algoritmos de límites, búsqueda de contornos, aplicación de máscaras, entre otros. En la siguiente sección describiremos algunos de los pasos comunes que incluyen estos algoritmos y que posteriormente serán utilizados para construir nuestro propio algoritmo de detección de frutas.

2.7. Algoritmos de procesamiento de imágenes

En esta sección se describen de forma general una selección de algoritmos de procesamiento de imágenes que posteriormente serán utilizados para construir nuestro propio algoritmo de detección de imágenes.

2.7.1. Suavizado o Blur

Filtros de suavizado son usados para difuminar y reducir el ruido en una imagen. Estos filtros son usados en tareas de preprocesamiento, tales como eliminar detalles de una imagen previo a la detección de objetos, reducir el ruido y realizar conexiones en pequeños espacios entre líneas o curvas. Estas tareas pueden ser realizadas con filtros lineales así como también con filtros no lineales [5].

Los filtros de suavizado consisten en reemplazar el valor de cada pixel con una función de los pixeles vecinos, incluyendo sí mismo. A esta función le llamamos kernel, y cuando se menciona el tamaño del kernel nos referimos al tamaño de la vecindad a considerar.

Como se mencionó anteriormente, existen los filtros lineales y no lineales. Un ejemplo de filtro lineal sería un filtro de promediado, donde la función o kernel que se utiliza promedia los pixeles vecinos con la misma ponderación. Un ejemplo de filtro no lineal sería un filtro

gaussiano donde los pixeles más lejanos tienen una menor ponderación en el resultado final del pixel que se está modificando. La ponderación o peso que se aplica se calcula de acuerdo a una función gaussiana [6]. Usualmente los filtros gaussianos se utilizan para reducir el ruido de la imagen así como también los detalles. El suavizado gaussiano también se usa como una etapa de preprocesamiento en los algoritmos de visión por computadora para mejorar las estructuras de la imagen en diferentes escalas. El efecto de un filtro gaussiano se puede observar en la Figura 2.3.

(a) imagen inicial (b) Gaussian Blur

Figura 2.3: Suavizado Gaussiano

2.7.2. Operaciones aritméticas bit a bit

Operaciones bit a bit, o "bitwise" se pueden usar en la manipulación de imágenes y nos permiten mejorar muchos aspectos de estas. Estas técnicas "bitwise" se utilizan en muchas aplicaciones de visión por computadora, como para crear máscaras de la imagen, agregar marcas de agua a la imagen y siendo posible crear una nueva imagen usando estos operadores bit a bit. Estas operaciones funcionan en los píxeles individuales de la imagen para brindar resultados precisos en comparación con otras técnicas de transformación. Esta característica nos permite filtrar la parte de la imagen que es relevante para nosotros.

Bitwise AND Esta función calcula la conjunción de píxeles de dos imágenes. Esta operación solo considera los píxeles que son comunes a ambas imágenes y los píxeles restantes se eliminan de la imagen de salida. Al usar la función AND solo quedan las regiones intersectadas en ambas imágenes.

Bitwise OR Esta función calcula la disyunción de los píxeles en ambas imágenes. Aquí realizamos un producto de la matriz "element-wise", no eliminaremos píxeles, sino que fusionará ambas imágenes.

Bitwise NOT Esta función invierte cada bit de una matriz. Reemplaza los píxeles blancos con píxeles negros y viceversa. Esta operación solo se puede realizar en una imagen

Bitwise XOR Esta función invierte los píxeles que se cruzan en dos imágenes y el resto de los píxeles permanece igual.

2.7.3. Transformación de distancia

La transformación de distancia es un operador que convierte imágenes binarias (pixeles blanco y negro) a imágenes donde a cada pixel asigna un valor de gris indicando la distancia desde

ese pixel al pixel con valor 0 más cercano [9].
En la Figura 2.4 se ve un ejemplo de la transformación de distancia aplicada a una imagen binaria.

(a) Imagen inicial (b) Transformación distancia

Figura 2.4: Ejemplo resultado transformación de distancia

2.7.4. Método de valor umbral o thresholding

"Thresholding" es una de las técnicas de segmentación más comunes en visión por computadora y nos permite separar el primer plano (es decir, los objetos que nos interesan) del fondo de la imagen. El método de "Thresholding" es la binarización de una imagen. En general, buscamos convertir una imagen en escala de grises en una imagen binaria, donde los píxeles son 0 o 255. [6], [8]

Un ejemplo simple de umbralización sería seleccionar un valor de umbral T y luego establecer todas las intensidades de píxeles inferiores a T en 0 y todos los valores de píxeles superiores a T en 255. De esta manera, podemos crear una representación binaria de la imagen.

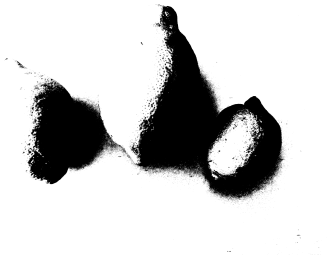
Para este método existen diversas formas, entre ellas las más comunes son:

1. Seleccionar un umbral simple donde proporcionamos parámetros manualmente para segmentar la imagen, esto funciona muy bien en condiciones de iluminación controlada donde podemos garantizar un alto contraste entre el primer plano y el fondo de la imagen.

2. El método de umbral de Otsu que calcula automáticamente el valor de umbral óptimo en función de la imagen de entrada, separando cada pixel en dos clases, primer plano y background. El valor del threshold se determina minimizando la varianza dentro de la misma clase [6], [7].

3. Thresholding adaptativo que, en lugar de tratar de encontrar un umbral para la imagen globalmente utilizando un valor único, divide la imagen en partes más pequeñas y encuentra el umbral cada una de estas piezas por separado e individualmente [6], [8].

(a) imagen inicial



(b) imagen threshold

Figura 2.5: Thresholding

2.7.5. Búsqueda de contornos

La búsqueda de contorno es una herramienta útil para la segmentación de imágenes, el análisis de formas y la detección y reconocimiento de objetos. La detección de bordes ha evolucionado rápidamente en las últimas décadas. Dos métodos principales pueden identificarse, primero está la comparación de plantillas (template matcing), un segundo método es la diferenciación de gradiente. En ambos casos el objetivo es buscar dónde la magnitud de intensidad de gradiente es suficientemente grande como para ser considerado un indicador de borde de un objeto [8]. Esto es asumiendo que la intensidad entre el pixel del borde del objeto y el fondo tienen suficiente diferencia. La diferencia entre ellos es cómo proceden a estimar la diferencia de gradiente local [8].

Para una de las etapas de búsqueda de contornos en este trabajo se utiliza la función findCountour de la librería openCV, la cual obtiene los contornos de una imagen binaria utilizando el algoritmo de Suzuki [10]. La idea principal del algoritmo de Suzuki es iterar sobre todos los pixeles de una imagen buscando pixeles pertenecientes a un borde previamente no explorado. Cuando tal pixel de borde es encontrado, el algoritmo de Suzuki provee un mecanismo para seguir este borde hasta que sea completamente encontrado.

En la Figura 2.6 se ve un ejemplo de la búsqueda de contornos luego de aplicar una función de threshold.

(a) Imagen inicial

(b) Contornos marcados

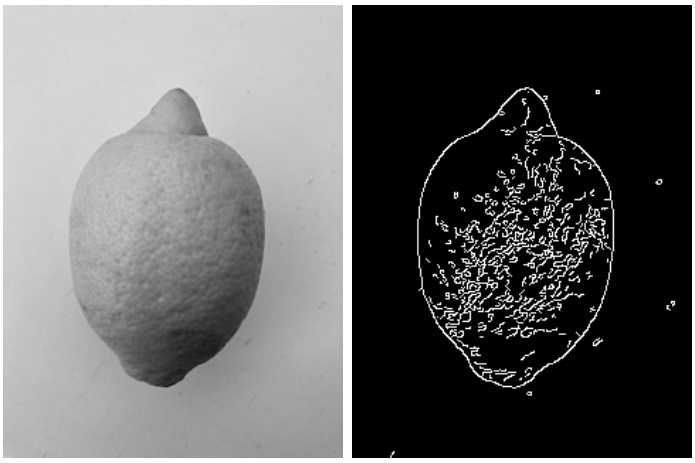
Figura 2.6: Ejemplo resultado de búsqueda de contornos

2.7.6. Algoritmo de Canny

El detector Canny es un algoritmo de detección de bordes que consiste en varias etapas para detectar una amplia gama de bordes en las imágenes. El operador de Canny primero suaviza la intensidad de una imagen y luego produce los segmentos de contornos extendidos al seguir las magnitudes altas de gradiente en la intensidad [6].

El operador de Canny produce fragmentos de contornos de la imagen y es controlado por un solo parámetro de suavizado. La imagen primero es suavizada por un filtro Gaussiano y luego se calculan la magnitud de gradiente y la dirección de este en cada pixel de la imagen. La dirección del gradiente es usada para adelgazar los bordes y eliminar cualquier pixel cuya intensidad de gradiente que no sea mayor que los pixeles vecinos en cualquiera de las posiciones a los lados de la dirección del gradiente. Con esto los contornos de alta magnitud son rastreados [6].

En la Figura 2.7 se ve un ejemplo del algoritmo de Canny.



(a) Imagen inicial (b) Canny

Figura 2.7: Ejemplo resultado algoritmo de Canny

2.8. Índice de usabilidad de software

Cada vez hay más publicaciones en la literatura que abordan el problema de cómo medir la usabilidad del software [20]. Estos proponen varios estándares o modelos diferentes para cuantificar y evaluar la usabilidad dentro de las comunidades de interacción humano-computadora (HCI) y de ingeniería de software (SE). La usabilidad generalmente se mide utilizando una serie de métricas observables y cuantificables.

El índice de usabilidad es una métrica que permite cuantificar la usabilidad de cualquier sistema. Las métricas son muy útiles cuando se trata de cuantificar la usabilidad durante la evaluación de la usabilidad de software, sitios web y aplicaciones. Las pruebas de usabilidad muestran qué tan bien una solución satisface las necesidades de las personas [21].

Aunque medir la usabilidad puede ser más costoso que realizar estudios cualitativos, vale

la pena gastar en métricas, dado que las métricas pueden ayudar a rastrear el progreso del diseño y respaldar las decisiones sobre cuándo lanzar un producto [23].

La norma ISO 9241-11 define la usabilidad como "la medida en que un producto puede ser utilizado por usuarios específicos para lograr objetivos concretos con eficacia, eficiencia y satisfacción en un contexto de uso específico" [22].

La norma ISO/IEC 9126-4 recomienda que las métricas de usabilidad incluyan:

Efectividad La precisión y exhaustividad con la que los usuarios logran objetivos específicos

Eficiencia Los recursos gastados en relación con la precisión y exhaustividad con la que los usuarios

Satisfacción La comodidad y aceptabilidad del uso.

Independiente de lo anterior, las formas reales de cómo estos aspectos deben medirse a menudo se dejan a discreción del evaluador. Normalmente, la usabilidad se mide en relación con el rendimiento de los usuarios en un conjunto determinado de tareas de prueba. Las medidas más básicas se basan en la definición de usabilidad como métricas de calidad, entre estas están la tasa de éxito (si los usuarios pueden realizar una tarea), el tiempo que requiere una tarea, la tasa de error y la satisfacción subjetiva de los usuarios [23].

2.8.1. Métricas de usabilidad para la efectividad

2.8.1.1. Tasa de cumplimiento "completition rate"

El éxito de la tarea mide la eficacia y responde a la pregunta ¿hasta qué punto pueden los usuarios completar con éxito una tarea determinada? La comprensión de los usuarios sobre cómo llegar a la solución y qué acciones tomar de principio a fin, son claros signos sobre la usabilidad del software. La eficacia se puede calcular midiendo la tasa de finalización. La tasa de finalización se calcula asignando un valor binario de "1" si el participante de la prueba logra completar una tarea y "0" si no lo hace. Luego se divide la suma de tareas completadas con éxito por el número total de tareas realizadas.

2.8.1.2. Número de errores

Otra medida consiste en contar el número de errores que comete el usuario al intentar completar una tarea. Los errores pueden ser acciones no intencionadas, errores u omisiones que comete un usuario al intentar una tarea. Es útil registrar cada instancia de un error junto con una descripción de este. Identificar las tareas con más errores permite mejorar el diseño y eliminar los elementos que llevan a los errores.

2.8.2. Métricas de usabilidad para la eficiencia

La eficiencia se mide en términos de tiempo de ejecución de una tarea, es decir, el tiempo que tarda el participante en completar con éxito una tarea. Si algunos usuarios tardan más tiempo en realizar las tareas, se puede analizar qué aspectos de la solución son más difíciles de comprender y posiblemente no son intuitivos o no están abordados correctamente en la documentación

2.8.3. Métricas de Usabilidad para Satisfacción

La satisfacción del usuario se mide a través de cuestionarios de satisfacción estandarizados que se pueden administrar después de cada tarea y/o después de la sesión de prueba de usabilidad.

2.8.3.1. Satisfacción del nivel de tarea

Después de que los usuarios realizan una tarea, independiente de si esta es exitosa o no, se les entrega inmediatamente un cuestionario para medir qué tan difícil fue esa tarea, este puede consistir en varias o una sola pregunta. El objetivo es recopilar información sobre la dificultad de la tarea desde la perfectiva de los usuarios. Existen varios cuestionarios populares para estos, el más usado para este caso es el SEQ (Single Ease Question), que consiste en una sola pregunta [25].

2.8.3.2. Satisfacción del nivel de prueba

La satisfacción del nivel de prueba se mide dando un cuestionario a cada usuario al final de la sesión, una vez que han realizado todas las tareas. Esto permite medir la impresión que tienen sobre la facilidad de uso general del sistema que se está probando. Para ello se pueden utilizar diferentes cuestionarios, entre ellos el más popular es el SUS (System Usability Scale), que consiste en 10 preguntas [26].

Capítulo 3

Metodología

3.1. Solución propuesta

Actualmente los inspectores realizan la clasificación de colores de acuerdo con una plantilla de clasificación que pueden ver en todo momento de forma física (impresa), como la que se muestra en la Figura 4.5.

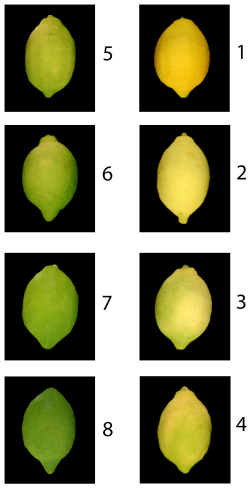


Figura 3.1: Planilla de colores

La solución que proponemos incorpora esta plantilla impresa en las imágenes a procesar, específicamente, una foto de esta planilla debe ser tomada en las mismas condiciones de luz que la foto del bin de limones, dado que las condiciones de luz pueden variar en los distintos lugares de inspección, y por lo tanto los colores capturados con la cámara pueden diferir de la plantilla en forma digital.

La función a implementar hará uso de las plantillas de clasificación de colores que tienen físicamente los inspectores. Al capturar la imagen de la plantilla en las mismas condiciones en las que se tomarán las imágenes de los bins de limones (ajustes de la cámara y condición de iluminación), se podrá realizar una comparación efectiva entre los patrones y los frutos. Esta solución implica que nuestro algoritmo debe identificar cada uno de las clases de colores para clasificación en la plantilla de colores, para luego continuar con el procesamiento de las

imágenes de frutas a evaluar.

El procedimiento consistirá en tomar dos imágenes, una del bin de limones y otra imagen de la plantilla de colores. Luego se deben realizar tres procesos, uno que identifique las distintas clasificaciones y sus valores específicos de comparación, un segundo proceso que identifique los limones en el bin y finalmente un tercer proceso que asigne un color de clasificación a cada uno de los limones detectados en el bin. Además, se considera que no es necesario detectar cada limón en el bin y una muestra de al menos 100 frutos es considerada como representativa del bin completo.

La modalidad propuesta tiene varias ventajas a considerar:

1. No requiere de un proceso de calibración, ajuste o corrección según las condiciones de la toma de imágenes, ya que tanto el patrón de comparación y los frutos se les fotografía en las mismas condiciones.

2. Permite una personalización de la clasificación de colores mediante la construcción propia del patrón de colores a utilizar.

3. Cambia el paradigma actual de clasificación de colores al introducir un método de medición real en contraste con la evaluación subjetiva (depende de la experiencia y capacidad visual) realizada por el inspector de fruta.

En resumen, esta nueva funcionalidad debe primero identificar la plantilla y las clases definidas. Luego en la imagen de las frutas a inspeccionar, las debe identificar por separado y clasificar cada una de acuerdo con el color que corresponde.

3.1.1. Aspectos técnicos de la solución propuesta

La aplicación actual de QCforms es una aplicación híbrida para Android, realizada con Cordova [11]. Esto quiere decir que la aplicación está mayormente desarrollada en javascript y Html, usando plugins que incorporan funcionalidades para las cuales estos lenguajes no tienen soporte. Por ejemplo, para la base de datos se utiliza un plugin que permite crear una base de datos local en sqlite [12]. Para el uso de la cámara [13], el sistema de archivos [14], entre otros, se utilizan plugins que implementan estas funcionalidades de forma nativa en Android y proveen una API para utilizarlas desde Javascript.

Para el trabajo propuesto se consideraron varias opciones, entre ellas estaba hacer un plugin para Cordova que permita el procesamiento de imágenes que se necesita, desarrollando las funcionalidades de procesamiento de imágenes para que funcionen de forma nativa en Android y luego crear una API para ser utilizadas en Javascript. Luego de varios experimentos se optó por seguir otro camino, este es usar directamente la librería OpenCV escrita para javascrip para el procesamiento de imágenes. Este último método tiene la dificultad de que la librería opencv.js escrita específicamente para javascript, cuenta con una documentación incompleta y pocos tutoriales. Además, varias de las funciones que se encuentran en OpenCV para C++ y Python, no se encuentran implementadas para javascript, por lo cual deben ser escritas desde su base. Sin embargo, se ha determinado que este es el mejor camino a seguir, especialmente dado los conocimientos técnicos previos y la facilidad que existe para testear el código.

3.2. Evaluación de la solución

Para realizar las pruebas en la etapa de desarrollo, primero se utilizarán imágenes de inspecciones pasadas con la información de clasificación de colores ingresada por los inspectores. Para esto se utilizarán las imágenes descritas en la sección 3.2.1.

Dada que una de las restricciones para clasificar colores correctamente es que las fotos sean tomadas desde arriba de la mesa de inspección (de esta manera todos los frutos se muestran con el mismo ángulo), de inspecciones pasadas tenemos aproximadamente 50 fotos de bins ya clasificadas que podemos usar. La desventaja que tenemos es que estas fotos de bins no están acompañadas de las respectivas fotos de la plantilla de colores, por lo cual se utilizará una imagen de la plantilla de colores tomada en la planta de inspección donde se tomaron las fotos de los bins, donde las condiciones de luz son en teoría las mismas, pero no hay certeza de aquello.

En la etapa final de desarrollo y luego para evaluar el éxito del proyecto, la aplicación con la nueva funcionalidad será puesta en producción en una de las ubicaciones donde se realizan inspecciones con QCforms. Específicamente, la aplicación será evaluada en dos aspectos. Primero será evaluada la eficacia de la clasificación de colores, para lo cual se realizará la clasificación automática desarrollada además de la clasificación manual, las cuales serán comparadas. En segundo lugar, será evaluada la usabilidad de la nueva función desarrollada para la aplicación siguiendo la metodología descrita en la sección 3.4.

3.2.1. Base de Datos

La base de datos para "testing" durante la etapa de desarrollo consiste en 50 imágenes de bins de limones tomadas en inspecciones pasadas, junto con la clasificación de color de 100 frutos aleatorios de dicho bin.

En la Figura 3.2 se muestra una de estas imágenes. La tabla 3.1 indica la clasificación de 100 frutos aleatorios tomados del bin de la Figura 3.2 de acuerdo a la tabla de colores en la Figura 4.5.

Figura 3.2: Bin de limones

Tabla 3.1: Clasificación de colores para Figura 3.2

color	c1	c2	c3	c4	c5	c6	c7	c8	c9	c10
#	0	0	4	31	37	15	10	3	0	0

3.3. Evaluación del algoritmo

La evaluación del algoritmo se hará en dos etapas. En primera instancia se utilizará la base de datos de "testing" descrita en la sección anterior. La desventaja de esta base de datos es que no se cuenta con la imagen de la carta de colores tomada junto con la imagen del bin a clasificar. Para esta etapa se utilizará la imagen de la carta de colores tomada en la planta donde se inspecciona, pero no necesariamente las condiciones de luz son las mismas.

Una segunda etapa de evaluación del algoritmo será realizada cuando la aplicación esté lista para ser puesta en producción. En esta instancia 4 inspectores utilizarán la aplicación para

hacer la clasificación de colores automática y posteriormente realizarán la clasificación de colores por inspección visual. Ambos resultados serán comparados.

3.4. Evaluación de usabilidad

Para la evaluación de usabilidad de la aplicación de QCForms con la nueva funcionalidad de clasificación de color automática será evaluada por 4 inspectores. Para realizar la evaluación de adoptará la metodología descrita en las secciones **??** y 2.8, esto es, una evaluación basada en tutoriales donde cada inspector debe seguir la tarea de utilizar la aplicación para realizar la clasificación automática de colores. Esta tarea será evaluada de acuerdo a preguntas que deben responder los inspectores además del éxito con el que cumplen las tareas y el tiempo que toman en realizarlas.

En primera instancia se realizará un tutorial para los usuarios en el cual se les explicará cómo utilizar la función de clasificación de color automático, posteriormente se les pedirá que utilicen esta función. Una vez que terminen la tarea asignada se tomarán las siguientes medidas por cada inspector:

Tasa de cumplimiento Se asignará el valor 1 si el inspector de la prueba logra completar la prueba y 0 si no lo hace.

Número de errores Se cuenta el número de errores que comete el inspector.

Tiempo de ejecución Se mide el tiempo que toma el usuario en completar la tarea.

Además de tomar las medidas anteriores se les preguntará a los usuarios respecto a la facilidad que fue realizar la tarea, independiente de si fue exitosa o no. Para esto el usuario debe asignar una nota de 1 a 7, donde 1 indica que fue "muy difícil" y 7 indica "muy fácil". Finalmente se le pedirá a los inspectores realizar observaciones respecto a la tarea asignada.

Capítulo 4

Algoritmo Desarrollado

4.1. Algoritmo de detección de frutas

A continuación, se describe el algoritmo desarrollado para la detección de limones, el cual ejemplificamos en la detección sobre la imagen que se muestra en la Figura 4.1. Un diagrama de los pasos involucrados en la detección de limones se ilustra en la Figura 4.3.

Figura 4.1: Bin de limones

4.1.1. Detección de bordes

Para la detección individual de los limones en la superficie de cada Bin, utilizamos la función para detección de bordes, sin embargo, para una mejor precisión en la detección de objetos sometemos la imagen a una serie de pasos de preprocesamiento. Estos pasos de preprocesamiento permiten reducir la cantidad de información a procesar, dejando sólo la información más relevante de la imagen, que en este caso es los contornos de los limones.

El primer paso en el procesamiento de la imagen para la detección de los limones, consiste en destacar el contorno de los frutos, lo cual se logra mediante un aumento del contraste de la imagen, el resultado se puede observar en la Figura 4.2.a.

A continuación, se transforma el espacio de color de la imagen de RGB a escala de grises, Figura 4.2.b.

El siguiente paso consiste en la aplicación de un algoritmo de difuminado de la imagen usando un filtro Gaussiano, esto permite que los bordes a detectar se limiten a los objetos más grandes y bien definidos, en este caso los limones, y no se confundan con bordes, aristas o texturas distintas en cada detalle, Figura 4.2.c. Para este paso se ajustó el parámetro del tamaño de kernel con el fin de obtener el resultado óptimo.

Luego se aplica el algoritmo Canny para detección de bordes, para un mejor resultado de este, invertimos la imagen al negativo, Figura 4.2.d. Este algoritmo destaca todos los bordes de la imagen, independiente de si estos son continuos o formas cerradas. Este paso permite que la información a ser procesada se reduzca. En este paso se ajustaron los parámetros de threshold y gradiente.

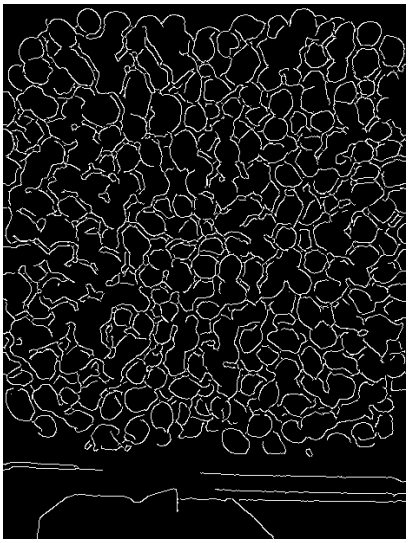
En el siguiente paso aplicamos una transformación de distancia, que en este caso consiste en determinar la distancia de los bordes de la imagen destacados en el paso anterior, al centro de los objetos, Figura 4.2.e. La intensidad de gris que se observa en la imagen corresponde a la distancia entre el borde(blanco) y pixel negro más lejano, el cual sería el centro de los frutos. Este paso destaca los centros de los objetos prominentes.

Luego, aplicamos una función de threshold binario sobre la imagen obtenida con la transformación de distancia. Esta función asigna el color negro a cada pixel que está sobre el valor del threshold utilizado, y blanco si es menor. Determinar el valor del threshold depende del paso siguiente, que consiste en aplicar una función que busca los contornos de la imagen, esto se refiere a contornos individuales. Para determinar el valor del threshold óptimo, iteramos con distintos valores límite, o threshold, y con cada uno de ellos procesamos la función de búsqueda de contornos. El valor de threshold que permite obtener la mayor cantidad de contornos es el que se utiliza. La Figura 4.2.f muestra la imagen obtenida al aplicar el threshold que maximiza la cantidad de contornos.

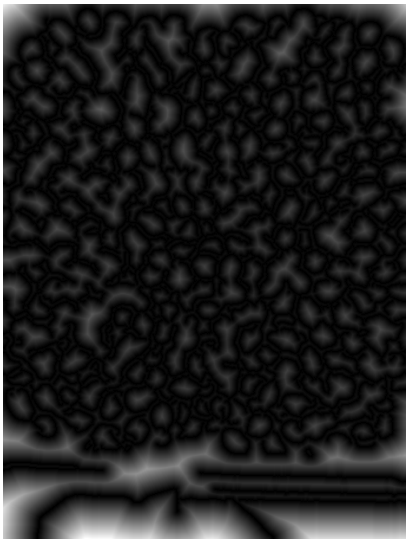
(a) Imagen con incremento del contraste

(b) imagen grayscale

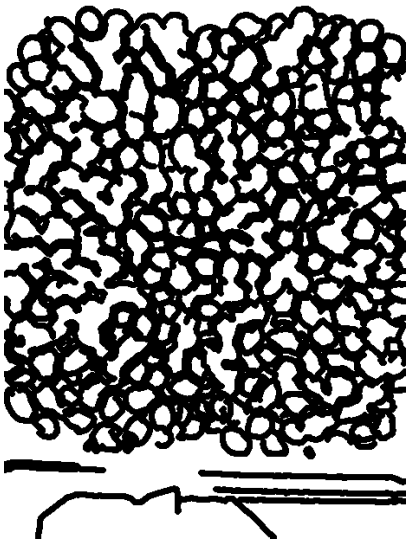
(c) GaussianBlur



(d) Canny detección de bordes



(e) Transformación de distancia



(f) Threshold

Figura 4.2: Detección de Bordes

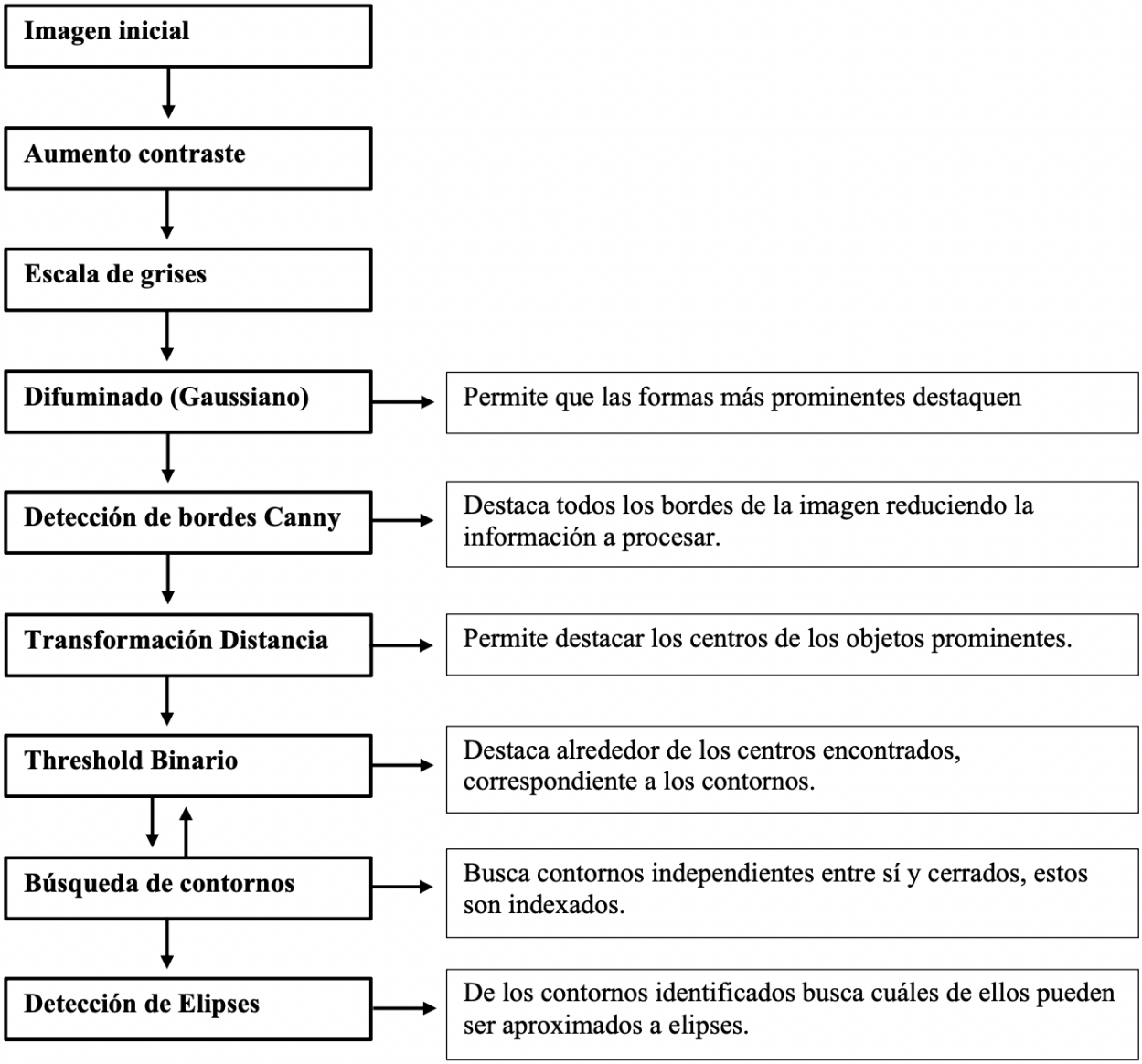


Figura 4.3: Pasos algoritmo de detección

4.1.2. Detección de elipses

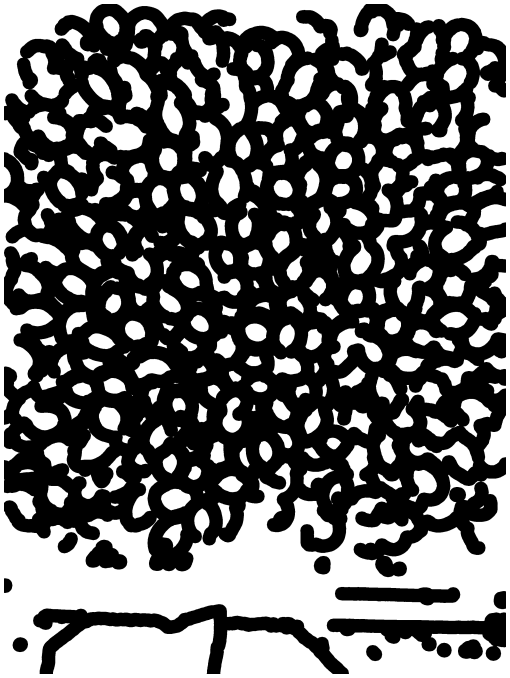
En la Figura 4.4.a se ve la imagen que se obtiene al aplicar la secuencia de pasos hasta el algoritmo de thresholding, cuyo proceso fue descrito en la sección anterior.

Una vez que obtenemos la imagen threshold, le aplicamos la función OpenCV de búsqueda de contornos, esta identifica contornos individuales y cerrados en la imagen, los cuales son indexados, el resultado se puede observar en la Figura 4.4.c.

Para la detección de limones nos basamos en el conocimiento de que estos tienen forma redondeada que puede ser representada mediante elipses, para lo cual iteramos sobre cada contorno encontrado en la etapa anterior intentando hacer calzar una elipse sobre este.

En este paso es primordial aplicar restricciones en la relación que debe existir entre el axis mayor y menor de la elipse, así como también en la relación entre el tamaño de la elipse y la imagen completa, la cual corresponde al tamaño del Bin (asumimos que el tamaño del Bin es siempre el mismo). El resultado final de esta etapa se puede observar en la Figura 4.4.d.

(a) Imagen inicial



(b) Threshold

(c) Detección de contornos

(d) Elipses detectadas

Figura 4.4: Detección de Elipses

4.2. Clasificación de color

La clasificación de color de los limones se hace de acuerdo a una plantilla patrón de colores tal como la que se observa en la Figura 4.5.

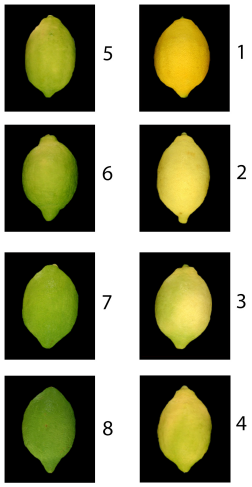
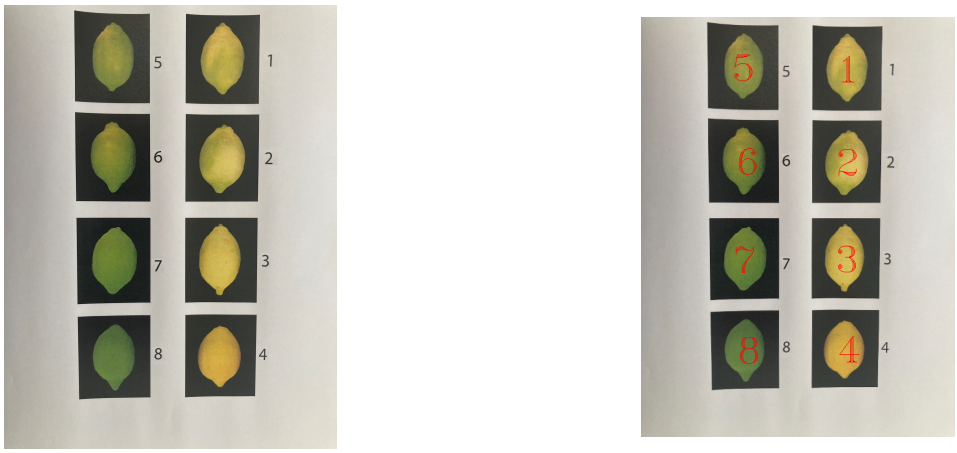


Figura 4.5: Carta de colores para limones

En la solución propuesta en el capitulo 3 se describió que para obtener las clases de color para la clasificación, se tomaría una foto de la plantilla patrón de colores en las mismas condiciones de iluminación que la foto del bin. En la Figura 4.6.a se muestra una foto de la carta de colores impresa.

Una vez obtenida la imagen de la carta de colores, el siguiente paso es segmentar cada uno de los limones, para esto utilizamos de referencia los rectángulos negros que encajonan los limones y así creamos una máscara que elimine el fondo de la imagen para poder separar las distintas cajas encontradas. Finalmente en el proceso enumeramos cada una de las clases encontradas, de esta forma el usuario puede visualizar y asegurarse de que todas las clases de colores fueron detectadas, Figura 4.6.b.

Finalmente, para clasificar los limones de cada Bin, el procedimiento consiste en comparar el valor medio de los pixeles de cada clase detectado en la plantilla patrón de colores con el valor medio de los pixeles en el interior de las elipses detectadas, y se les asigna entonces la clase color más cercana a cada elipse (limón) como perteneciente a esta.



(a) Imagen inicial

(b) Detección de clases

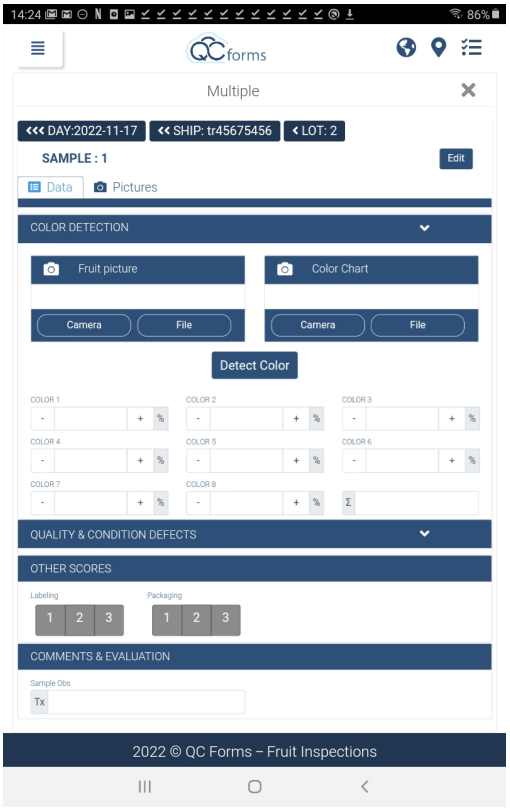
Figura 4.6: Detección de Clases

4.3. Integración a la Aplicación

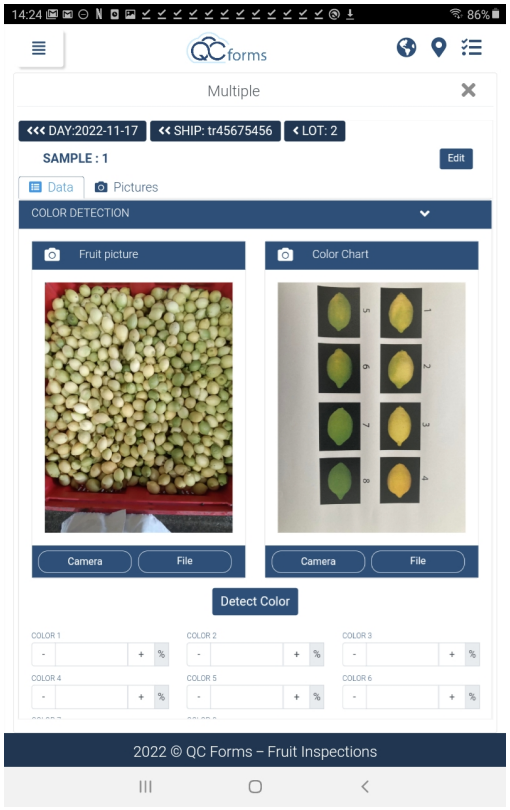
En la Figura 4.7 se muestra la integración de la nueva funcionalidad a la aplicación móvil de QCForms. Anteriormente en la sección de color sólo se contaba con los campos de entradas manuales para cada uno de los colores a clasificar. La integración consiste en que ahora al ingresar al formulario se incorporan dos nuevas celdas para subir fotos, ya sea desde la cámara o desde los archivos, un campo corresponde a la foto del bin de limones y la segunda foto corresponde a la plantilla de colores, Figuras 4.7.a y 4.7.b.

Una vez que se cargan las fotos el usuario las procesa mediante el botón “Detect Color”, tras lo cual se ejecuta el algoritmo de detección y clasificación, el cual tarda menos de dos segundos. Una vez terminado el proceso del algoritmo tenemos dos resultados, primero son las imágenes del bin con los limones detectados marcados, y la imagen de la plantilla de colores con la numeración de color en cada limón de acuerdo al orden en que serán clasificados, Figura 4.7.c. Esto permite al usuario asegurarse de que la detección tanto de los limones en el Bin como la plantilla patrón de colores es adecuado, y de no ser satisfactorio, puede repetir el proceso de tomar las fotos.

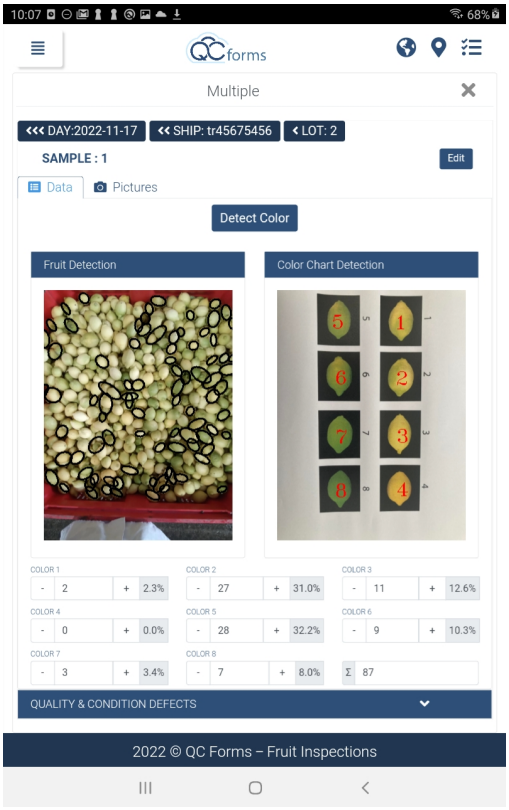
El segundo resultado que se obtiene es la clasificación de cada uno de los limones detectados en una de las clases de color, este resultado se observa en el llenado automático de los campos de entrada de cada color, donde se observa el número de limones clasificados en cada color, a qué porcentaje del total esto corresponde y el número de limones totales detectados (Figura 4.7.c). Una vez que se obtiene este resultado el inspector puede analizar los porcentajes y hacer modificaciones si le parece necesario, como por ejemplo, eliminar los visiblemente fuera de rango.



(a) Layout inicial



(b) Cargar imágenes



(c) Resultados Detección

Figura 4.7: Integración a la APP

Capítulo 5

Evaluación y Análisis de Resultados

En este capítulo analizamos los resultados de los algoritmos de detección de limones, detección de plantilla patrón de color y clasificación de colores.

5.1. Evaluación de los algoritmos

5.1.1. Algoritmo detección de frutas

El algoritmo de detección de frutas, en este caso detección de elipses, fue desarrollado originalmente utilizando OpenCV Python y luego implementado en OpenCV Javascript. Una vez implementado en Javascript se trabajó en un ambiente para debug y desarrollo de pruebas local usando Chrome. Una vez listo el algoritmo, este fue traspasado como aplicación móvil a una tablet. En esta sección revisaremos como los resultados difieren entre la implementación en Python y Javascript, así como también se ve una diferencia en el desempeño del algoritmo entre Chrome local y la aplicación móvil.

En esta sección también se analiza cómo la calidad de la imagen afecta el desempeño del algoritmo, independiente de la plataforma de desarrollo.

En la Figura 5.1 vemos una comparación ejemplo del desempeño del algoritmo de detección de frutas en las distintas plataformas y lenguajes. Para este ejemplo la implementación en Python, Figura 5.1.a, se detectó un gran número de frutos, específicamente 176 limones, además se observa que la mayoría de las detecciones se adhieren a la forma de los limones, contando solo con algunos outliers.

A continuación en la implementación en Javascript y ejecutada en Chrome, Figura 5.1.b, se observa una menor cantidad de frutos detectados, específicamente 132 limones. Además, se observa que, si bien las elipses se adhieren en su mayoría a la forma de los limones, estas sobrepasan su forma, lo cual puede generar problemas cuando se obtiene el valor medio del color.

Finalmente tenemos la implementación del algoritmo en Javascript pero ejecutado en la aplicación móvil, este es el caso en el que tenemos el peor desempeño en cuanto al número de frutos detectados, en el ejemplo que se ve en la Figura 5.1.c se cuenta con sólo 87 limones detectados. En este caso además se observan elipses que abarcan más de un limón, lo cual afecta el cálculo del color promedio para la posterior clasificación.

(a) Python (b) Javascript Chrome (c) Javascript APP

Figura 5.1: Detección de frutas

Considerando el conjunto total de las imágenes de la base de datos descrita en 3.2.1, la implementación en Python reporta un número de limones detectados promedio de 147 con una desviación estándar de 54.9. La implementación en Javascript ejecutada en Chrome encuentra un promedio de frutos de 79.7 con una desviación estándar de 42.8 y la implementación en Javascript con ejecución en la aplicación móvil encuentra un promedio de limones de 69.8 con una desviación estándar de 42.1.

Lenguaje/Plataforma	Promedio de Limones	Desviación Estándar
Pyhton	147	54.9
Javascript/Chrome	79.7	42.8
Javascript/APP	69.8	42.1

La diferencia en el desempeño del algoritmo entre Python y Javascript, se puede explicar por dos razones. En primer lugar, tenemos que si bien las implementaciones son similares, no son exactamente iguales, dado que no todas las funciones de OpenCV disponibles para Python están disponibles para javascript. En particular la búsqueda de elipses a partir de contornos está implementado de forma distinta. Una segunda razón, es que la librería de OpenCV para Python tiene versiones más nuevas que la librería para Javascript, y es mantenida más regularmente, por lo que las funciones en Python, como búsqueda de contornos, tienen un mejor desempeño que en Javascript. Además de un mejor desempeño, se observa que las distintas implementaciones generan resultados ligeramente distintos en funciones básicas como filtros de suavizado.

Del conjunto de imágenes de la base de datos, se observa que el algoritmo de búsqueda de frutos implementado en Javascript es más sensible a la calidad de la foto que el algoritmo implementado en Python, en particular las imágenes borrosas tienen muy mal desempeño en Javascript, pero no así en Python, como se puede observar en el ejemplo de la Figura 5.2.

(a) Python (b) Javascript

Figura 5.2: Detección de frutas (en imagen borrosa) en distintos lenguajes

Con respecto a la alta desviación estándar en el número de limones detectados en las imágenes, esto se debe a que dentro de la base de datos con la que se trabajó, varias de las imágenes correspondes a bins no completos, como se ve en la Figura 5.3.

Figura 5.3: Bin de limones incompleto

En cuanto a la diferencia de desempeño de la implementación de Javascript cuando se ejecuta en Chrome o la aplicación móvil, la única explicación por el momento es que la aplicación móvil dispone de menos memoria RAM y al cargar la imagen reduce la calidad de esta, lo cuál puede afectar el resultado. Intuitivamente este no debiese ser el único motivo, por lo que se deben realizar más pruebas para determinar la diferencia en los resultados.

5.1.2. Algoritmo detección plantilla de colores

El algoritmo que detecta los colores para clasificación a partir de una foto de la plantilla de colores fue desarrollado directamente en Javascript. A diferencia del algoritmo de detección de limones, donde se contaba con una extensa base de datos y fotos para pruebas durante el desarrollo, en este caso se contó inicialmente con una sola foto del patrón de colores sacada en la planta de inspección (Figura 4.6). Posteriormente se logró obtener más fotos, pero los resultados no fueron satisfactorios, ya que el algoritmo no es lo suficientemente robusto frente a variadas condiciones de luz, como se ve en la Figura 5.4.

La razón por la cual el algoritmo no es lo suficientemente robusto se debe a que existen varios parámetros que fueron ajustados usando la imagen disponible inicialmente. En particular uno de estos parámetros es el filtro de colores que se utiliza para generar la máscara para la

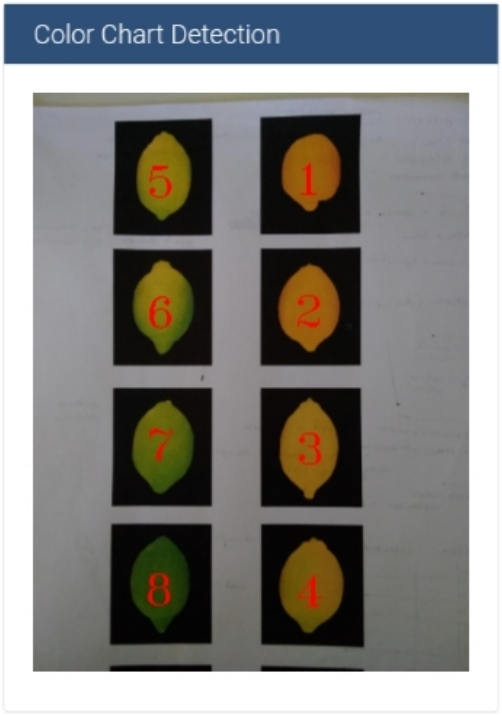
segmentación.

(a) Detección exitosa

(b) Detección fallida

Figura 5.4: Detección de Clases

La Figura 5.5 muestra más ejemplos de la detección de la plantilla patrón de colores, donde podemos ver que distintas condiciones de iluminación generan diferentes resultados. Además, se ve que las fotos donde hay menor iluminación dan mejor resultado en la detección, pero esto afectará el resultado del cálculo del color promedio de cada limón.



(a) Detección exitosa

(b) Detección fallida

(c) Detección fallida

(d) Detección fallida

Figura 5.5: Detección de clases en distintas condiciones de iluminación

5.1.3. Algoritmo de clasificación de colores

Una vez que se obtiene la detección de los limones, se procede a obtener el valor del color medio de cada uno de estos y se compara con el valor medio de cada uno de los limones en

la plantilla de colores.

Dado que el número de limones detectados es variable, para evaluar los resultados trabajamos con los porcentajes encontrados para cada clase de color. Estos son comparados con los resultados obtenidos por los inspectores mediante inspección visual y conteo manual para cada uno de los bins que forman parte de nuestra base de datos, para los cuales contamos con una foto y los porcentajes de clasificación. La tabla 5.1 muestra un ejemplo de clasificación para la muestra representada por la imagen de la Figura 5.1.

Tabla 5.1: Clasificación Figura 5.6

Inspección / Color	c1	c2	c3	c4	c5	c6	c7	c8
Manual	28 %	33 %	12 %	0 %	22 %	5 %	0 %	0 %
Automática	3.7 %	42.3 %	7.5 %	0 %	27.5 %	7.1 %	9.1 %	2.8 %

Figura 5.6: Bin de limones

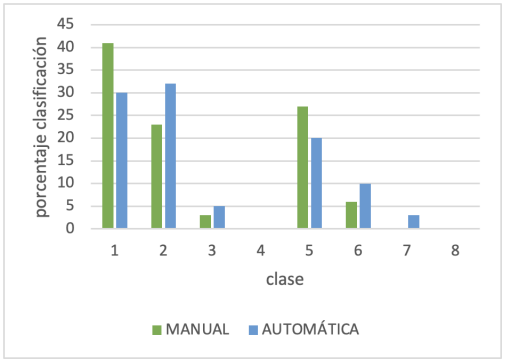
En los problemas clásicos de clasificación, para evaluar los clasificadores automáticos se cuenta con información sobre la clase real y clase asignada automáticamente para cada muestra, en este caso sería saber el color asignado por un inspector a un limón específico y luego el color automático, con esto se podría establecer qué porcentaje de los limones fueron clasificados correctamente, e idealmente construir una matriz de confusión para identificar las clases que se confunden. Sin embargo, no se cuenta con la información a ese nivel de detalle. La información con la que se cuenta es la distribución de clasificación del bin completo en base a una muestra representativa clasificada manualmente por conteo del inspector y la distribución de clases determinada por sistema de clasificación automática. Es por esto que no se puede aplicar directamente una métrica para la evaluación y sólo se puede evaluar el algoritmo en base a la distribución de clases del bin completo, haciendo un análisis visual de ambas distribuciones.

Se escogieron 8 bins representativos de los resultados de la clasificación para mostrar y analizar sus resultados, Figura 5.7. Estos gráficos representan el porcentaje de clasificación asignado a cada color tanto por el inspector, en este caso descrito como clase “manual” y por el clasificador automático.

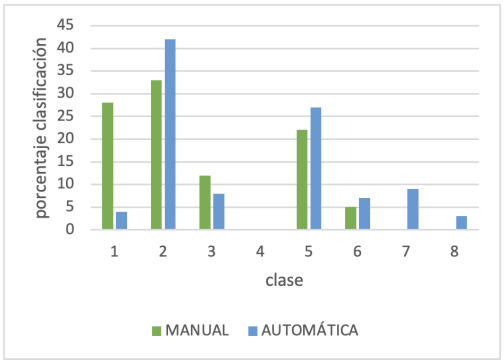
Una observación que podemos realizar es que el clasificador automático tiene una tendencia a asignar un porcentaje bajo de clasificación a las clases 6, 7 y 8 cuando la inspección visual

no lo hace. Si nos referimos a la plantilla patrón de colores en la Figura 5.4, vemos que estas clases corresponden a los colores más oscuros (verdes). Esto podría deberse a las sombras de la imagen, pero intuitivamente se puede pensar que la razón es que los inspectores tienen una tendencia a asignar los porcentajes de color a un grupo reducido de colores en el centro de la escala de maduración.

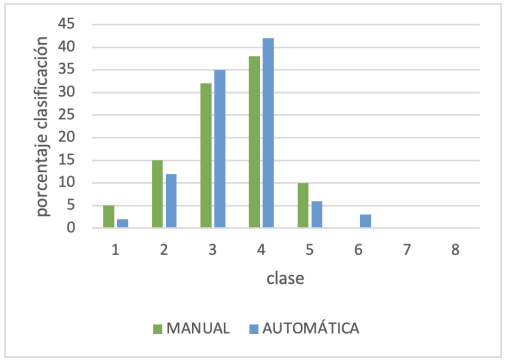
Otra observación que se puede realizar es que, en varios de los resultados, específicamente Figuras 5.7.a, 5.7.f, 5.7.g y 5.7.h, los porcentajes para las clases 1 y 2 parecieran estar invertidos entre la clasificación real y la automática. En referencia a la plantilla de colores en la Figura 5.4 podemos notar que los limones representativos de estas clases tienen colores casi idénticos, incluso podemos argumentar que visualmente la clase 2 contiene más verde que la clase 1, cuando se supone que la clase 2 representa un limón más maduro (desverizado) que la clase 2. Para evitar problemas de este tipo se debiese tener una plantilla de colores donde cada clase difiere más la una de la otra.



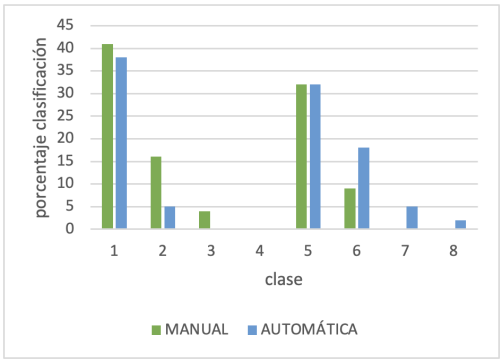
(a) Clasificación Bin 1



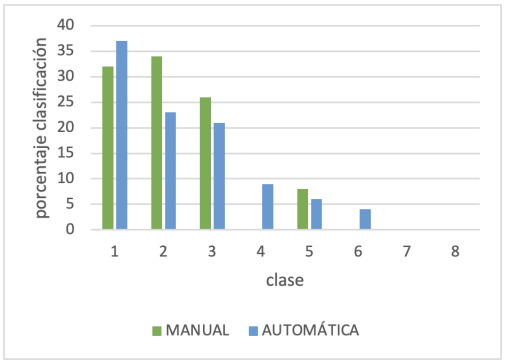
(b) Clasificación Bin 2



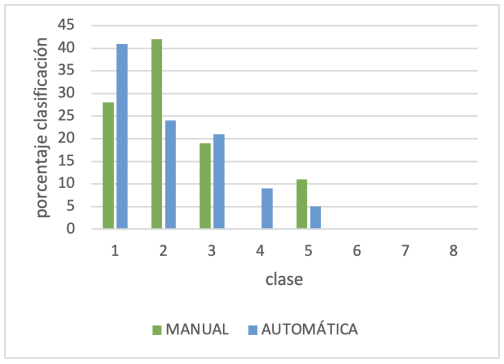
(c) Clasificación Bin 3



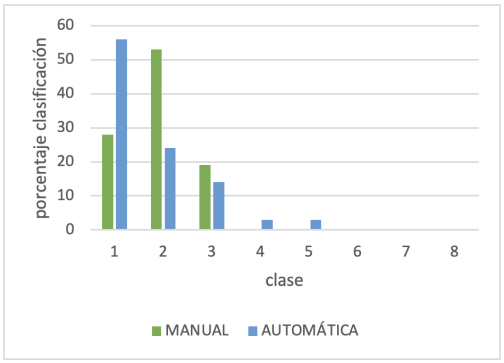
(d) Clasificación Bin 4



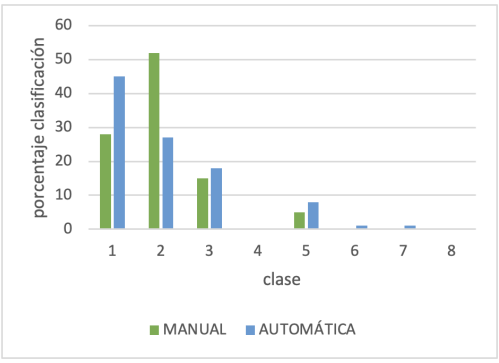
(e) Clasificación Bin 5



(f) Clasificación Bin 6



(g) Clasificación Bin 7



(h) Clasificación Bin 8

Figura 5.7: Clasificación de colores por Bin

5.2. Resultados evaluación de usabilidad

En el capítulo 3.4 se describe el procedimiento que se llevará a cabo para evaluar la usabilidad de la nueva funcionalidad de la aplicación desarrollada en este trabajo. En esta sección describimos los resultados de la evaluación.

Dado que en la sección anterior vimos que existían problemas con la detección de la plantilla patrón de colores, se tomó la decisión de realizar dos tareas con los usuarios. En primera instancia los usuarios deben utilizar la foto de la plantilla ya cargada en la Tablet, dado que sabemos que esta no tiene problemas para detectar cada clase. De esta forma podemos concentrarnos en evaluar la respuesta de los usuarios a la detección de limones en el Bin. En segunda instancia los usuarios deben tomar fotos tanto del Bin como de la plantilla.

Respecto a la clasificación de color de los limones, esta no pudo ser evaluada por los usuarios en esta instancia, ya que en la fecha en que se realizó estas pruebas, los limones ya habían pasado por un proceso de lavado y clasificación de color (separación por color), previo a un proceso de desverdizado.

Los parámetros medidos son los descritos en el capítulo 3.4, esto es, tasa de cumplimiento, número de errores y tiempo de ejecución. Además de estos parámetros se medirá el número de intentos que realizan (cuántas veces deben sacar las fotos) para lograr el éxito de la tarea. Si el usuario debe tomar más de una foto esto será sumado al número de errores. Para ejecutar la tarea los usuarios deben abrir la aplicación y navegar hasta una inspección de limones ya guardada, el tiempo de ejecución se calcula desde que abren la aplicación.

Una vez realizada cada tarea los usuarios asignan una nota respecto a la facilidad de esta. Al final de ambas tareas se le pide a los usuarios realizar comentarios.

5.2.1. Tarea 1

La instrucción para la primera tarea fue utilizar el módulo de clasificación automática de colores de la aplicación usando la imagen de la plantilla patrón de colores guardada en la tablet. Los usuarios deben cargar la imagen de la plantilla patrón de colores desde los archivos y utilizar la cámara para tomar la foto del Bin de limones. La imagen del Bin debe ser capturada nuevamente si la detección de limones no logra un número de detecciones que les parece representativo del Bin (aproximadamente 80 limones).

Los resultados de esta tarea se muestran en la tabla 5.2. Como se puede apreciar en la tabla, todos los usuarios cumplieron la tarea con éxito, con un promedio de tiempo de ejecución de 1min 17seg. En este caso 2 usuarios debieron repetir la captura de la foto del Bin, uno de ellos realizando la captura 3 veces. Esto se debió a que las imágenes no eran lo suficientemente nítidas, lo que llevó a un bajo número de limones detectados.

Tabla 5.2: Resultados usuarios Tarea 1

Métrica / Usuario	usuario 1	usuario 2	usuario 3	usuario 4
Tasa de cumplimiento	1	1	1	1
Número de errores	0	0	2	1
Tiempo de ejecución	1min 7seg	1min 20seg	2min 6seg	1min 40seg
Número captura fotos Bin	1	1	3	2
Nota facilidad para realizar la tarea	7	7	6	6

5.2.2. Tarea 2

La instrucción de la segunda tarea es utilizar el módulo de clasificación automática de colores de la aplicación haciendo uso de la cámara para capturar tanto la imagen del bin como de la plantilla patrón de colores. La imagen del bin debe ser capturada nuevamente si la detección de limones no logra un número de detecciones que les parece representativo del bin (aproximadamente 80 limones). La imagen de la plantilla patrón también debe ser capturada de nuevo si la detección no es exitosa.

Los resultados de esta tarea se muestran en la tabla 5.3. En este caso sólo un usuario logró completar la tarea exitosamente, mientras que los otros 3 usuarios no lograron completarla. La razón por lo cual no completaron la tarea se debió a que tras múltiples capturas de la foto de la plantilla patrón de colores, para ninguna de estas fotos la detección de los colores a clasificar fue realizada con éxito.

Tabla 5.3: Resultados usuarios Tarea 2

Métrica / Usuario	usuario 1	usuario 2	usuario 3	usuario 4
Tasa de cumplimiento	1	0	0	0
Número de errores	2	3	6	4
Tiempo de ejecución	3min 10seg	2min 38seg	3min 40seg	3min 20seg
Número captura fotos patrón	3	4	7	5
Número captura fotos Bin	1	1	1	1
Nota facilidad para realizar la tarea	5	2	2	1

5.2.3. Comentarios de los usuarios

Los usuarios se mostraron satisfechos con el avance realizado respecto a la detección automática de limones, particularmente tras realizar la tarea 1. Expresaron que era un gran problema no poder tomar la foto de la plantilla patrón de colores con éxito, pero que, si este aspecto se arregla, entonces la nueva funcionalidad de la app debiese ahorrarles mucho tiempo en las inspecciones de calidad de limón.

Respecto a la clasificación de color de los limones, se les explicó a los usuarios y sus supervisores cuáles son los resultados que obtuvimos de acuerdo con las imágenes de prueba. El comentario más realizado es que efectivamente existen clases de colores muy cercanas las

unas de las otras y que sería ideal poder hacer una mejor distinción entre ellas. También expresaron que cuando existía un muy bajo porcentaje de limones en las clases de colores periféricas, estos se agrupaban a las clases de colores más cercanas, en este caso el poder modificar los resultados de la clasificación posterior a la detección automática sería muy útil.

El consenso de los usuarios y supervisores fue que la nueva funcionalidad presenta un gran progreso pero que para poder ser puesta en producción debe ser mejorada, particularmente con respecto a la detección de la planilla patrón de colores.

Capítulo 6

Conclusiones

En el presente trabajo, se incorporó una nueva funcionalidad para la aplicación móvil de QC-Forms, específicamente en el módulo para la inspección de control de calidad en recepción de limones. Esta funcionalidad consiste en detectar los limones a partir del procesamiento de una foto del bin recibido y clasificarlos en colores de acuerdo con el procesamiento de una foto de una plantilla patrón de colores.

En particular, se cumplió con el objetivo de implementar un método de detección de colores en un bin de limones, con los algoritmos específicos planteados, estos son, los métodos para la segmentación de limones, el reconocimiento de la plantilla de colores y la clasificación de colores de los limones detectados de acuerdo a la plantilla.

El nuevo módulo para la detección de limones se implementó con las restricciones planteadas, es decir, se ejecuta de manera local en el dispositivo sin necesidad de acceso a Internet y su tiempo de ejecución es menor a 5 segundos.

En este trabajo se presentaron las herramientas y funciones utilizadas, así como los distintos algoritmos desarrollados, junto con sus resultados. Se analizó cómo el lenguaje de programación y plataforma de ejecución afecta los resultados, y se realizó un análisis de las posibles causas de estas diferencias.

Se realizaron variadas pruebas en el desarrollo del algoritmo y con esto se pudo realizar un análisis de las debilidades y fortalezas de las variadas funciones disponibles para el procesamiento de imágenes, considerando las condiciones en terreno para la obtención de imágenes y las potenciales acciones que se pueden realizar para mejorar y corregirlas. En particular vimos que el algoritmo de detección de la plantilla patrón de color es sensible a la iluminación y es necesario tener una base de datos de imágenes de pruebas más extensa para poder desarrollar un algoritmo más robusto.

Con respecto al algoritmo de detección de limones, vimos que este tiene un resultado distinto en su implementación en Python que en Javascript, y ya que en la aplicación móvil se utiliza Javascript, es necesario hacer un análisis más extenso de este para poder mejorar los resultados. Sin embargo, los resultados ya obtenidos son satisfactorios para la implementación final.

En cuanto a la clasificación de colores, se obtuvieron buenos resultados, pero estos podrían

ser mejorados si la plantilla patrón de colores presentara mayores diferencias entre clases adyacentes.

Al final del trabajo se realizó una prueba con los usuarios, la cual nuevamente confirmó el problema de la detección de la plantilla patrón de colores, lo cual generó un problema para el cumplimiento de la tarea propuesta. Sin embargo, el resto de la implementación mostró gran potencial, lo cual fue expresado por los usuarios y sus supervisores.

Concluimos que el trabajo realizado es funcional y un buen punto de partida para una implementación final, pero que aún presenta detalles que deben ser mejorados. Para esto es necesario tener una fuente de fotos más extensa y verificar el comportamiento de las funciones de proceso de imágenes en Javascript en distintas condiciones, esto permitirá definir un proceso controlado para la toma de fotos, que incluyen ubicación de la cámara, condición de iluminación entre otras.

La utilización de procesamiento de imágenes para la clasificación de colores de frutas es un gran avance para el control de calidad de fruta, la clasificación de colores manual que hacen los inspectores, por la similitud de colores en el patrón resulta en un aspecto subjetivo del inspector en que dos personas pueden diferir en su interpretación, requiere experiencia en su evaluación y sobre todo consumo de tiempo en el conteo de una muestra significativa (mayor a 60 frutos) para su clasificación.

6.1. Trabajo futuro

Con respecto al trabajo futuro, como se mencionó anteriormente lo primero que se necesita para continuar es tener una base de datos más extensa, que contemple casos con mayores diferencias en iluminación, esto es necesario tanto para las imágenes de bin de limones como la plantilla patrón de colores. Una base de datos que contemple casos con mayores diferencias de iluminación permitirá que el algoritmo sea más robusto.

Una vez que se obtenga una base de datos que comprenda más casos, se debe analizar la razón por la cual el algoritmo se comporta de forma distinta en Javascript que en Python, y utilizar este análisis para mejorar el algoritmo. Otra posible opción es evaluar la posibilidad de incorporar Python a la aplicación, una ruta que no se siguió en este trabajo dada la limitación de tiempo.

En cuanto al algoritmo de clasificación de colores, diversos análisis se pueden realizar en base a intuición, entre ellos el hecho de que dos clases de colores se confunden entre ellas, pero para poder tener resultados más concretos y de esta forma avanzar en el desarrollo, necesitamos obtener muestras de las imágenes de bin de limones donde cada limón este claramente clasificado. Con cada limón clasificado de forma manual y automática, se podrá realizar un análisis concreto y en base a medidas estadísticas.

Es decir, necesitamos una base de datos más completa, con mayor información de la que se tiene. Actualmente por cada imagen sólo se tiene la información sobre la distribución de colores de acuerdo con la inspección visual. Si además se obtiene la clasificación de colores

de cada limón específico se pueden aplicar métricas de evaluación, obteniendo así un análisis más objetivo de los resultados.

Finalmente cabe mencionar que la intención es utilizar estos algoritmos como punto de partida para posteriormente extender la detección y clasificación a distintos tipos de frutas, y crear una función genérica que pueda ser utilizada para una gran variedad de ellas.

Bibliografía

[1] QCforms “Quality Control mobile app and web platform,”, http://www.qcforms.com

[2] Cyganek, B. (2013). Object detection and recognition in digital images.

[3] Chiagoziem C. Ukwuoma, Qin Zhiguang, Md Belal Bin Heyat, Liaqat Ali, Zahra Almaspoor, Happy N. Monday, “Recent Advancements in Fruit Detection and Classification Using Deep Learning Techniques”, 2022

[4] Nur-E-Aznin Mimma, Sumon Ahmed, Tahsin Rahman, Riasat Khan, “Fruits Classification and Detection Application Using Deep Learning”, 2022

[5] Gonzales, Rafael & Woods, Richard.“Digital Image Processing Third Edition”, (pp. 174, pp. 714, pp. 760, pp. 791, pp. 883), 2019.

[6] Shapiro, L. G., & Stockman, G. C.“Computer Vision”, (pp. 153), 2001

[7] Otsu, Nobuyuki. “A threshold selection method from gray level histograms”, IEEE Transactions on Systems, Man, and Cybernetics, Volume 9, (pp. 62-66), 1979

[8] Davies, E. R. “Computer and machine vision: Theory, algorithms, practicalities”, (pp. 82-93), 2012.

[9] Borgefors G. “Distance transformations in digital images”, Computer Vision, Graphics, and Image Processing, Volume 34, Issue 3, (pp. 344-371), 1986

[10] Suzuki, S and Abe, K. “Topological Structural Analysis of Digitized Binary Images by Border Following”, 1985

[11] Cordova Apache “open-source mobile development framework,” http://cordova.apache.org

[12] SQLite Plugin Corova “storage plugin for Cordova,” http://www.npmjs.com/package/cordova-sqlite-storage

[13] Camera Plugin Corova “API for taking pictures and for choosing images from the system’s image library,” http://cordova.apache.org/docs/en/10.x/reference/cordova-plugin-camera/

[14] File Plugin Corova “File API allowing read/write access to files residing on the device,” http://cordova.apache.org/docs/en/11.x/reference/cordova-plugin-file/

[15] Software Sustainability Institute “Software Evaluation Guide,” https://www.software.ac.uk

[16] Mike Jackson, Steve Crouch, Rob Baxter “Software Evaluation: Criteria-based Assessment,” Software Sustainability Institute, 2011.

[17] Mike Jackson, Steve Crouch, Rob Baxter "Software Evaluation: Tutorial-based Assessment," Software Sustainability Institute, 2011.

[18] European Space Agency "ESA Software Engineering Standards: Issues 2," European Space Agency, series: ESA PSS-05-0 Issue 2, 1991.

[19] Han X. Lin, Yee-Yin Choong & Gavriel Salvendy "A proposed index of usability: A method for comparing the relative usability of different software systems", Behaviour and Information Technology, 16:4-5, 267-277, 1997.

[20] Seffah, Ahmed Donyaee, Mohammad Kline, Rex & Padda, Harkirat. "Usability measurement and metrics: A consolidated model. " Software Quality Journal. 14. 159-178., 2006.

[21] Justin Mifsud. "Usability Metrics – A Guide To Quantify The Usability Of Any System", https://usabilitygeek.com/usability-metrics-a-guide-to-quantify-system-usability/

[22] "Software engineering, Product quality, Part 4: Quality in use metrics", ISO/IEC TR 9126-4:2004(E), 2004.

[23] Jakob Nielsen, "Usability Metrics", Nielsen Norman Group, 2001, https://www.nngroup.com/articles/usability-metrics/

[24] Chalmers, P.A., "Software Usability Metrics and Methods", Baranauskas, C., Palanque, P., Abascal, J., Barbosa, S.D.J. (eds) Human-Computer Interaction – INTERACT 2007. INTERACT 2007. Lecture Notes in Computer Science, vol 4663. Springer, Berlin, Heidelberg, 2007 https://doi.org/10.1007/978-3-540-74800-7_95

[25] "The Single Ease Question (SEQ)", https://measuringu.com/seq10/

[26] "System Usability Scale (SUS)", https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html