Reconocimiento y cuantificación de manzanas

Mercedes Marzoa Tanco

Sebastián Caggiano Tejera

Tutores:
Gonzalo Tejera - Matias Di Martino - Juan Saavedra
Facultad de Ingeniería - Universidad de la República

30 de septiembre de 2015

Resumen

El objetivo de este documento es introducir el trabajo realizado en el marco del proyecto de grado pgfrutos. Se espera que con la lectura del mismo el lector comprenda en forma general el problema de localización y conteo de manzanas mediante visión por computador.

Se realizó un estudio del estado del arte sobre el reconocimiento de frutos en árboles utilizando visión por computador.

Se llevaron a cabo diferentes técnicas para reconocer manzanas.

En una primera etapa se realizó el reconocimiento de los pixeles pertenecientes a manzanas utilizando diferentes técnicas y comparando los resultados de las mismas, como características se utilizó el matiz, la saturación y la densidad de bordes presentes en las manzanas. Las técnicas utilizadas para la clasificación fueron K Nearest Network, Support Vector Machine y Árbol de Decisión.

Se estudió la efectividad de las diferentes técnicas y se realizó una comparación de las mismas.

Luego de tener reconocidos los pixeles pertenecientes a manzanas, se estudió y desarrolló una técnica para reconocer manzanas en sí mismas y realizar el conteo por imagen. Para ello se utilizaron operaciones morfológicas, detección de círculos de Hough y estudio del área e intersecciones de los círculos obtenidos.

En el presente documento se encuentran los detalles de implementación y la forma de evaluación las diferentes técnicas.

Finalmente se presentan los resultados obtenidos, las conclusiones del proyecto y el trabajo a futuro.

Índice general

Índice de figuras

Capítulo 1

Introducción

Este capítulo presenta una introducción al proyecto de detección y cuantificación de manzanas en cultivos mediante visión por computador. Se presenta la motivación para realizar el mismo, un resumen del estado del arte y los aportes.

1.1. Motivación

En gran parte de los sistemas de producción actuales hay una necesidad creciente de obtener productos de mayor calidad a menor costo, aumentando así la competitividad.

El desarrollo de sistemas automáticos que permitan utilizar con mayor eficiencia los recursos humanos en términos de precisión, repetitividad o tiempo son una buena alternativa. La estimación del rendimiento de los cultivos es una tarea importante en el manejo del huerto de manzanas.

Una predicción de rendimiento precisa, ayuda a los productores a mejorar la calidad de la fruta y reducir los costos de operación. También se beneficia el embalaje industrial, ya que los administradores pueden utilizar los resultados de la estimación para planificar la capacidad óptima para el envasado y almacenamiento. La estimación automática del rendimiento del cultivo sería una mejora significativa para la industria [27].

A modo de ejemplo, el costo de realizar la cosecha de cítricos completamente a mano, puede fluctuar entre el 25 % y 33 % del costo total de producción [16].

1.2. Estado del arte

Actualmente las estimaciones de cultivo se realizan mayoritariamente a partir de datos históricos, análisis de condiciones climáticas y conteo de manzanas realizado por trabajadores en lugares específicos de muestreo. Este proceso consume tiempo y mano de obra, y el tamaño limitado de la muestra suele no ser suficiente para reflejar el rendimiento de distribución a través del cultivo, especialmente en aquellos con alta variabilidad espacial. Una solución ampliamente adoptada para la estimación automática de rendimiento de la fruta es utilizar la visión artificial para detectar, contar y cosechar la fruta en los árboles. Los esfuerzos actuales sobre la estimación de rendimiento de manzanas utilizando la visión por ordenador, se pueden clasificar en dos categorías:

1. La estimación contando manzanas
2. La estimación por la detección de la densidad de la flor.

Investigadores han trabajado en la primera categoría utilizando imágenes en color [19, 22, 25], imágenes hiperespectrales [23], e imágenes térmicas [24]. Su punto en común es que sólo se ocupan de detección de manzanas en la huerta.

Aggelopoulou et al. [7] trabajaron en la segunda categoría. Se muestrearon imágenes de árboles en flor de un huerto de manzanas, y se encontró una correlación entre la densidad de la flor y el rendimiento de los cultivos. Sin embargo, este método no es convincente porque hay múltiples factores impredecibles (como las condiciones meteorológicas) durante el largo período entre la floración y la cosecha, haciendo que la correlación pueda variar de año en año.

Al llevar a cabo la detección de manzanas basada en un sistema de visión por ordenador, se enfrentan tres desafíos debido a las características de los entornos de la huerta:

- Desafío 1: variación en la iluminación natural.
- Desafío 2: oclusión de fruto causado por el follaje, ramas y otras frutas.
- Desafío 3: múltiples detecciones de misma manzana en imágenes secuenciales.

Para los distintos desafíos se han tomado distintas estrategias. Para superar el desafío de la variación de iluminación natural, se ha realizado la adquisición de imágenes en la noche [27, Pág. 1], iluminando con luz artificial. Para la oclusión de fruto se han aplicado transformaciones de Hough de círculos [16, Pág. 3] donde se reconocen formas elípticas con cierta tolerancia a oclusiones. Finalmente para el desafío de múltiples detecciones de mismas manzanas en secuencia de imágenes, se descartan manzanas duplicadas a través de cámara stereo, ya que triangula la posición en 3D y descarta las manzanas que coincidan en el espacio [27, Pág 9].

A partir del último desafío mencionado donde se puede calcular la ubicación espacial de los frutos, es que se hace posible la cosecha robótica, gracias a calcular la ubicación del fruto, se puede lograr la recolección utilizando un brazo mecánico. Para esto es necesario entonces el cálculo de la distancia a la que se encuentra con respecto al sensor de adquisición. En [16, Pág. 3] se calcula la distancia (segmentación profunda) a través de la cámara stereo.

A continuación se presenta un resumen de varias técnicas de reconocimiento de frutos, así como los resultados experimentales de las mismas.

Jun Zhao et al. [30] localizan manzanas en una sola imagen. Consideran manzanas en el árbol utilizando el contraste de manzanas rojas y verdes. Encontraron que la rojez, tanto en manzanas rojas como verdes, se puede utilizar para diferenciar las manzanas del resto del cultivo. Combinaron detección basada en bordes, rojez y umbral del área seguido de la búsqueda del círculo apropiado para determinar la ubicación de las manzanas en la imagen. En el caso de entornos muy ocluidos, utilizaron filtros Laplacianos para resaltar aún más los bordes del follaje y así generar que la diferencia de texturas entre el follaje y las manzanas incremente, facilitando la separación de las mismas. Lograron separar en muchos casos de forma exitosa tanto manzanas rojas como manzanas verdes, incluso cuando estaban juntas u ocluidas.

Para ello siguieron las siguientes etapas:

(i) Convertir imagen de RGB(Red, Green, Blue) a rojiza usando $r = 3R - (G + B)$.

(ii) Utilizar erosión y dilatación para disminuir el efecto de la oclusión parcial tanto como sea posible.

(iii) Usando detección de bordes separar manzanas del follaje.

(iv) Utilizando área y circularidad reconocer cada fruto.

(v) Utilizar filtros Laplacianos en el caso que el contraste sea bajo como para distinguir los bordes.

Se puede ver en la figura 1.1 ejemplos de los resultados de los pasos recién mencionados.

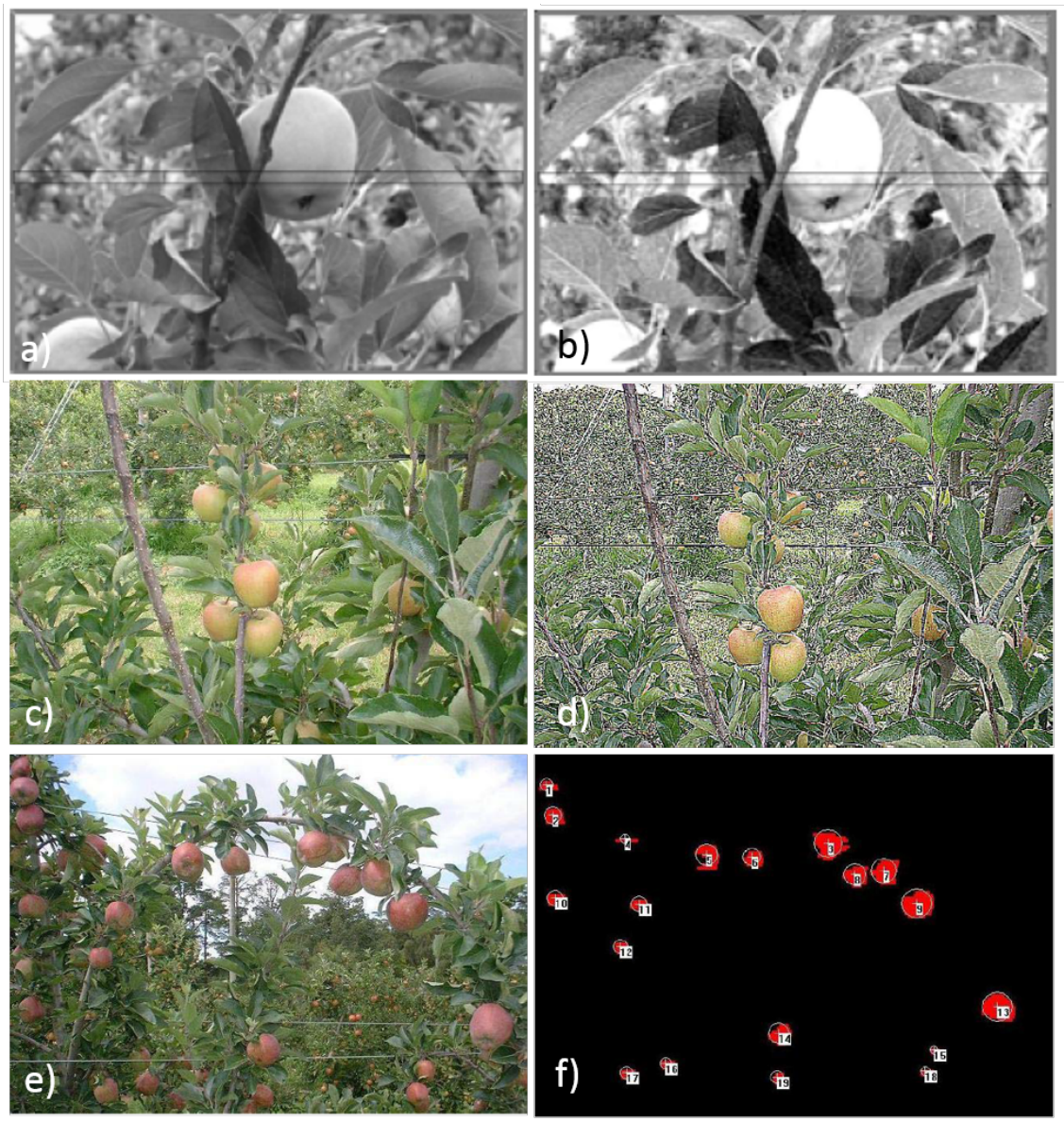


Figura 1.1: Imágenes ilustrativas de las técnicas utilizadas en el sistema de Jun Zhao et al. *a) Imagen en escala de grises. b) Imagen luego de aplicar la fórmula de rojez. c)Manzanas en un fondo tupido. d)Imagen filtrada con filtro Laplaciano. e)Imagen original. f)Resultado de (e).*[30]

Wei Ji et al. [15] proponen el desarrollo de un sistema de visión para el reconocimiento de manzanas en un robot cosechador. Utilizan una cámara a color para capturar imágenes y luego un sistema para procesar la imagen para el reconocimiento de frutos. Se aplica un filtro de vector de mediana para remover el ruido de las manzanas e investigan un método de segmentación basado en regiones crecientes y características de color. Luego, se extraen las características de color y forma de la imagen, y se introduce un nuevo algoritmo de clasificación basado en máquinas de vectores de soporte para el

reconocimiento de manzanas para mejorar la precisión y eficiencia. Finalmente, en el 2009, testearon el sistema bajo condiciones naturales y mostraron una taza reconocimiento de aproximadamente el 89 % y un promedio de tiempo de 352 ms. En la figura 1.2 se puede observar el resultado de esto.

Figura 1.2: Imágenes de las técnicas utilizadas por Wei Ji et al. [15] *a)Imagen original b)Imagen luego de filtrada c)Imagen luego de aplicar operaciones morfológicas y segmentación d)Imagen resultado*

Qi Wang et al [27] desarrollaron un sistema automático basado en visión rápido y preciso para la estimación del rendimiento de un cultivo. El sistema utiliza dos cámaras y escanea de los dos lados las filas de árboles durante la noche. Un algoritmo basado en visión por computadora detecta y registra manzanas tomadas en imágenes secuenciales y luego genera el conteo de manzanas para la estimación. Encontraron que el sistema funcionó de forma correcta tanto con manzanas rojas como con manzanas verdes. Presentan un estudio completo de casos de los resultados según tipo de corte de los arboles así como manchas en las manzanas, acumulación de manzanas y entornos ocluidos, cómo se puede observar en la figura 1.3.

Para ello siguieron las siguientes etapas:

(i) Utilizan un umbral de Matiz(H) y Saturación(S) para detectar píxeles pertenecientes a manzanas rojas y verdes.

(ii) En las manzanas verdes deben buscar el reflejo especular debido a que luego de la clasificación según H y S quedan clasificados como píxeles no manzana.

(iii) Segmentan las manzanas individualmente utilizando operaciones morfológicas.

(iv) Por ultimo buscan la posición global de la manzana para así evitar que sea contada más de una vez.

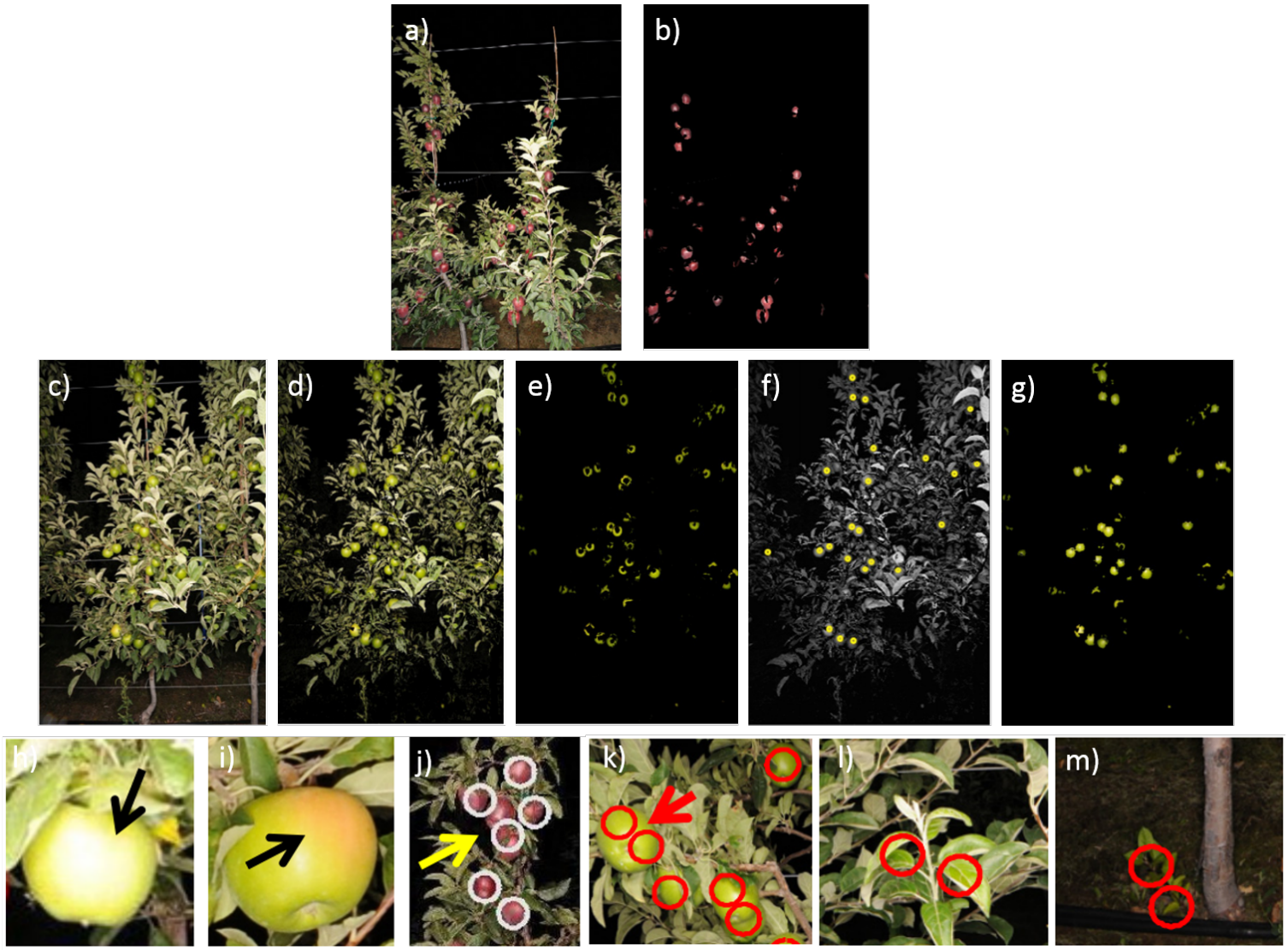


Figura 1.3: Imágenes ejemplo de las técnicas utilizadas por Qi Wang et al. *[27]. a) y c) Imagen RGB. b)Imagen resultado de la segmentación de a). d) Segmentación según Hue. e) Segmentación según Saturation. f)Detección de reflejo especular. g)Combinación de los resultados de e) y f). h), i), j) Ejemplos de manzanas visibles que no son detectadas por el software por la h) Sobreexposición, i) Mancha del sol y j) Pérdida de manzanas al encontrarse varias juntas. k), l), m) Ejemplos de detecciones falso positivas, k) Manzanas cercanas a la cámara l)Hojas nuevas con color similar al de las manzanas verdes, m)Yuyos con color similar.*

L. Yang et al. [29] presentan un método para detectar y reconocer tomates maduros en una planta que contiene tomates amontonados y ocluidos con propósitos de recolección automática. Como sensor de visión utilizan una cámara estereo. Los métodos propuestos realizan una reconstrucción 3D con la información recolectada por la cámara para crear un ambiente 3D de procesamiento. El método de capas de colores de un cultivo es introducido para segmentar las frutas maduras de las hojas, tallo, fondo y ruido. Las frutas objetivo pueden ser ubicadas por segmentación de profundidad. Los datos experimentales fueron recolectados de un invernadero de tomates y el método justificado por los resultados experimentales.

Para ello siguieron las siguientes etapas:

(i) Aplican suavizado y detección de bordes.

(ii) Realizan segmentación por color (CLG) utilizando la técnica de región creciente, con píxeles semillas para luego comenzar el estudio de los píxeles próximos y unirlos a la región en caso de similitudes. La similitud de los píxeles se determina comparando el nivel de color

(iii) Eliminan la información innecesaria utilizando un filtro de profundidad.

A.R. Jiménez et al. [16] presentan una metodología para reconocer frutas esféricas bajo condiciones naturales, lidiando con situaciones complejas: sombras, áreas con brillo, oclusión y frutas superpuestas. Como sensor utilizan un láser rangefinder que da la distancia al objeto censado. Deducen que la mejor alternativa es estudiar la forma del contorno y color, el sistema de reconocimiento utiliza un modelo de láser rangefinder y un algoritmo de análisis dual color/forma para ubicar la fruta. Utilizan cuatro características para el reconocimiento de frutos: color, intensidad, forma y textura.

La posición tridimensional de la fruta, radio y reflectancia se obtiene luego de las etapas de reconocimiento, obteniendo los datos de las coordenadas esféricas (3D) de cada punto de la escena, así como también el valor de atenuación de energía del láser provocada por la distancia, el tipo de textura y la orientación de la superficie detectada. En la figura 1.4 se presenta los distintos ejemplos.

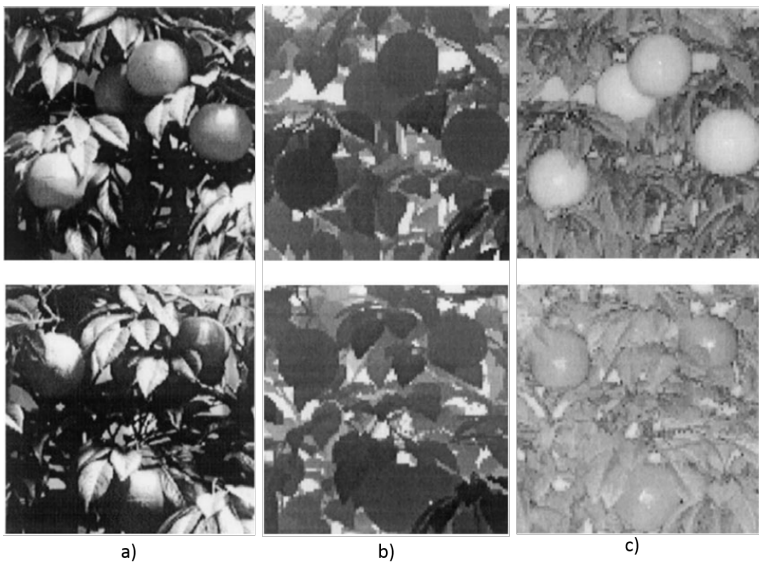


Figura 1.4: Imágenes ejemplo de las técnicas utilizadas por A.R. Jiménez et al. Algunos ejemplos de imágenes de rango y de atenuación de dos escenas de naranjos artificiales diferentes. En la parte superior, de izquierda a derecha: escena con cuatro naranjas maduras; la imagen del rango y la imagen de atenuación. En la parte inferior, otra secuencia para una escena con cuatro naranjas verdes.[16]

En el cuadro 1.1, se puede ver un resumen de los proyectos recién mencionados.

Referencia	Fruta a reconocer	Tipo de imagen utilizada	Métodos utilizados	Rendimiento
Jun Zhao et al. [30]	Manzanas rojas	Rojiza: $r = 3R-(G+B)$	Erosión y Dilatación	No especifica
	Manzanas verdes		Detección de bordes	
Wei Ji et al. [15]	Manzanas rojas	RGB	Filtro de mediana	89 %
Qi Wang et al [27]	Manzanas rojas	HSV, nocturnas	Clasificación según H y S	96.8 %
	Manzanas verdes		Segmentación según circularidad y tamaño	
L. Yang et al.[29]	Tomates	Imágenes a color con una cámara stereo	Suavizado y detección de bordes	No especifíca
A.R. Jiménez et al. [16]	Naranjas	Imagen de rango y atenuación	Segmentación según color, forma, intencidad y textura	87 %
	Manzanas		Obtienen datos de las coordenadas 3D de cada punto	
	Durazno		Hough para estudiar forma de contorno	

Cuadro 1.1: *Resumen de las técnicas estudiadas para el reconocimiento de manzanas en arboles.*

En cuanto a la selección del radio promedio, en [27, 3.3] se realiza la cuantificación de manzanas estudiando su forma basándose en el cálculo del radio promedio ($\bar{D}$) de las manzanas en cada imagen.

Primero se toma la imagen binaria representando las regiones de manzanas. Se calcula la excentricidad (E) de cada región en la imagen, y se utiliza un umbral definido para encontrar regiones relativamente circulares. Estas regiones tienen una alta probabilidad de representar manzanas completas, sin oclusiones, y sin contacto con otras manzanas. Las regiones así identificadas son buenas candidatas para calcular el radio promedio.

A su vez se detectan regiones circulares considerablemente más pequeñas que el tamaño promedio, estas regiones puede ser resultado de manzanas con varias oclusiones a su alrededor. Para eliminarlas se calcula el área individual (S) de estas regiones y luego se calcula su área promedio ($\bar{S}$). Se conservarán solamente las regiones cuya área $S > \bar{S}$.

Debido a que las manzanas no son del todo circulares, son elípticas, se toma el eje menor de la elipse para calcular el radio promedio $\bar{D}$.

Con el radio promedio calculado, se continúa con la siguiente etapa que consiste en el estudio de las regiones con dos o más manzanas en contacto. Se calcula el diámetro mayor (L_{mayor}) de los ejes de centro de masa de la región, y se aplica $L_{mayor} > 2\bar{D}$.

El artículo citado indica que el método de cuantificación es indicado para aquellas regiones que presentan hasta dos manzanas en contacto, como se visualiza en Figura 1.5. En este caso ubica el centro de cada manzana como el centro de cada segmento resultado de la división entre dos del eje mayor de la elipse.

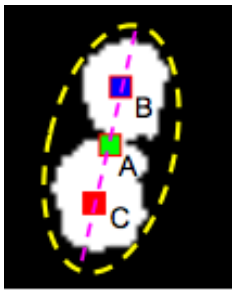


Figura 1.5: Resultado de dividir una región conteniendo dos manzanas. *A es el centro del segmento B-C donde B y C son los centros de cada manzana.*

1.3. Aportes

El objetivo general de este proyecto es diseñar e implementar un sistema automatizado para la detección y conteo de manzanas rojas en árboles, que sirva cómo base, para un sistema más completo de estimación de rendimiento de cultivos o de un sistema de recolección automático de manzanas.

Se dejan documentadas diferentes técnicas para el reconocimiento de píxeles manzana y la comparación de las mismas.

Quedan disponibles para proyectos futuros una base de datos etiquetada, lo cual es muy útil para futuras investigaciones.

Capítulo 2

Marco teórico

Este capítulo presenta el marco teórico sobre los temas que abarca el proyecto. En la Sección 2.1 se introducen los conceptos básicos de las imágenes digitales, en la Sección 2.2 se presentan las etapas clásicas de un sistema de reconocimiento de patrones y en la Sección 2.3 se describe la selección y extracción de características. En la Sección 2.4 se presentan los tres métodos utilizados como clasificadores, y por último, en la Sección 2.5 se presentan las técnicas para el análisis de formas.

2.1. Fundamentos básicos de imágenes digitales

2.1.1. Adquisición de imagen digital

Previo a aplicar cualquier procesamiento a una imagen, debe de haberse realizado la adquisición de la misma. El proceso de adquisición de imágenes consiste de los siguientes tres pasos [20, Página 7]: Energía reflectada del objeto de interés, un sistema óptico que enfoca la energía y finalmente un sensor que mide la cantidad de energía. En la figura 2.1 se visualizan los tres pasos para el caso de una cámara fotográfica ordinaria con el sol como fuente energética.

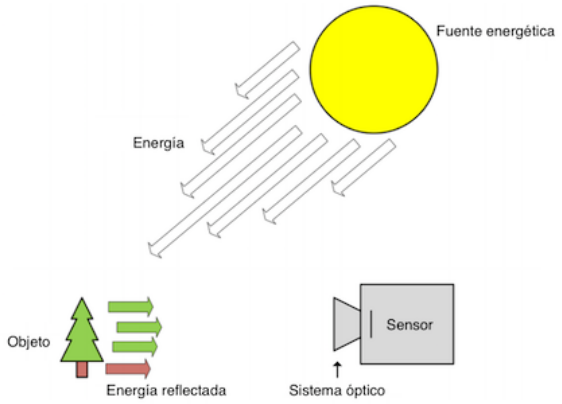


Figura 2.1: Esquema de adquisición. *Escenario típico para el proceso de adquisición de imagen, con el sol como fuente energética o lumínica, un árbol como objeto, y una cámara para capturar la imagen. En el caso de una cámara analógica se utilizaría un negativo, y en una cámara digital un sensor.*

Iluminación

Para capturar una imagen, es necesario iluminar la escena con algún tipo de fuente de luz. En la figura 2.1 el sol es la fuente. Comúnmente se utiliza luz, ondas electromagnéticas que pueden ser percibidas por un humano, aunque existen otras formas, en frecuencias no visibles.

Si se procesan imágenes adquiridas, no hay mucho para hacer en cuanto a la iluminación presente en el momento de su adquisición. Pero cuando se está a cargo de realizar la adquisición es de mucha importancia tomar en cuenta la iluminación, ya que es un factor muy importante dentro del proceso de adquisición. En la figura 2.2 se muestra esto, donde el único cambio realizado en la adquisición de las cuatro imágenes fue la ubicación de la fuente, una lámpara. Se puede observar una gran variación entre una imagen y la otra.

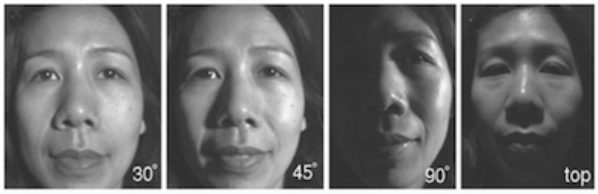


Figura 2.2: Iluminación. *El efecto de iluminar una cara desde 4 direcciones distintas* [20, Figura 2.3, Página 9].

Sistema óptico

Luego de haber iluminado el objeto de interés, la luz reflejada desde el objeto ahora debe ser capturada por la cámara. Si el material sensible a la luz reflejada se encuentra próximo al objeto, una imagen del objeto será capturada. Aunque, como se muestra en la figura 2.3-a), luz desde diferentes puntos del objeto se mezclarán, teniendo como resultado una imagen inútil, sumándole a esto también la adquisición de luz del entorno. La solución a esto se observa en la figura 2.3-b), consiste en ubicar algún tipo de filtrado entre el objeto de interés y el material sensible a la luz. Nótese que como consecuencia la imagen se invertirá verticalmente, ver la figura 2.3-c). Para corregir esto se aplican mecanismos de software y hardware.

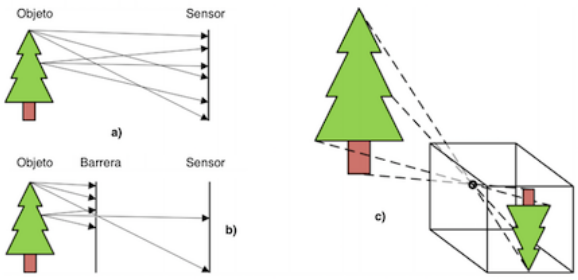


Figura 2.3: Sistema óptico. *Antes de introducir un filtro, los rayos de luz de distintos puntos del árbol impactan en múltiples puntos del sensor y en algunos casos en el mismos punto (a). Introduciendo un filtro con un hoyo pequeño, reduce significativamente estos problemas, b) y c).*

Mediante un pequeño orificio conseguimos formar una imagen del objeto. Esta estrategia es una de las más antiguas que se conoce para «enfocar» la luz. Actualmente, para obtener un mejor aprovechamiento de la luz disponible, se utiliza una lente que es básicamente un vidrio que enfoca la luz entrante en un sensor, como se visualiza en la figura 2.4. La idea es que dado un plano a la izquierda, plano en foco, existe un plano a la derecha, plano imagen, tal que los rayos que provienen de un punto en el plano foco van a parar al mismo punto en el plano imagen. En la figura 2.4, tres rayos de luz son ilustrados desde diferentes puntos. Los tres rayos se intersectan en un punto en particular a la derecha del lente. Enfocar esos rayos es exactamente el propósito del lente. Esto significa que la imagen del objeto se formará a la derecha del lente, y será la imagen que capture la cámara posicionando un sensor en la posición exacta. Notar que los rayos en paralelo se intersectan en un punto F, denotado cómo **Punto focal**. La distancia desde el centro del lente, el centro óptico O, al plano donde todos los rayos paralelos se intersectan se denota como la **Distancia focal f.** La línea donde se encuentran los puntos O y F se le llama el **Eje óptico**.

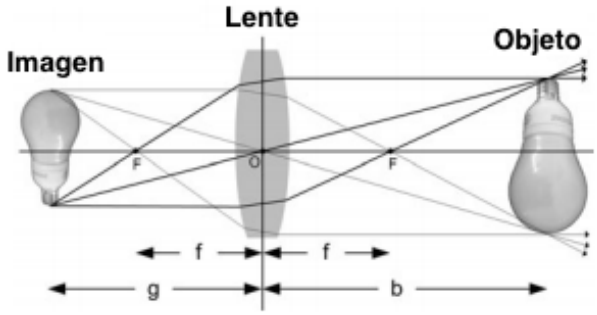


Figura 2.4: Lente. *La figura muestra cómo los rayos de un objeto (una lamparita) se enfocan a través del lente. La bombita real se encuentra a la izquierda y la imagen formada a la derecha* [20, Figura 2.6, Página 12].

Sensor de imagen

La luz reflejada desde el objeto de interés es enfocado por un sistema óptico, y es necesario ahora que se grabe en la cámara. Para este propósito se utiliza un sensor de imagen. Un sensor de imagen consiste de un arreglo 2D de celdas, como se ve en la figura 2.5. Cada celda denota un píxel y es capaz de medir la cantidad de luz incidente que luego se convierte en voltaje, para finalmente convertirse en un número digital.

Cuanto más luz incidente haya, mayor será el voltaje y mayor será el número digital. Antes de que la cámara capture una imagen, todas las celdas son vaciadas, de tal forma que no haya carga presente. Cuando la cámara se prepara para capturar una imagen, se permite el ingreso de luz y las cargas empiezan a acumular en cada celda. Luego de pasado un cierto tiempo, **tiempo de exposición**, la luz incidente se detiene, controlado por un obturador. Si el tiempo de exposición es muy poco o mucho, el resultado será una imagen sub-expuesta o sobre-expuesta respectivamente, como se puede ver en la figura 2.6. Vale destacar que se clasifica a una imagen como con correcta exposición,

cuando se presenta en ella un alto grado de similitud (en cuanto a luz) con la escena donde se adquirió la imagen.

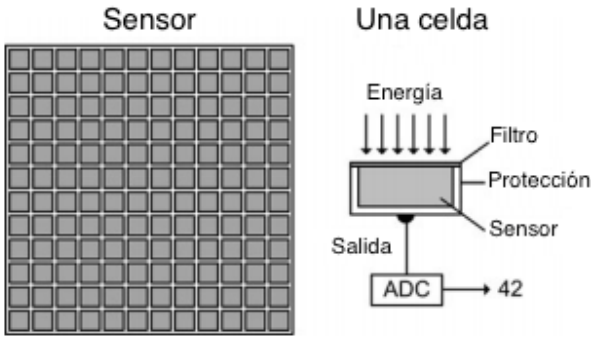


Figura 2.5: Sensor. *El sensor consiste de un arreglo de celdas interconectadas. Cada celda consiste de una protección que soporta el filtro, un sensor y una salida. El filtro controla qué tipo de energía es permitida entrar al sensor. El sensor mide la cantidad de energía como voltaje, que luego es convertido a un número digital a través de un conversor análogo-digital (ADC)* [20, Figura 2.13, Página 17].

Figura 2.6: Tiempo de exposición. *La imagen con correcta exposición, es debido a que el tiempo de exposición fue el correcto. En las imágenes sobre-expuesta y sub-expuesta se debió al mucho y poco tiempo de exposicón respectivamente* [20, Figura 2.4, Página 17].

Imagen digital

Para transformar la información de un sensor en una imagen, el contenido de cada celda se convierte al valor de un pixel en el rango de [0:255]. Tal valor se interpreta como la cantidad de luz que impactó sobre la celda durante el tiempo de exposición, también se denota como la intensidad del pixel. En la figura 2.7 se visualiza como se expresa el valor en escala de grises donde la intensidad va del valor más pequeño, 0, representando al negro, hacia el mayor valor, 255, representando el blanco.

Figura 2.7: Escala de gris. *La relación entre el valor numérico y las distintas tonalidades de gris.*

Como se verá a continuación, el píxel de una imagen se puede representar como la composición de 3 valores numéricos, RGB. Para obtener estos valores, el sensor cuenta con una celda dedicada para cada valor.

2.1.2. Espacio de color

Un espacio de color significa una forma única de definir un color. Comúnmente se utilizan varios espacios de colores, dependiendo de la industria particular y/o tipo de aplicación involucrada. Por ejemplo como humano se determina normalmente el color por parámetros tales como el brillo, matiz e intensidad. En computadoras es más común describir el color mediante tres componentes, normalmente rojo, verde y azul. Estos están relacionado con la excitación de los fósforos rojos, verdes y azul en los monitores de las computadoras. Otro sistema similar que es más aplicado en la industria de impresión, utiliza el cyan, magenta y amarillo, que se encuentran mas relacionados con la reflectancia y absorción de la tinta en papel.

A continuación se describirán 2 espacios de colores que serán con los que se trabajarán en este proyecto.

Espacio RGB

Se basa en la combinación de tres señales de luminancia cromática distinta: Rojo, Verde, Azul (Red, Green, Blue). La forma más sencilla de obtener un color específico es determinar la cantidad de color rojo, verde y azul que se requiere combinar para obtener el color deseado, ver la figura 2.8 para lo cual se realiza la suma aritmética de las componentes: X = R + G + B, gráficamente representada por un cubo.
En la figura 2.8 se presentan ejemplos de colores definidos mediante esta tripleta.

Color	R	G	B
Blanco	1	1	1
Rojo	1	0	0
Amarillo	1	1	0
Verde	0	1	0
Turquesa	0	1	1
Gris	0.5	0.5	0.5
Rojo Oscuro	0.5	0	0
Azul	0	0	1
Aguamarina	0.5	1	0.83
Negro	0	0	0

Figura 2.8: *Colores RGB.*

En la figura 2.9, en la recta que une el vértice de negro con el blanco (recta punteada), se encuentran ubicados los grises (escala de gris) debido a que sus tres componentes son iguales. Cuando una cámara adquiere una imagen a color, para cada píxel en color se tienen en realidad tres componentes, una para cada uno de los colores básicos (rojo, verde y azul), donde en cada componente se almacena la cantidad de color respectivo que se encuentra presente en el píxel.

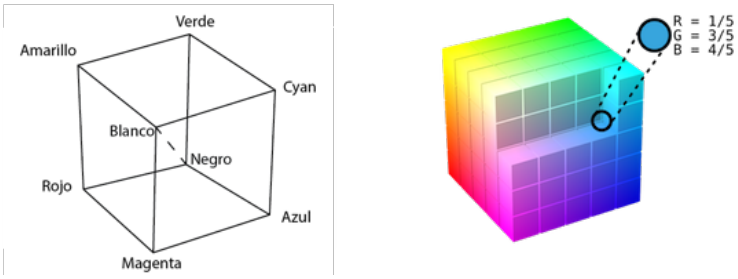


Figura 2.9: *Cubo RGB*.

Espacio HSV

Este espacio de color es una representación de los colores RGB en forma de coordenadas cilíndricas. Con este espacio se intenta que sea más intuitivo y perceptivo que en la representación cartesiana (cubo RGB), a través del mapeo de los colores en un cilindro tradicional de color. El ángulo alrededor del eje vertical corresponde a la matiz o tono, en inglés Hue, la distancia con el eje corresponde a la saturación, en inglés Saturation, y la altura del cilindro corresponde a luminosidad, en inglés Value. En la figura 2.10 se visualiza lo descripto.

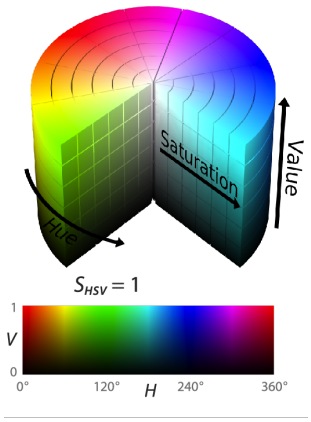


Figura 2.10: Cilindro HSV. *Value corresponde a Valor, Hue a matiz o tono y Saturation a saturación. En la gráfica inferior, se especifíca el tono en función de Valor, con saturación fija en 1. (Figura extraída de [5])*

2.2. Esquema clásico de reconocimiento de patrones

El reconocimiento de patrones es el estudio de cómo las maquinas observando el entorno pueden aprender a distinguir patrones de interés de un fondo, y tomar decisiones acertadas y razonables acerca de la categoría de los mismos (Anail Jain)[14, Pág. 3]

Surje de la definición anterior la necesidad de responder la siguiente pregunta, ¿Qué es un patrón? Watanabe [15] define un patrón como "lo opuesto al caos, es una entidad, vagamente definida, a la que se le puede dar un nombre".

El problema de reconocimiento es una tarea de clasificación o categorización. Se define a *Clasificación Supervisada*, como al tipo de clasificación que cuenta con un conocimiento a priori, es decir para la tarea de clasificar un objeto dentro de una categoría o clase ya se cuenta con modelos ya clasificados. Por otro lado, se define también otro tipo de clasificación llamada *Clasificación No Supervisada,* que a diferencia de *Clasificación Supervisada*, en este caso no se cuenta con conocimiento a priori, por lo que se debe tener un área de entrenamiento disponible para la tarea de clasificación [14, Pág. 17].

El diseño de un sistema de reconocimiento de patrones esencialmente involucra los siguientes tres aspectos, como se visualiza en la figura 2.11:

1. Adquisición y pre procesamiento de datos
2. Extracción de características
3. Entrenamiento/clasificación

Para ayudar a comprender en qué consisten los distintos pasos anteriores, se describirá el siguiente ejemplo:

Se plantea el caso hipotético en que se desea construir un sistema capaz de clasificar manzanas y bananas en una cinta transportadora. El primer paso que corresponde a *Adquisición y pre procesamiento de datos*, consistirá en primero posicionar la cámara en un lugar apropiado para la adquisición, luego adquirir la imagen y realizar el pre procesamiento, el cuál en este caso consiste en aplicar un filtro para quitar el ruido causado por la velocidad con la que pasa la fruta por debajo de la cámara y pasar a otro espacio de colores que permita trabajar mejor con el tono de las frutas consideradas.

Como segundo paso, se deberá realizar *Extracción de características*, que consiste en elegir qué propiedades de los objetos los discriminan mejor y cómo medirlas. Debido a que las manzanas y bananas se diferencian entre otras cosas por su tamaño y tonalidad, serán estas las características a utilizar como discriminador entre una fruta y la otra.

Cómo último paso, *Entrenamiento/clasificación*, se debe establecer el algoritmo de clasificación, escoger la estructura del clasificador que se utilizará y fijar parámetros ajustables, por ejemplo, fronteras de decisión en la clasificación.

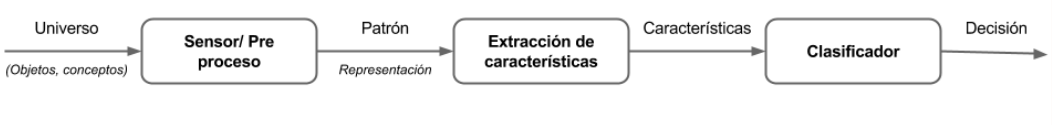


Figura 2.11: *Etapas en un Sistema de Reconocimiento de Patrones.*

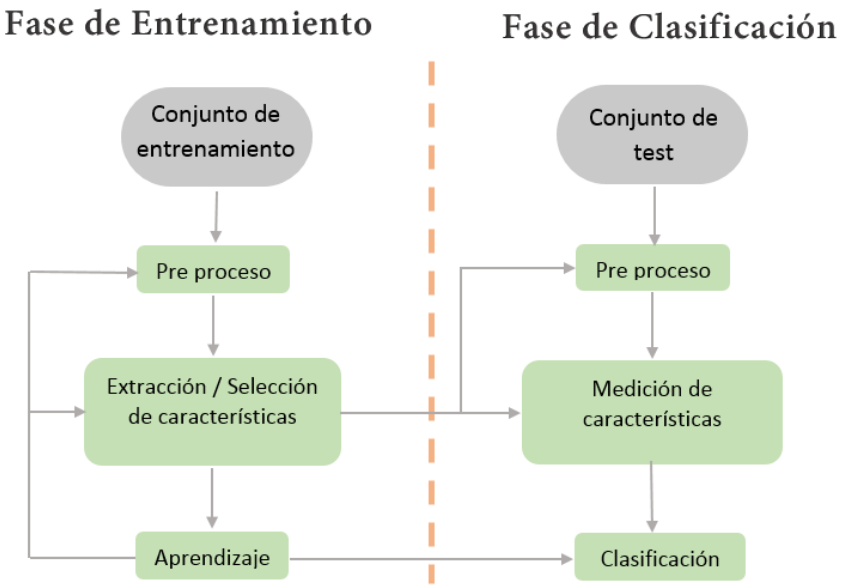


Figura 2.12: *Esquema del SRP Estadístico.*

Si bien las etapas del sistema de reconocimiento de patrones son las recién presentadas, generalmente el sistema de reconocimiento de patrones se divide en dos fases, fase de entrenamiento y fase de clasificación, cómo se ve en la figura 2.12.

El rol del módulo de pre procesamiento consiste en segmentar el patrón de interés del resto, remover el ruido, normalizar el patrón, y cualquier otra operación que contribuya a definir una representación compacta del mismo.

El módulo de extracción de características en la fase de entrenamiento, es quien se encarga de descubrir las características apropiadas para la representación de los patrones de entrada, mientras que en la fase de clasificación, se encarga de dividir el espacio de características.

El módulo clasificador, durante el entrenamiento aprende y durante la clasificación es el encargado de asignar a el patrón de entrada la etiqueta que corresponda a una de las clases.

Existe una trayectoria de retroalimentación entre el entrenamiento y la clasificación para permitir a un diseñador optimizar las estrategias de pre procesamiento y extracción/selección de características.

2.3. Extracción y selección de características

Selección de características

Dado un conjunto de d características, la Selección consiste en seleccionar un subconjunto de tamaño m que conduzca al menor error de clasificación.

Hay un gran interés por aplicar métodos de selección de características, debido al gran número de características que se pueden encontrar causadas por las siguientes situaciones:

1. Fusión de sensores: Características computadas por diferentes sensores, son concatenadas formando un vector de características con un número mayor de componentes.

2. Integración de modelos de múltiple información: Datos de sensores pueden ser modelados usando diferentes acercamientos, donde los parámetros de los modelos sirven como características, causando que los parámetros de diferentes modelos se agrupen formando un vector de mayor dimensión.

Sea Y el conjunto de características de cardinalidad d, y m el número deseado de características en un subconjunto X, $X \subset Y$. Sea $J(X)$ la función que representa el criterio de selección de características para el subconjunto X. Asumiendo que un valor superior de J indica un mejor subconjunto de características, una elección natural para el criterio es $J = (1 - P_e)$ donde P_e denota el error de clasificación. El uso de P_e en la función usada como criterio, hace que los procedimientos dependan del clasificador específico que se utiliza y de los tamaños de los conjuntos de prueba y entrenamiento.

El método más sencillo de selección de características requiere:

1. Examinar todas las $\binom{d}{m}$ posibles de subconjuntos de tamaño m

2. Seleccionar el subconjunto con el mayor valor de $J(X)$

Sin embargo, el número de posibles subconjuntos combinatorios crece, haciendo la búsqueda exhaustiva poco práctica, inclusive para valores moderados de m y d.

Un método "óptimo" de selección de características, que evita la búsqueda exhaustiva, es basado en el algoritmo de **Branch and bound**. Este procedimiento utiliza resultados intermedios para obtener cotas en el valor del criterio final. La clave del algoritmo es la propiedad monótona de la función $J(X)$: dado dos subconjuntos X_1 y X_2, si $X_1 \subset X_2$, entonces $J(X_1) < J(X_2)$. En otras palabras, la performance de un subconjunto de características debe mejorar siempre que una característica le sea agregada. El problema es que la mayoría de las funciones utilizadas no satisfacen este criterio de monotonía.

Se ha argumentado que mientras la selección de características es típicamente off-line, el tiempo de ejecución de un algoritmo particular no es tan crítico como la optimización del subconjunto que genera. Si bien esto es verdad para un conjunto de tamaño moderado de características, varias aplicaciones recientes, involucran cientos de características. En esos casos, los requerimientos computacionales de un algoritmo de selección son extremadamente demandantes.

Como el número de características de un subconjunto de evaluación puede fácilmente convertirse en prohibitivo, se han propuesto una serie de técnicas de selección sub óptimas que esencialmente compensan el óptimo del subconjunto seleccionado con la eficiencia computacional.

En general, buenos y grandes conjuntos de características, no necesariamente incluyen los conjuntos adecuados y pequeños. Como resultado, el simple método de seleccionar solamente la mejor característica individual, puede fallar dramáticamente.

Puede ser útil, sin embargo, como primer paso seleccionar algunas características adecuadas individualmente, disminuyendo el conjunto de características. Además la selección debe ser hecha por métodos más avanzados, que tomen en cuenta dependencias futuras [14].

Extracción de características

Los métodos de extracción de características determinan un subespacio adecuado de dimensión m, a partir del espacio original de características de dimensión d, $m \leq d$ [14, Página 12].

Las transformaciones lineales, tales cómo análisis de componentes principales (PCA), análisis factorial o análisis de discriminante lineal (LDA) entre otros, han sido utilizados ampliamente en reconocimiento de patrones para la extracción de características y reducción dimensional.

Utilizando métodos lineales y no lineales, todas las variables son utilizadas para transformar la información, y así reducir las dimensiones del dominio de datos. Por tanto, el cometido es reemplazar las variables originales por un conjunto menor de variables transformadas.

Razones para utilizar extracción de características [28, Página 464]:

1. Reducir la cantidad de datos de entrada.
2. Proveer un conjunto de características relevantes para el clasificador, obteniendo una mejora en la eficacia.
3. Reducir redundancia.
4. Obtener nuevos significados de variables transformadas o características que describen los datos, guiando a un mejor entendimiento de los procesos de generación de datos.
5. Producir una representación de datos en pocas dimensiones, con un mínimo de información perdida. De esta forma se permite visualizar los datos para que puedan relacionarse y estructurarse fácilmente.

2.4. Clasificadores

En esta sección se presentan tres de los métodos más utilizados como clasificadores.

2.4.1. Árboles de decisión

Es natural e intuitivo clasificar un patrón a través de una secuencia de preguntas, donde la siguiente pregunta depende de la respuesta a la pregunta actual.

Tal secuencia de preguntas se muestra en un árbol de decisión, empezando por convención en la cima, el nodo raíz, y conectando por sucesivos aristas o ramas a otros nodos, hasta que llegamos a los nodos terminales u hojas, que no tienen más aristas.

La clasificación de un patrón particular comienza en el nodo raíz, quien pregunta por el valor de una propiedad en particular del patrón. Los diferentes aristas desde el nodo raíz, corresponden a diferentes posibles valores. Se debe seguir el enlace apropiado a la respuesta. En estos árboles, los enlaces deben ser mutuamente excluyentes y exhaustivos, es decir uno y sólo un enlace va a ser tomado.

El siguiente paso es tomar la decisión en el nodo del sub árbol, que se puede considerar la raíz del nuevo sub árbol. Se debe continuar de esta forma hasta alcanzar el nodo hoja, que no tiene más preguntas. Cada nodo hoja representa una categoría y el patrón testeado es asignado a la categoría alcanzada.

Se puede ver en la figura 2.13 que uno de los beneficios de utilizar árboles frente a otros clasificadores, tales como redes neuronales, es la interpretación. Es sencillo representar información en un árbol como expresiones lógicas.

En primer lugar, podemos interpretar fácilmente la decisión de cualquier patrón de prueba en particular, como la conjunción de decisiones a lo largo de la ruta a su nodo hoja correspondiente.

En segundo lugar, algunas veces es posible conseguir interpretaciones claras de las categorías en sí mismas, mediante la creación de descripciones lógicas utilizando conjunciones y disyunciones.

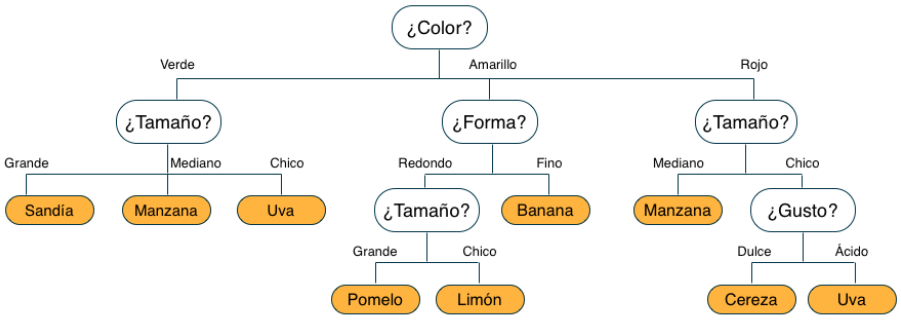


Figura 2.13: Clasificación en un árbol de decisión básico de arriba abajo. *Las preguntas de cada nodo corresponden a una propiedad de un patrón en particular, y los aristas que salen de ellos corresponden a lo posibles valores. Se visitan sucesivos nodos hasta que se llega al nodo terminal que contiene la etiqueta de la categoría. La misma pregunta puede aparecer en diferentes lugares del árbol, y en los diferentes lugares la misma pregunta puede tener diferente cantidad de ramas. Además, varios nodos hoja, mostrados en anaranjado, pueden pertenecer a la misma categoría.*

Otra ventaja de los árboles de decisiones es la rápida clasificación, empleando una secuencia de simples consultas.

2.4.2. Knn - K Vecinos más cercanos

Se plantea el caso hipotético donde se encuentran 3 características definidas y se cuenta con una cantidad reducida de puntos (o píxeles), tal que se conoce sus valores correspondientes a cada característica y su clasificación. Se desea distribuir estos puntos en un dominio donde cada dimensión corresponde a cada característica, donde se tendrá entonces en este caso un dominio en 3 dimensiones. Debido a que las características se eligen de tal forma que a partir de ellas se pueda deducir su clasificación (discriminar entre una clase y la otra), es de esperar que los píxeles de la misma clase se encuentren en cercanía en el dominio de características planteado. Es aquí la idea básica de la técnica *Knn - Vecinos más próximos*, está técnica consiste en clasificar el punto entrante a partir de su ubicación espacial en el dominio de características, y clasificándolo en función de sus vecinos más próximos que se hallan en dicho dominio.

Knn como un clasificador

Haciendo referencia a [10, Pág 124-126], se extenderá la definición de Knn, adaptándolo al problema de clasificación dentro del contexto de *Reconocimiento de patrones*, ya que éste método es utilizado también como un clasificador.

Tomando cómo ejemplo el reconocimiento de píxeles de hojas y manzanas. Supongamos que situamos en un espacio de 2 dimensiones a los píxeles que pertenecen a hojas y manzanas rojas. Una dimensión corresponde al valor de Matiz (H) y la otra dimensión a un indicador de 0 a 1 donde se especifica el grado de textura en el entorno del punto. Para el caso de las hojas, presentan una textura rugosa, próximo a 1, y para las manzanas una textura lisa, próximo a 0.

Para cada punto entrante, se deberá clasificar si es punto (o píxel) de hoja o de manzana. Suponga entonces que se debe clasificar un píxel entrante con el conocimiento de ciertos puntos ya clasificados, según se muestra en la figura 2.14.

Considerando la distancia euclidiana, el punto vecino más próximo al entrante, es clasificado como manzana, color rojo. El segundo vecino más próximo, también es manzana, color rojo, pero el tercer vecino más próximo es clasificado como hoja, color verde.

La idea de Knn es clasificar el punto entrante basándose en los K vecinos más próximos. Para esto se toma los K vecinos más cercanos, se obtiene la clase a la que pertenece cada uno, hoja o manzana, para luego definir el punto entrante como perteneciente a la clase a la que pertenece la mayoría de los K puntos tomados.

En la figura 2.15.a), se observa los vecinos más próximos al píxel entrante. Si se tomara K=1, el 1er vecino más próximo se encuentra clasificado como manzana, por tanto el píxel entrante también se clasificaría como manzana. Si K=3, se tomarían los 3 vecinos más próximos, dos de esos píxeles pertenecen a la clase de manzanas, y uno a la clase de hoja. En este caso, la clase manzana sigue siendo a la que pertenece la mayoría, por tanto el píxel entrante también se clasificaría nuevamente como manzana.

Se seguiría clasificando como manzana inclusive tomando a K=4, y K=5. Por tanto es coherente que el píxel entrante sea clasificado como manzana.

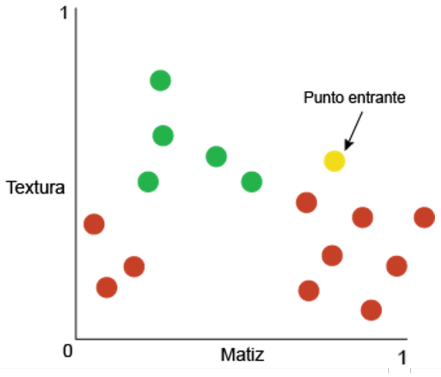


Figura 2.14: Ejemplo clasificación Knn. *Píxeles distribuidos en un espacio de 2 dimensiones, ubicados según los valores de matiz (H) y textura. Los píxeles verdes son clasificados como píxeles de hojas, y los rojos a manzanas. El punto amarillo es el píxel entrante que se debe clasificar.*

Suponga que ahora el píxel entrante es según la figura 2.15.b). En este caso si se toma K=1, el píxel entrante será clasificado como manzana. Si se tomara K=2, el 2do vecino más próximo es clasificado como hoja, por tanto habría un empate, y se debería tomar alguna estrategia como forma de desempate. Por esta razón es apropiado tomar K como número impar, para así evitar empate entre clases. Si K=3, predomina la clase manzana, si K=4, habría otro empate, y si K=5 predomina nuevamente la clase manzana.

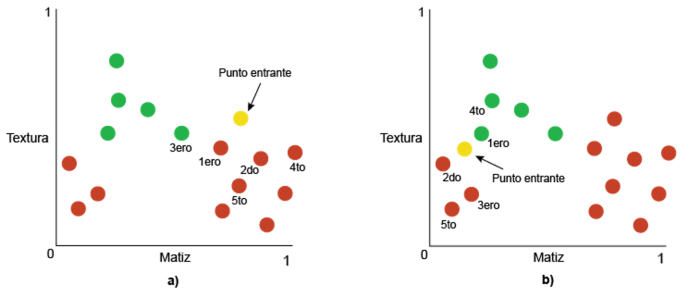


Figura 2.15: Los vecinos más próximos al punto entrante, tomando como distancia la Euclidiana.

Elección de K

La elección del valor de K apropiado es uno de los puntos más críticos del método Knn, ya que la eficacia y eficiencia del método depende de este parámetro.

Si se tomara K muy chico, se estaría frente a clasificaciones muchas veces erróneas para aquellos píxeles que se encuentran en los valores límites entre una clase y otra. Si se tomara un K muy grande, se puede dar o no que la eficacia sea mejor, pero la eficiencia no, ya que tiene un alto coste de procesamiento el hallar los K vecinos más

próximos para K grandes. Por tanto el K más adecuado, es aquel valor mínimo para el cuál el método tiene una tasa de eficacia aceptable.

Normalización de las características

Uno de los principales desperfectos es el de calcular distancias directamente del conjunto de datos, donde cada característica es un tipo de variable que puede tener distinta naturaleza. En el ejemplo anterior los valores numéricos de la textura varían entre 0 y 1, mientras que para la matiz varía entre 0 y 180. Como se decidió utilizar la distancia euclidiana, es necesario normalizar el espacio de características antes de comenzar las etapas de entrenamiento /clasificación.

Una de las formas para normalizar, es identificando el máximo y mínimo dentro del conjunto de datos:

$$X_N = \frac{X - X_{min}}{X_{max} - X_{min}}$$

2.4.3. SVM - Support Vector Machine

Dado un conjunto de muestras de entrenamiento, donde los datos se encuentran marcados como pertenecientes a una u otra clase, un algoritmo de entrenamiento *SVM - Support Vector Machine* (Máquina de vectores de soporte) construye un modelo para ajustar cada una de las clases. Intuitivamente, es un modelo que representa a los puntos de entrenamiento en el espacio, separando las clases por un margen lo más amplio posible. Cuando las nuevas muestras se ponen en correspondencia con dicho modelo, se clasifican en una u otra clase según de qué lado del espacio caen.

Además de realizar la clasificación lineal, SVM puede ser utilizado para clasificar de forma eficiente una clasificación no lineal utilizando lo que se define cómo Kernel, mapeando de forma implícita las entradas a un espacio de dimensionalidad superior.

Como en la mayoría de los métodos de clasificación supervisada, los datos de entrada son vistos como un vector d-dimensional.

SVM busca un hiperplano que separe de forma óptima a los puntos de una clase de la de otra, que eventualmente han podido ser previamente proyectados a un espacio de dimensionalidad superior.

En ese concepto de "separación óptima" es donde reside la característica fundamental de las SVM: estos tipos de algoritmos buscan el hiperplano que tenga la máxima distancia (margen) con los puntos que estén más cerca de él mismo. De esta forma, los puntos del vector que son etiquetados con una categoría estarán a un lado del hiperplano y los casos que se encuentren en la otra categoría estarán al otro lado.

Función Kernel

La manera más simple de realizar la separación es mediante una línea recta, un plano recto o un hiperplano N-dimensional.

Desafortunadamente los universos a estudiar no se suelen presentar en casos idílicos de dos dimensiones, sino que un algoritmo SVM debe tratar con más de dos variables

predictoras, curvas no lineales de separación y casos donde los conjuntos de datos no pueden ser completamente separados.

La representación por medio de funciones Kernel ofrece una solución a los problema en los cuales las clases no son linealmente separables, proyectando la información a un espacio de características de mayor dimensión. Es decir, se mapea "virtualmente" el espacio de entradas X a un nuevo espacio de características de mayor dimensionalidad (ver figura 2.16), como se define a continuación:

$$F = \{\varphi(x)|x \in X\},\ x = \{x_1, x_2, .., x_n\} \rightarrow \varphi(x) = \{\varphi(x_1), \varphi(x_2), .., \varphi(x_n)\}$$

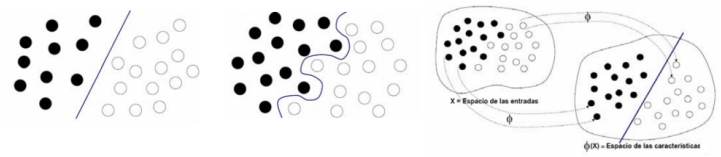


Figura 2.16: *a) Datos linealmente separables en el espacio de las entradas, b) Datos no linealmente separables en el espacio de las entradas, c) Datos linealmente separables en el espacio de las características.*

Tipos de función Kernel

- Polinomial-homogénea: $K(x_i, x_j) = (\gamma x_i^T x_j), \gamma > 0$. En la figura 2.17 se ilustra el resultado de aplicar este tipo de Kernel.

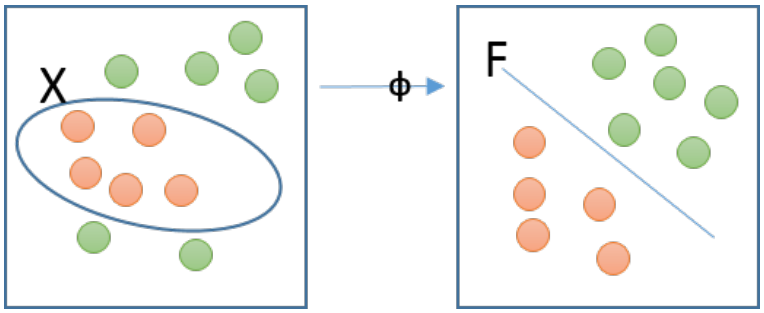


Figura 2.17: *Ilustración de la distribución de puntos luego de aplicar el Kernel de tipo Polinomial-homogénea.*

- Perceptron: $K(xi, xj) = ||x_i - x_j||$. En la figura 2.18 se ilustra el resultado de aplicar este tipo de Kernel.

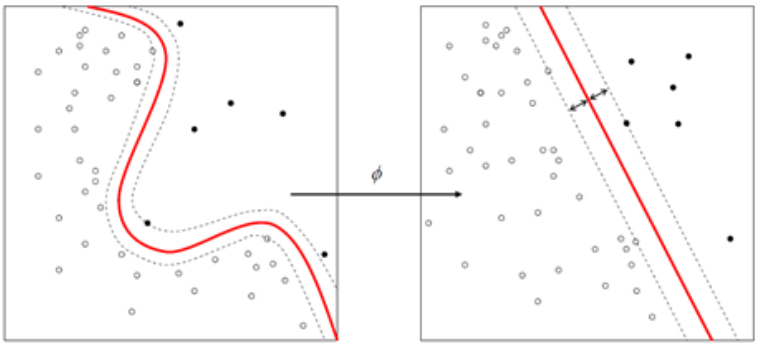


Figura 2.18: *Ilustración de la distribución de puntos luego de aplicar el Kernel de tipo Perceptron.*

- Función de Base Radial Gaussiana: separado por un hiperplano en el espacio transformado. $K(x_i, x_j) = e^{-\sigma||x_i - x_j||^2}, \sigma > 0$. Aquí el parámetro σdetermina qué tan suave es la función en el espacio de base dimensional, una ilustración de esto se puede observar en la figura 2.19.

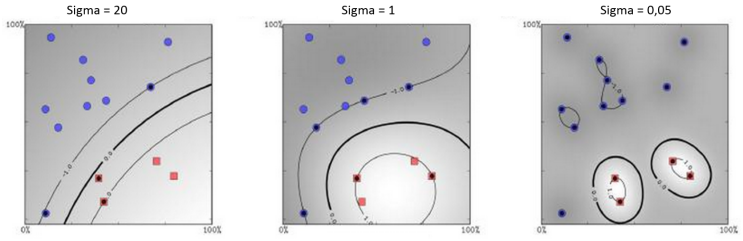


Figura 2.19: *Ilustración de la distribución de puntos luego de aplicar el Kernel de tipo Base Radial Gaussiana.*

2.5. Técnicas para el análisis de formas

Se plantea el caso hipotético donde en una imagen se encuentra un conjunto de x frutos, el desafío de reconocer los píxeles que forman parte del conjunto no es suficiente para reconocer la cantidad de frutos allí presentes. Por tal motivo además de identificar los píxeles que forman parte de los frutos, es necesario identificar la ubicación y cantidad de frutos.

A continuación se hará una introducción a las herramientas utilizadas en los métodos de detección de frutos.

Operaciones morfológicas

Las operaciones morfológicas son la rama del procesamiento de imágenes que estudia las estructuras geométricas[12, Página 1].

La idea básica de las operaciones morfológicas consiste en el estudio mediante una imagen con una forma predefinida simple, si esta forma es contenida o no en las formas de la imagen a estudiar. Esta imagen predefinida que se utiliza es la llamada **elemento estructurante.**

A modo de ejemplo, en la figura 2.20 se observa una imagen binaria (a) y un cuadrado como elemento estructurante(b), donde el elemento estructurante en (a) se encuentra contenido a la izquierda, mientras en la otra posición, a la derecha, no esta contenido.

Para obtener información estructural se ubica el elemento estructurante en diferentes posiciones y se determina si está o no contenido en la imagen. Dicha información depende del tamaño y forma del elemento elegido. Por lo tanto la información obtenida dependerá de la elección del mismo.

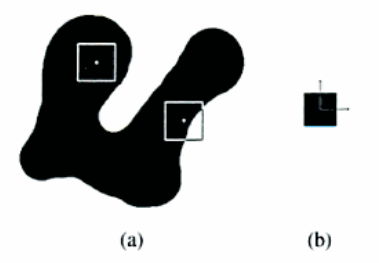


Figura 2.20: *(a) Imagen a analizar (b) Elemento estructurante.*

A continuación se describirán dos de las operaciones morfológicas más utilizadas por contribuir a la eliminación de ruido y en la mejora de la definición de las formas geométricas.

Para ello se definen dos conjuntos de píxeles: los negros constituyen el conjunto de píxeles de primer plano y los restantes el conjunto de fondo. Se utilizará la letra A para referirse al conjunto de píxeles de primer plano, y la letra B para el conjunto que representa al elemento estructurante.

La traslación de un conjunto A por un punto x es denotado por A_x y se define por:

$$A_x = \{a + x : a \epsilon A\}$$

Geométricamente se encuentra ilustrado en la figura 2.21.

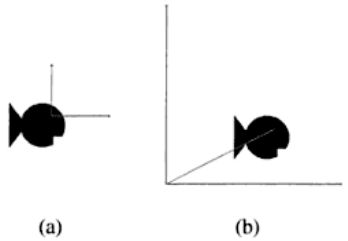


Figura 2.21: *a) Imagen A, b) A trasladada por el vector x.*

La rotación de un conjunto A con centro en un puto o, un ángulo α es una transformación geométrica que hace corresponder a cada punto $p \epsilon A$ otro punto p' tal que: $\overline{po} = \overline{p'o}$ y $\widehat{pop'} = \alpha$

Erosión

La erosión del conjunto A por el conjunto B, es denotada como $A \ominus B$ y se define como:

$$A \ominus B = \{x : B_x \subset A\},$$

Es decir, $A \ominus B$ consiste en los puntos x para los que la traslación B por x cae dentro de A.

A modo de ejemplo en la figura 2.22 se muestra en gris claro la imagen previa a realizar la erosión y en gris oscuro la imagen resultado luego de realizar la erosión usando el elemento estructurante (b).

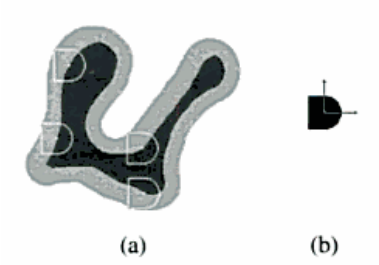


Figura 2.22: *a) Imagen erosionada en negro, el área erosionada en gris, b) Elemento estructurante.*

Como resultado, la erosión retorna figuras de menor tamaño.

El elemento estructurante que se elija, en cuanto a forma y tamaño es un factor importante a la hora de aplicar operaciones morfológicas, ya que puede generar deformaciones de la imagen de entrada, o eliminaciones no deseadas. Uno de los principales uso de la erosión es para la eliminación o reducción de ruido, ya que el ruido en imágenes binarias se constituye básicamente de pequeñas regiones de píxeles aisladas, y estas

desaparecerán al aplicar la erosión para aquellas regiones cuyas área sea menor a la del elemento estructurante.

Dilatación

La segunda operación básica dentro de las operaciones morfológicas es la **dilatación**. Se define a partir de la erosión como su conjunto complemento.

La dilatación del conjunto A por B se denota como $A \otimes B$ y se define como:

$$A \otimes B = (A^c \ominus \check{B})^c.$$

Para dilatar A por B, B es rotado alrededor del origen para obtener $\check{B}$, A^c es erosionado por $\check{B}$, y luego se toma el complemento de la erosión. Cómo se ve en la figura 2.23 donde B es un disco, si B contiene al origen, entonces la dilatación de A por B tiene como resultado una expansión de A.

La dilatación expande las figuras contenidas en la imagen original.

Uno de los posibles usos de la dilatación es la eliminación de huecos en la figura, dependiendo del elemento estructurante que se tome se reducen en mayor o menor grado.

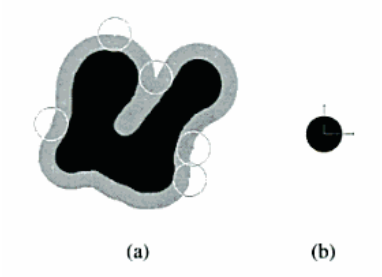


Figura 2.23: *a) Imagen original en negro, dilatación en gris, b)Elemento estructurante.*

2.5.1. Detección a partir de círculos de Hough

La transformación de Hough es una técnica que puede ser utilizada para identificar una determinada forma geométrica dentro de una imagen [8, Cap. 4]. La transformación de Hough se utiliza comúnmente para la detección de curvas regulares como líneas, círculos, elipses, etc. Una transformación generalizada de Hough puede ser aplicada donde una simple descripción analítica de una característica no es posible. La principal ventaja es que es tolerante a discontinuidades y formas geométricas con imperfecciones que se asemejan a una cierta forma geométrica.

La idea de Hough para detección de líneas, es que para cada pixel se obtengan las coordenadas (x, y), y se almacenen los parámetros que definen las rectas a las que ese pixel pertenece. Repitiendo este procedimiento con todos los píxeles de la imagen, contribuirá a resaltar los parámetros que definen las rectas físicas presentes en la imagen.

En la figura 2.24 se visualiza las posibles soluciones a éste problema. Aquí la falta de conocimiento sobre el número de líneas convierte esto a un problema sin restricciones.

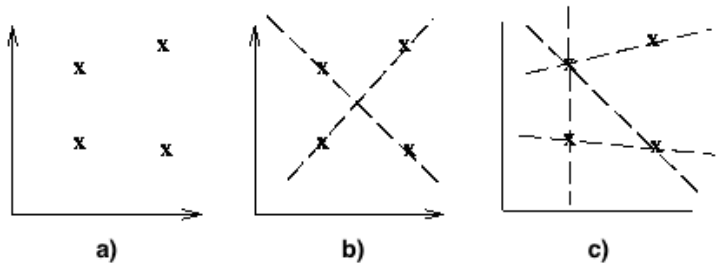


Figura 2.24: *a) Coordenadas de puntos, b) y c) posibles rectas formadas a partir de los puntos.*

Se describirá a continuación el método de Hough, partiendo del reconocimiento de líneas para su fácil comprensión. Sea una recta definida por la ecuación:

$$x.cos(\phi) + y.sen(\phi) = r$$

Se dispone de un espacio de dos dimensiones de parámetros, donde en el eje horizontal se encuentran los valores de ϕ y en el eje vertical los valores de r. Para cada píxel de la imagen, de coordenadas (x_i, y_i), se incrementará en uno al punto (ϕ_i, r_i) que corresponde a la solución de la recta que pasa por el punto (x_i, y_i). Luego de iterar por todos los píxeles de la imagen, si hay una recta física presente, entonces habrá un punto (ϕ_f, r_f) en el espacio de parámetros con un valor acumulado mucho mayor que el resto. La coordenada de ese punto (ϕ_f, r_f) será la solución a la ecuación de la recta física presente en la imagen.

Para hallar círculos se realiza el mismo algoritmo que para recta, solamente que la ecuación paramétrica es el de una circunferencia:

$$(x - a)^2 + (y - b)^2 = r^2$$

El espacio de parámetros es de dimensión 3, representando en cada eje los valores de a, b, r.

La transformación de Hough se caracteriza por ser tolerante al reconocimiento de formas geométricas incompletas, con imperfecciones y saltos. Por ejemplo, en a) de la figura 2.25, se reconoce el borde exterior del iris del ojo aún cuando el mismo no se encuentra visible en su totalidad. La detección del círculo se realiza a partir de que el acumulador de celdas es lo suficientemente alto.

Esta característica de Hough es una gran ventaja a la hora del reconocimiento de frutos permitiendo reconocer manzanas no ocluidas, levemente ocluidas o en un conjunto. Ver figura 2.26

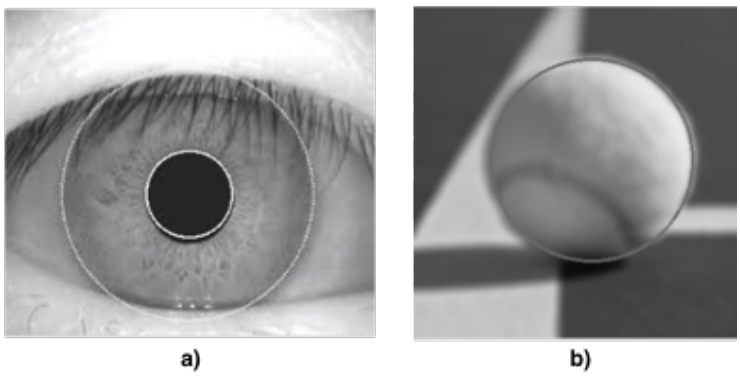


Figura 2.25: Círculos de Hough. *a) y b) Círculos en la imagen reconocidos por Hough utilizando la ecuación paramétrica de circunferencia.*[3][4]

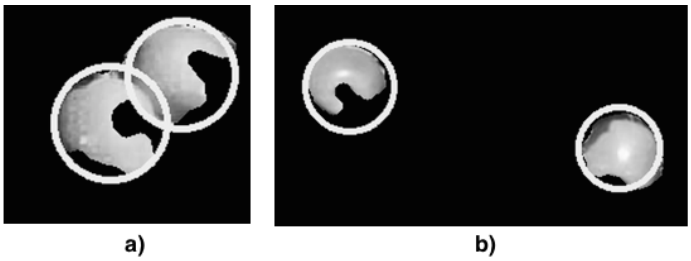


Figura 2.26: Ejemplo detección de círculos. *a) Círculos en conjunto de manzanas, b) Círculos en manzanas incompletas*

Capítulo 3

Solución propuesta

En este capítulo se presenta la solución desarrollada en el proyecto para el reconocimiento de manzanas. En la sección 3.1 se plantean los requerimientos del sistema, en la sección 3.2 se describe el hardware y las plataformas utilizadas, en la sección 3.3 se describe el sistema desarrollado para el reconocimiento de pixeles manzana y por último en la sección 3.4 el sistema desarrollado para el conteo de manzanas.

3.1. Requerimientos

Debido a las características cambiantes propias del entorno para el que fue diseñado el sistema, el mismo debe ser robusto en cuanto a cambios de iluminación. Las imágenes son adquiridas en una huerta con luz natural lo cual hace que sea un entorno muy cambiante en cuanto a intensidad, dirección de dicha luz y sombras.

A su vez, se quiere que el sistema no esté limitado a un tipo de poda en particular. Como se ve en la figura 3.1, el tipo de poda que se le realice a los arboles genera diferente oclusión de follaje y diferente densidad de manzanas por rama.

El sistema debe reconocer también diferentes especies de manzanas, es decir manzanas de distintas tonalidades como se puede ver en la figura 3.2.

Por las aplicaciones que puede tener a futuro el sistema, se quiere que se reconozcan las manzanas en un tiempo cercano a tiempo real.

Por último se quiere que el sistema funcione con una gran variedad de dispositivos de adquisición de imágenes, dando la posibilidad así de ajustarse a los dispositivos que el usuario disponga.

3.2. Plataforma utilizada

Hardware

Como sensor de adquisición se utilizaron varias cámaras:

- Cámara Nikon D3200
- Filmadora Sony DCR SR-68
- Cámara Olimpus XZ-1

El motivo principal por lo que se utilizaron diferentes cámaras es que las imágenes fueron tomadas por distintos integrantes en dos visitas a la estación experimental "Wilson Ferreira Aldunate", INIA Las Brujas.

(a) Tall Spindle (b) Bi-Axis

Figura 3.1: *Muestra de distintos tipos de podas.*[26]

Figura 3.2: *Distintas especies de manzanas.*

Software

Para el desarrollo del software se utilizó la plataforma Eclipse en C++ con la biblioteca OpenCV.

OpenCV [6] es una biblioteca open source que incluye cientos de algoritmos de visión por computador. Fue diseñada para ser computacionalmente eficiente y con un enfoque fuerte de procesamiento en tiempo real. Está escrita en C y C++ utilizando C

optimizado y permite tomar ventaja de múltiples procesadores. Cuenta con la ventaja de tener una documentación completa y al día, incluyendo además variados ejemplos. Su gran cantidad de usuarios y disponibilidad de información la hacen una excelente opción.

Si bien OpenCV se puede utilizar en varios lenguajes, se recomienda el uso en C++ por ser el lenguaje en el cual esta implementada. Es por esto que es el primero en incorporar nuevas funcionalidades.

3.3. Reconocimiento de píxeles manzana

En esta sección se describirán las soluciones que se llevaron a cabo para el reconocimiento de píxeles manzana a partir de imágenes.

Para este desarrollo se siguió el esquema clásico de reconocimiento de patrones presentado en el marco teórico. Primero se adquirieron las imágenes, luego se extrajeron las características y por último se clasificó según las mismas, utilizando diferentes métodos.

En la sección de Evaluación experimental se evaluará la eficiencia de cada uno de estos métodos.

3.3.1. Extracción de características

Como primera aproximación a la solución se eligió utilizar para la clasificación la característica más notoria, el rango del matiz (H). Luego, observando que existen otras características que diferencian a las manzanas del resto de los elementos, se decidió unir las mismas a la clasificación según matiz. Primero se agregó la saturación para eliminar los pixeles que usando H clasifican, pero tienen poca saturación, es decir son blancos o cercanos al mismo, agregando esto se lograron descartar píxeles pertenecientes a los troncos. Luego se agregó la densidad de bordes viendo que las manzanas presentan una superficie lisa en comparación con el pasto y las hojas.

Si bien se puede observar teóricamente que esto es correcto, se realizaron pruebas prácticas para comprobar si existían mejoras reales al agregar el análisis de más características. Mediante las pruebas se verificó que efectivamente existen mejoras, estos resultados se pueden ver en la sección Evaluación experimental.

Color

Como se puede observar en la figura 3.3, existe una gran diferencia entre el color rojo de las manzanas y los demás colores del huerto como ser tierra, pasto, ramas, follaje, alambres y cielo entre otros.

Dado que las imágenes son adquiridas bajo diferentes condiciones de luminosidad, es importante elegir un espacio de colores donde el tono no cambie al cambiar la misma. Uno de los espacios que cumple esta propiedad es el espacio de colores HSV, donde el matiz (H) no se ve alterado por los cambios de luminosidad. Además, comparado con RGB, HSV facilita la percepción del color, la componente H tiene una relación más cercana a la manera que las personas perciben el color.

Figura 3.3: *Colores presentes en un huerto.*

Si bien se determinó para cada técnica de clasificación el matiz de las manzanas que daba mejores resultados, los rojos en el espacio HSV, como se vio en el marco teórico, se encuentran aproximadamente entre 0° y 30° y entre 350° y 360°, esto se puede visualizar en la figura 3.4.

Figura 3.4: *Ejemplo de segmentación usando H. A la izquierda se encuentra la imagen de la base de datos, a la derecha la imagen luego de realizar segmentación según H.*

Otra característica a tener en cuenta es la saturación (S). Dado que el matiz no está definido para píxeles sin saturación cómo blanco, gris o negro, se debe tener en cuenta con el objetivo de excluir algunos objetos del fondo, como se ve en la figura 3.5.

Densidad de bordes

Otra característica que se puede observar es que las manzanas presentan menos bordes en su superficie que los que presenta el pasto, hojas o troncos.

En la figura 3.6 se puede observar cómo mientras las manzanas casi no presentan bordes, las hojas y pasto sí.

Se puede ver fácilmente que en un borde la intensidad de los píxeles cambia de forma notoria. Una buena forma de expresar estos cambios es usando derivadas, un cambio grande en un gradiente indica un cambio grande en la imagen. Por lo que un método

para detectar bordes en una imagen es ubicar los pixeles donde el gradiente es mayor que el de sus vecinos, o para generalizar donde es mayor que un determinado número.

Figura 3.5: *Ejemplo de segmentación utilizando H y S. La imagen del centro es segmentando según H, mientras que la de la derecha usando H y S.*

Sobel

El operador Sobel calcula el gradiente de la intensidad de una imagen en cada píxel. Así, para cada punto, este operador da la magnitud del mayor cambio posible, la dirección de éste y el sentido desde oscuro a claro. El resultado muestra cuan abruptamente o suavemente cambia una imagen en cada píxel analizado y, en consecuencia, cuán probable es que éste represente un borde en la imagen.

Matemáticamente, el operador utiliza dos Kernels de nxn elementos para aplicar convolución a la imagen original y así calcular aproximaciones a las derivadas. Se utiliza un kernel para los cambios horizontales y otro para los verticales.

Suponiendo que la imagen a la que se le quieren obtener los bordes es I:

Se calculan las dos derivadas:

1. Cambios horizontales: Se computa convulsionando I con el kernel G_x de tamaño impar. En este caso utilizamos un kernel de tamaño 3, siendo
$$G_x = \begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix} * I$$

2. Cambios verticales: Se computa convulsionando I con el kernel G_y de tamaño impar. En este caso utilizamos un kernel de tamaño 3, siendo
$$G_y = \begin{matrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{matrix} * I$$

Para cada punto de la imagen, se calcula una aproximación del gradiente en dicho punto combinando ambos resultados: $G = \sqrt{G_x^2 + G_y^2}$. Como se sugiere en la documentación de Sobel, muchas veces por simplicidad se utiliza la siguiente ecuación para combinar el gradiente: $G = |G_x| + |G_y|$.

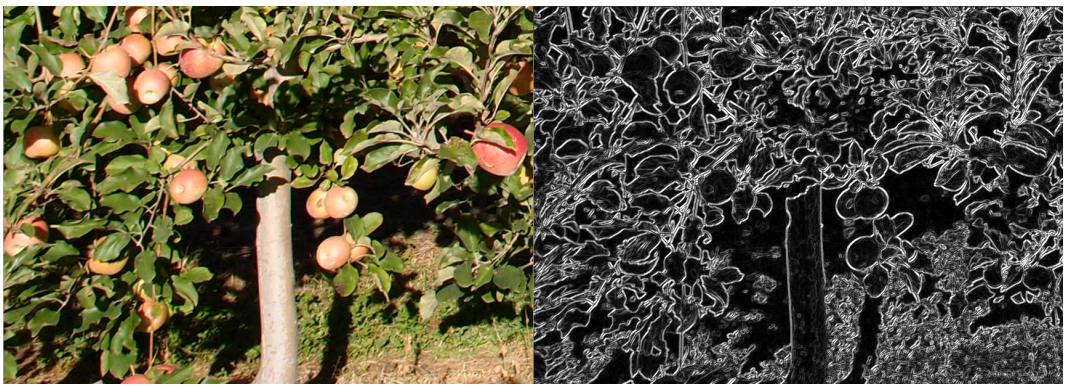


Figure 3.6: Ejemplo de aplicar Sobel. *A la izquierda se ve una imagen de la base de datos, a la derecha la imagen luego de aplicarle Sobel.*

3.3.2. Clasificación

A continuación se describen las tres técnicas utilizadas para la clasificación de los píxeles manzana, árbol de decisión, K vecinos más próximos y Support Vector Machine.

3.3.2.1 Árbol de decisión

Una posible solución para el reconocimiento de píxeles de manzanas, es utilizar el árbol de decisión de la figura 3.7. Los parámetros *Hmin* y *Hmax* son los límites considerados para el matiz de los tonos de rojo, *Smin* es el mínimo valor de saturación que puede tomar un píxel para ser considerado manzana, *BordesMax* es el valor límite para considerar si la densidad de bordes del píxel estudiado está por debajo del límite para ser considerado manzana. Todos estos valores son ajustados durante el ajuste de parámetros.

Realizando la clasificación de pixeles mediante el árbol de decisión, se crea una nueva imagen donde a cada pixel se le asigna el valor 1 o 0 según si el píxel de la imagen se clasificó como manzana o no. Esto genera una máscara la cual indica entonces cuáles píxeles pertenecen a manzanas y cuáles no.

Ajuste de parámetros:

El ajuste de los parámetros se realizó de forma iterativa debido a que estudiar todas las combinaciones de valores posibles requeriría un tiempo muy grande. Por tal motivo primero se estimaron los rangos de valores del matiz para los que se consiguen mejores resultados. Luego, utilizando un rango cercano al hallado, se determinó el rango de Saturación (S). Por último se realizó el mismo procedimiento para obtener el rango de densidad de bordes, este flujo se puede visualizar en la figura 3.8.

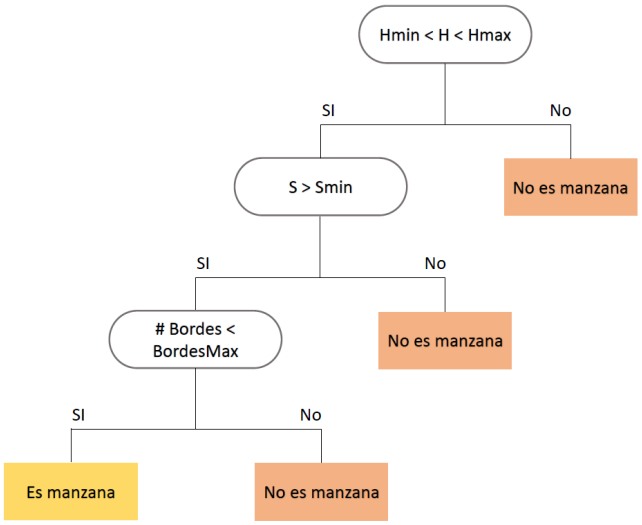


Figura 3.7: Diagrama del Árbol de decisión desarrollado.

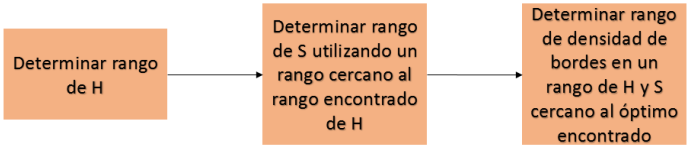


Figura 3.8: Ajuste de parámetros en el árbol de decisión. *Esquema del proceso de ajuste de valores válidos para los rangos de las características extraídas.*

3.3.2.2 Knn - K Vecinos más próximos

Otra posible solución para el reconocimiento de píxeles manzana es utilizar el método K Vecinos más próximos, Knn. Para ejecutar este método, se debe brindar un conjunto de datos de entrenamiento, donde se encontrará para cada píxel el valor correspondiente de cada característica y su respectiva clasificación. Para este caso se proponen las siguientes 2 clases posibles a las que pertenecerá un píxel:

- Manzana: píxel que corresponde a una manzana en la imagen.
- No manzana: píxel que no corresponde a una manzana.

Se utilizarán las 3 características descritas en Extracción de características, H, S y Densidad de bordes. Por tanto para el método Knn, el conjunto de datos de entrenamiento consistirá de:

- Una matriz de 3 columnas y N filas, siendo N la cantidad de píxeles del conjunto de datos. Cada fila i representa en sus celdas los valores respectivos de H, S y Densidad de bordes para el correspondiente píxel i.
- Un vector de tamaño N, donde el valor contenido en el índice i corresponde a la clase a la que pertenece el correspondiente píxel i.

Se puede ver en la figura 3.9, la estructura del conjunto de datos de entrenamiento.

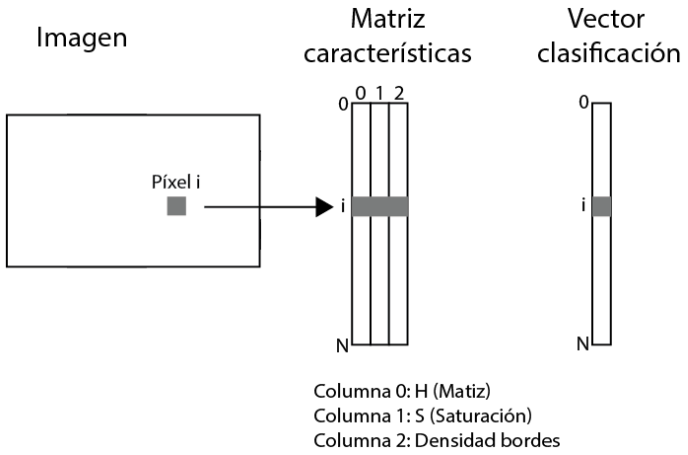


Figura 3.9: Estructura de datos de entrenamiento. *A partir de la imagen se extrae para cada píxel su correspondiente valor de H, S y Densidad de bordes, los cuales serán almacenados en la matriz de características. En el vector clasificación se almacena la clase a la que corresponde el píxel, Manzana o No manzana.*

FLANN - Fast Approximate Nearest Neighbor Search

FLANN es una biblioteca que contiene una colección de algoritmos optimizados para una rápida búsqueda de vecinos en conjuntos de datos de gran dimensión de características [2]. La idea básica es construir una estructura de datos donde almacenar los índices correspondientes a los puntos de entrenamiento con un criterio de ordenación.

Árbol KD

Árbol KD es una estructura de datos para organizar puntos en un espacio K-dimensional, en este caso de tres dimensiones. Tiene varias aplicaciones, tales como la búsqueda de distancias entre puntos en espacios multidimensionales [21].

Nota: No confundir la K de Árbol KD, con K de Knn, es coincidencia que ambos utilicen la misma letra en su definición.

Es un caso particular de Partición binaria del espacio (BSP), donde cada nodo no-hoja, además de representar un punto, implícitamente representa el corte en dos de un hiperplano en una de las dimensiones del espacio, la mitad izquierda del hiperplano se representa por el subárbol a la izquierda del nodo padre, y la mitad derecha por el subárbol a la derecha. Por tanto, esta estructura se amolda perfectamente para la búsqueda de Knn.

Luego de obtener los K índices de los vecinos más próximos al punto a evaluar, para cada índice j en $[0..K]$ se obtiene su clasificación almacenada en la j-ésima fila del Vector clasificación. Luego de iterar para todos los K índices, finalmente se obtiene la clase predominante, con la que se clasificará el punto entrante. En la figura 3.10 se ilustra el proceso de búsqueda y obtención de la clasificación de los vecinos al punto entrante i.

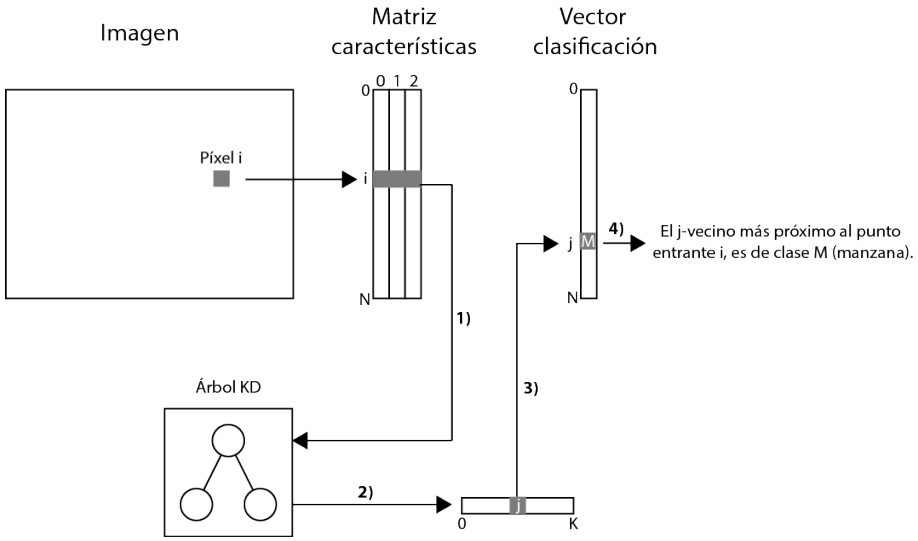


Figura 3.10: Búsqueda y obtención Árbol KD. *1) Se busca en Árbol KD los índices de los K vecinos más próximos al punto con índice i. 2) Se obtiene un vector con K índices de los vecinos más próximos. 3) Para cada j del vector obtenido en el paso anterior, se obtiene la clasificación del punto.*

A continuación se describe el algoritmo de búsqueda, que corresponde a las estructuras de la figura 3.10.

```
-Para cada punto i a clasificar:

    -Encontrar en Árbol XD los índices de los K vecinos más próximos
    al punto i, cuya cercanía será calculada a partir de los valores
    de las 3 características del punto.
    -Para cada índice devuelto:

        -Encontrar su clasificación en el vector de clasificación.
            -Si es clasificado cómo manzana:

              -Se suma un voto a que el píxel i sea clasificado cómo manzana.
            -Si no es manzana:

              -Se suma un voto a que el píxel i sea clasificado cómo fondo.
-El píxel i será clasificado según la mayoría de votos entre la clase
manzana y fondo.
```

Elección de K

Como se vio en el Marco teórico, la elección adecuada de K es el punto más crítico. Si se elige un K pequeño, se puede clasificar de forma errónea debido a la poca información que se tiene en el entorno local del punto entrante. Si se elige un K muy grande, los costes de tiempo y procesamiento son significativamente mayores, donde también puede clasificar de forma incorrecta ya que clasificaría en base a información global y no local. Se agrega la restricción de que K sea un número impar, para así evitar empates entre clases.

En la sección Evaluación experimental se evaluará tomando distintos valores de K impar, eligiendo el K que presente mejores resultados. Para optar por el mejor resultado, se tomará en cuenta la eficacia y eficiencia. Es decir, qué tanto ha acertado considerando también el tiempo que consumió.

3.3.2.3 SVM - Support Vector Machine

Otro desarrollo implementado para la clasificación de píxeles en manzana o no manzana es utilizando SVM.

Primero se preparan los datos para el entrenamiento al igual que se hace para Knn, luego se elige que kernel se va a utilizar, y se realiza el entrenamiento. Cuando ya se tiene el entrenamiento realizado, se pasa a realizar la clasificación.

Para la implementación de SVM se utilizaron las funciones que brinda OpenCV.

3.4. Conteo de manzanas

Luego de reconocidos los píxeles manzana, se deben identificar las manzanas presentes que se forman a partir de los mismos, determinando la ubicación y cantidad de manzanas en cada imagen. Para esto se utilizarán técnicas que se aplicarán sobre la imagen binaria resultado del reconocimiento de píxeles.

El proceso que se propone aplicar consta de tres etapas:

- Operaciones morfológicas: Primero se aplicarán operaciones morfológicas para la eliminación de ruido y el alisado de bordes de las regiones. Con esto se busca ayudar a la detección de círculos, y que sean descartadas regiones cuya área sea pequeña.

- Reconocimiento de círculos: Como segunda parte, se aplicarán técnicas de detección de círculos utilizando la transformación de Hough para círculos. Dado que las manzanas presentan una forma geométrica muy cercana a la de una esfera, en imágenes de dos dimensiones se verán como círculos, cualquiera sea la ubicación de donde se adquiera la imagen.

- Pos-procesamiento de círculos: Finalmente, luego de reconocidos los círculos, se estudiará el área de las regiones contenidas en ellos, así como los círculos intersectados. El fin de esto es descartar tanto la falsa detección de manzanas como la detección repetida.

3.4.1. Operaciones morfológicas

En las imágenes binarias resultado del reconocimiento de píxeles manzana se encuentran regiones con áreas de distintos tamaños, desde regiones de tamaño significante a otras que simplemente son un conjunto pequeño de píxeles ruido como se puede ver en la figura 3.11. El ruido es generado debido a follaje, ramas, y otros elementos presentes que por sus características se aproximan a los píxeles manzana, pero resultan ser regiones aisladas y pequeñas que deben ser descartadas.

Figura 3.11: *a) Imagen original b) imagen resultado de aplicar reconocimiento de píxeles. Allí se puede notar regiones con áreas de tamaño apreciable y contiguas, y otras más pequeñas y aisladas, ruido.*

Se propone el uso de las operaciones morfológicas descritas en el Marco teórico: erosión y dilatación. La erosión provoca la eliminación de ruido y la reducción de regiones pequeñas, pero también provoca un aumento de los huecos contenidos en las regiones de tamaño significante. Por otro lado la dilatación genera lo inverso, es decir la expansión de las regiones y disminución de huecos dentro de las mismas.

Por lo descrito en el párrafo anterior, es que se aplicarán cuatro veces operaciones morfológicas, cómo se visualiza en la figura 3.12. En una primera etapa se busca quitar el ruido sin perder el tamaño del área de las regiones de tamaño significante, y en la siguiente etapa se busca alisar los bordes y unir las regiones contiguas, manteniendo también el tamaño del área. En la figura 3.12 se visualiza un ejemplo del proceso.

Como elemento estructurante se utiliza una circunferencia, debido a que las manzanas en las imágenes presentan una forma bastante cercana a la misma. El tamaño de la elipse debe ser razonable para la eliminación del ruido pero cuidando de no eliminar regiones de interés. En la sección Evaluación experimental, se evaluará cuál es el tamaño más adecuado a partir de los resultados en las pruebas.

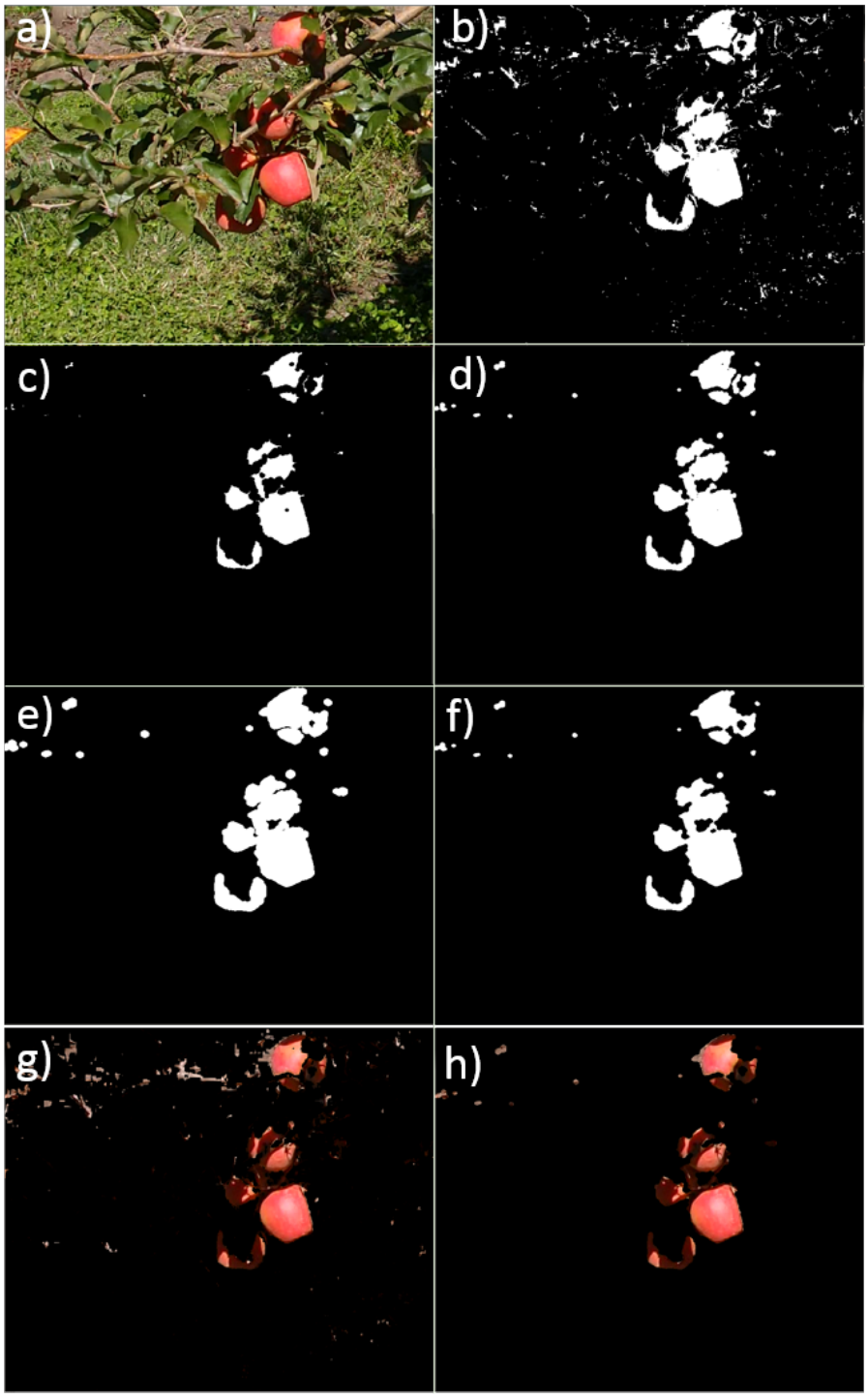


Figura 3.12: Erosión - Dilatación.*a) Imagen original. b)Máscara resultado del reconocimiento de píxeles. c) y d) Se aplica erosión y dilatación respectivamente para quitar ruido pero manteniendo el área de las regiones apreciables. e) y f) se aplica dilatación y erosión respectivamente para generar formas más uniformes y lisas. g)Máscara en color resultado del reconocimiento de píxeles. h) Máscara a color luego de aplicar erosión, dilatación, dilatación y erosión.*

3.4.2. Detección de círculos

Como ya se mencionó, debido a la forma de las manzanas, se propone reconocer los círculos presentes en la imagen binaria resultado del reconocimiento de píxeles.

Como método para reconocer los círculos presentes se propone utilizar la transformada de Hough para círculos.

3.4.3. Pos-procesamiento de círculos

Luego de obtener una colección de círculos, conociendo el radio y centro de los mismos, se debe estudiar cada uno para para detectar círculos erróneos.

Estudio del área mínima en círculos

Como ya se mencionó, las operaciones morfológicas se realizan para alisar los bordes de las regiones y para quitar ruido. Igualmente es posible que se detecten círculos erróneos como en las siguientes situaciones:

- Manzanas muy ocluidas generan áreas pequeñas pero no tanto como para ser descartadas por las operaciones morfológicas. Si además las mismas presentan bordes curvos, se detectará un círculo que si bien pertenece a una manzana, no representa la ubicación real de la misma. Además puede darse que una misma manzana al estar ocluida genere más de un área separada y curva, contando entonces una única manzana como varias. Esto se puede observar en la figura 3.13.

- Otra situación posible se visualiza en la figura 3.14, en este caso se detectó un círculo debido a que la región superior presenta una curvatura circular hacia afuera, lo cual provoca la detección del círculo indicado. Como resultado, se detecta un círculo vacío en su interior.

Para solucionar las situaciones mencionadas, se propone descartar círculos detectados cuya área sea menor que un área mínima. Esto quiere decir que para cada círculo, se estudiará el área de la región que incluye, y se descartará el círculo si dicha área es menor que un porcentaje definido, siendo 100 % el área total del círculo. En la parte Evaluación experimental, se evaluarán distintos umbrales para obtener el más adecuado.

Intersección de círculos

Como se ve en la figura 3.15, si una región presenta huecos circulares en su interior es posible que se detecte más de un círculo en ella.

Para evitar estas falsas detecciones se propone realizar el estudio del área en cada círculo, restando la región que intersecta con otros círculos ya estudiados. En el caso de la figura, donde primero se detecta el círculo superior, se debe estudiar el área del segundo círculo, restando el de la intersección señalada mediante la flecha roja. En este caso se descartará el segundo círculo porque el área es muy chica, si por lo contrario el área restante del segundo círculo fuera apreciable, no se descartaría, lo cual es razonable ya que se tomaría en cuenta como una segunda manzana.

Figura 3.13: *Debido a manzanas ocluidas, se puede observar marcadas por las flechas: regiones de manzanas de área pequeña, o varias regiones separadas que corresponden a una uníca manzana.*

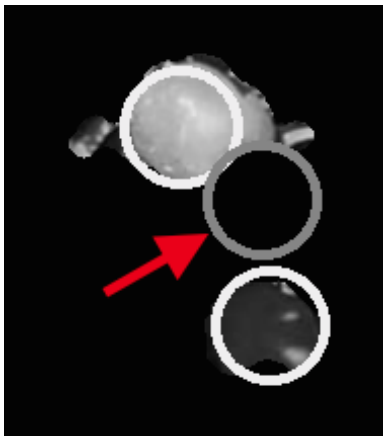


Figura 3.14: *El círculo indicado por la flecha, es detectado a partir de la curvatura hacia afuera que presenta la región superior.*

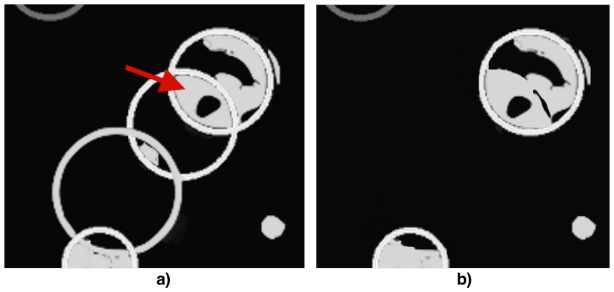


Figura 3.15: *a) Falsas detecciones, la flecha muestra la región de intersección entre dos círculos. Se debe descartar a uno de ellos, ya que la región corresponde a una única manzana, b) resultado de aplicar la técnica quitando la intersección y área mínima.*

3.5. Proceso realizado

A continuación se indica el tiempo dedicado a cada etapa llevada acabo durante el proyecto:

- Estudio de Tratamiento de Imágenes: 6 meses.
- Estado del arte: 2 meses.
- Etiquetado base de datos: 2 meses.
- Implementación: 8 meses.
- Documentación: 5 meses.
- Estudio de resultados: 2 meses.

En la figura 3.16 se puede ver una gráfica mostrando la relación de tiempo en meses entre las distintas etapas.

Figura 3.16: Tiempo de dedicación en meses a las distintas etapas del proyecto.

Capítulo 4

Implementación

En este capítulo se describe la implementación de las técnicas descritas en la solución propuesta.

4.1. Reconocimiento de píxeles manzana

A continuación se presentan los detalles de implementación para la extracción de características y para las técnicas utilizadas para la extracción de píxeles manzanas presentadas en el capítulo anterior.

4.1.1. Extracción de características

OpenCV carga las imágenes en BGR, por lo que el primer paso es transformarlas al espacio de colores HSV utilizando la siguiente función:

```
cvtColor( imagen_bgr, imagen_hsv, CV_BGR2HSV)
```

Donde:

- imagen_bgr: Imagen de entrada en el espacio de colores BGR
- imagen_hsv:Imagen de salida en el espacio de colores HSV
- CV_BGR2HSV: Código de conversión del espacio de color, en este caso convierte de BGR a HSV

Luego, teniendo la imagen en HSV, se pueden obtener la característica Matiz y Saturación pidiendo el canal correspondiente:

```
vector<Mat> canales;
split(imagen_hsv, canales);
Mat hue_mat = canales[0];
Mat sat_mat = canales[1]
```

Donde:

- canales: vector de tres matrices, cada matriz representa un canal.
- split: recibe la imagen en el espacio de colores HSV y retorna en canales la matriz de H, de S y de V.
- hue_mat: Matriz correspondiente a H.
- sat_mat: Matriz correspondiente a S.

Para obtener la densidad de bordes se siguió el diagrama de flujo que se ve en la figura 4.1. Se utilizaron las siguientes funciones que brinda OpenCV:

- Se convierte la imagen a tonos de gris:

  ```
  cvtColor( img, img_gray, CV_BGR2GRAY );
  ```

  Donde:

  - img: imagen de entrada
  - img_gray: imagen de salida en tonos de grises
  - CV_BGR2GRAY: código de conversión del espacio de color, en este caso convierte de BGR a escala de grises.

- Se calculan las derivadas según X e Y:

  ```
  Sobel(img_gray, grad_x, ddepth, x_order, y_order,
  ksize=3, scale=1, delta=0, borderType=BORDER_DEFAULT );
  Sobel( img_gray, grad_y, ddepth,x_order,y_order,
  ksize=3, scale=1, delta=0, borderType=BORDER_DEFAULT );
  ```

  Donde la función toma los siguientes argumentos:

  - img_gray: Imagen entrada de tipo 8-bits de enteros sin signo (0..255)
  - grad_x/grad_y: Imagen salida
  - ddepth: profundidad de la imagen, del tipo CV_16S, 16-bit de enteros con signo (-32768..32767), para evitar overflow
  - x_order: El orden de la derivada en la dirección x. Para calcular el gradiente en x x_order = 1 y y_order = 0, análogo para gradiente según y,
  - y_order: El orden de la derivada en la dirección y.
  - ksize: Tamaño el kernel de sobel: Debe ser 1, 3, 5 o 7.
  - scale: Es un valor opcional para el factor de escala de los valores de la derivada. Por defecto no se aplica escala
  - delta: Valor opcional que es sumado al resultado antes de que se almacenen los valores en la imagen destino
  - borderType: método para la extrapolación de los pixeles. BORDER_DEFAULT

- Se convierten los resultados nuevamente a CV_8U, 8-bits enteros sin signo:

```
convertScaleAbs( grad_y, abs_grad_y );
convertScaleAbs( grad_x, abs_grad_x );
```

Donde:

- grad_y/grad_x : derivadas con respecto a x y a y del paso anterior.
- abs_grad_y/abs_grad_x:derivadas con respecto a x e y del paso anterior, llevadas a 8-bits de enteros sin signo.

- Finalmente, se trata de aproximar el gradiente sumando los dos gradientes direccionales, si bien no es un resultado óptimo es un buen resultado para estos propósitos.

```
addWeighted( abs_grad_x, 0.5, abs_grad_y, 0.5, 0, grad );
```

Donde:

- abs_grad_y/abs_grad_x: derivadas del paso anterior
- grad: imagen resultado de unir los dos gradientes.

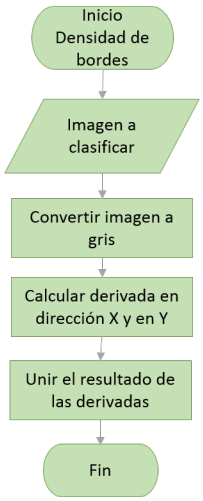


Figura 4.1: Diagrama de flujo de obtención densidad de bordes

4.1.2. Árbol de decisión

Para la implementación del árbol de decisión se sigue el diagrama de la figura 4.2. El algoritmo implementado es el siguiente:

- Cargar imagen
- Filtrar según Hue y Saturation:

```
hsv.getMask(hue_min, hue_max, sat, img, mask_hsv);
```

Donde:

- hue_min/hue_max: rangos de Hue

- sat: rango de Saturation
- img: Imagen de entrada a ser filtrada
- mask_hsv: Imagen de retorno, donde si el valor de un píxel es 1 significa que el valor de ese píxel en img tiene su componente H en el intervalo [0:hue_min] o [hum_max:1] y S en [sat:1]

- Filtrar según Densidad de Bordes:

```
img_edges = edges.getSobelEdges(img);
inRange(img_edges,0,threshold,mask_edg);
```

  Donde:

  - img: Imagen de entrada a calcular la densidad de bordes.
  - img_edges: Imagen resultado de calcular la densidad de bordes como se mostró en la sección anterior.
  - threshold: valor límite considerado como densidad de borde de una manzana.
  - mask_edges:Imagen resultado de filtrar por densidad de bordes.

- Unir mascaras:

```
cv::bitwise_and(mask_hsv, mask_edg,union_mask);
```

  Donde:

  - mask_hsv : mascara obtenida anteriormente al filtrar según H y S
  - mask_edg: mascara obtenida en el paso anterior resultado de filtrar según densidad de bordes.
  - union_mask: Mascara resultado de unir mask_hsv con mask_edg utilizando un and lógico pixel a pixel.

Tanto la función *edges::getSobelEdges como hsv::getMask* son las funciones recién explicadas en obtención de características.

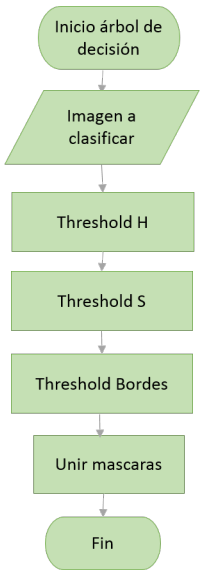


Figura 4.2: Diagrama de flujo del Árbol de decisión

4.1.3. Knn

Para implementar las estructuras necesarias para ejecutar Knn, se utilizaron arreglos y matrices nativas de C++ y OpenCV. Una vez definida la estructura de datos, se debe proseguir a entrenar Knn. OpenCV ofrece métodos nativos que lo implementan, pero como ya se mencionó anteriormente, debido a los largos tiempos de ejecución que estos presentan para la clasificación del píxel entrante, se optó por utilizar FLANN.

- Primero se especifica que el tipo de estructura de datos de índices será en forma de Árbol KD.

  `KDTreeIndexParams`

- Luego se completa la estructura de datos elegida (Árbol KD) para almacenar y ordenar los índices (estructura_ind) de los puntos, a partir de los datos de entrenamiento (matriz_características).

  `Index(matriz_características, estructura_ind):`

- Por ultimo se buscan los K vecinos más próximos (K) del punto entrante a evaluar (punto_entrante), y devuelve los índices de los mismos (índices), tal que el índice *i* corresponde al punto que se encuentra en la i-ésima fila de Matriz características. También devuelve la distancia (dist) a la que se encuentra cada vecino con respecto al punto entrante. Debido a que el Árbol KD puede tener una gran cantidad de niveles jerárquicos, se especifica una condición de parada (prof) para la búsqueda en profundidad de niveles del árbol.

  `KnnSearch(punto_entrante, índices, dist, K, prof):`

4.1.4. SVM

Para utilizar SVM lo primero que se debe realizar es el entrenamiento, para ello se deben disponer los datos en una única matriz, en este caso de tres columnas, H, S y Densidad de bordes y cantidad de filas tantas como píxeles se quieran utilizar para el entrenamiento. También se debe contar con un vector que indique a que clase corresponde cada píxel, en este caso hay dos clases, manzana o no manzana. Este procedimiento es el mismo que se realizó para Knn, el cual se ilustra en la figura 3.9.

Como se vio en el en el marco teórico, SVM puede ser utilizado tanto para casos linealmente separables como para casos en que no, donde hay que recurrir a una función Kernel, por ello es que se deben definir algunos parámetros antes de realizar el entrenamiento.

OpenCV utiliza la clase CvSVMParams para almacenar los siguientes parámetros:

- *svm_type:* Tipo de SVM. En este caso se eligió el tipo *CvSVM::C_SVC* que puede ser utilizado para la clasificación de n-clases, $n \geq 2$. Este tipo funciona con separaciones imperfectas de las clases como por ejemplo separaciones no lineales.

- *kernel_type:* Los tipos de kernel soportados son: *linear, poly, rbf, sigmoid, precomputed.* En este caso se utilizó el *CvSVM:RBF,* en general es un kernel razonable como primera opción, mapea muestras no lineales a un espacio de mayor dimensionalidad.

- *term_ crit:* El procedimiento de entrenamiento es implementado resolviendo un problema de optimización cuadrática de forma iterativa. Se especifica el máximo número de iteraciones y un error de tolerancia, permitiéndole al algoritmo finalizar en menos pasos incluso si el hiperplano óptimo no es encontrado. Este parámetro se define en cvTermCriteria. En este caso: cvTermCriteria(CV_TERMCRIT_ITER+ CV_TERMCRIT_EPS, max_iter , epsilon) donde el primer parámetro es la condición de parada. En este caso luego de una cantidad máxima de iteraciones o cuando el acierto es menor que epsilon.

Luego de tener definidos los parámetros recién explicados, se llama a la función de entrenamiento:

```
train(trainingDataMat, labelsMat, Mat(), Mat(), params);
```

Donde trainingDataMat es la matriz de los datos de entrenamiento, labelsMat a que clase corresponden los mismos y params los parámetros que se crearon en CvSVMParams.

Por último, con el entrenamiento realizado, para predecir a que clase pertenece un píxel, se utiliza el método de predecir de OpenCV,el cual recibe los datos que a predecir y retorna la clase a la que pertenecen.

```
Mat pixel_to_classify = (Mat_<float>(1,3) << h,s,b);
float response = SVM.predict(pixel_to_classify);
```

Donde:

- h, s y b: Características matiz, saturación y densidad de bordes del pixel a clasificar.
- response: Resultado del método, es la clase a la que pertenece el píxel.

4.2. Conteo de manzanas

A continuación se presentan los detalles de implementación para el conteo de manzanas presentado en el capítulo anterior.

4.2.1. Operaciones morfológicas

Como se menciono en el la sección 3.4.1 primero se realiza erosión y dilatación y luego dilatación y erosión. Para ello se utilizan las funciones que brinda OpenCV.

- Primero se debe definir el elemento estructurante:

```
Mat element = getStructuringElement( type,
Size( 2*size + 1, 2*size+1 ), Point( size, size ) );
```

Donde:
type es el tipo del elemento estructurante, en este caso una elipse CV::MORPH_ELLIPSE
Size(2*size + 1, 2*size+1)es el tamaño del kernel
Point(size, size) es el punto donde se pone el elemento estructurante

- Luego se debe invocar a la función erosión o dilatación según corresponda:

```
erode(imagen,imagen_resultado,element);
dilate(imagen,imagen_resultado,element);
```

Donde :

- imagen es la imagen de entrada, en este caso la mascara resultado de alguna de las técnicas de reconocimiento de pixeles.
- imagen_resultado es la imagen que retorna la función
- element es el elemento estructurante definido en el punto anterior.

4.2.2. Detección de círculos

Para la detección de círculos se utiliza Hough para círculos, OpenCV presenta dicho método de forma nativa [1]:

```
HoughCircles(imagen_entrada, circulos, CV_HOUGH_GRADIENT,
dp, dist_min, canny, acumulador, min_radio, max_radio)
```

Donde:

- imagen_entrada es la imagen que recibe, en este caso será siempre una imagen binaria.
- círculos son los círculos detectados, es el lo que retorna la función.
- CV_HOUGH_GRADIENT es el método de detección, actualmente es el único que acepta.
- dp es la resolución del acumulador, si por ejemplo dp=1, tendrá la misma resolución que la imagen, si dp=2, tendrá la mitad de alto y ancho que la imagen. En este caso se utiliza 1, ya que se desea mantener la misma resolución que la imagen.
- dist_min refiere a la distancia mínima que puede haber entre cada círculo detectado .
- canny refiere al umbral de Canny, esto es necesario debido a que parte del método consiste en aplicar reconocimiento de bordes utilizando Canny.
- acumulador es el umbral del acumulador, cuanto menor sea este número, mayor será la sensibilidad de reconocimiento de círculos, por tanto aumenta la probabilidad de detecciones falsas. Por el contrario si el número es muy grande, puede ser que haya círculos en la imagen que no sean detectados.
- min_radio y max_radio, especifican el tamaño de círculos que se desea detectar.

El radio máximo y mínimo se dejan fijos, ya que se considera una restricción del problema que las manzanas estén dentro de un rango determinado.

En la parte Evaluación experimental, se realizarán variaciones de los siguientes dos parámetros que se detallan a continuación: Distancia mínima entre círculos, crítica en caso de acumulación de manzanas ya que cuando hay dos manzanas muy próximas, se deben detectar dos círculos en vez de uno y el otro parámetro más importante a variar es el umbral del acumulador, ya que éste será el nivel de sensibilidad de la detección de círculos.

4.2.3. Pos-procesamiento de círculos

Estudio de área mínima

Para calcular el área de pixeles dentro de un circulo se utiliza la siguiente función:

```
double getAreaPercent(Mat img, Mat mascara_circulo, int radio)
```

Donde:

- img: máscara entrante
- mascara_circulo: mascara compuesta por el circulo a evaluar
- radio: radio del circulo a evaluar

Intersección de círculos

Para la intersección de círculos, se tomará aquellos círculos devueltos por la operación HoughCircles, y se estudiará a los que se intersecten. Para aquellos intersectados, se estudiará el área que contiene cada círculo y se descartará aquel círculo que contenga menos porcentaje de área.

Capítulo 5

Evaluación experimental

En este capítulo se presenta la evaluación experimental realizada durante el proyecto. En la sección 5.1 se presentan las técnicas de evaluación, en la sección 5.2 se presenta la composición de la base de datos. En la sección 5.3 se presentan los resultados para las diferentes técnicas utilizadas para el reconocimiento de píxeles manzana, en la sección 5.4 se describen los resultados para la detección de manzanas. Por último en la sección 5.5 se muestran ejemplos de imágenes resultado de las técnicas estudiadas.

5.1. Técnicas de validación

En esta sección se describen las técnicas utilizadas para la evaluación de los resultados obtenidos.

5.1.1. ROC

Consideremos un problema de predicción de clases binario, en la que los resultados se etiquetan positivos (p) o negativos (n). Hay cuatro posibles resultados a partir de un clasificador binario como el propuesto. Si el resultado de un experimento es p y el valor dado es también p, entonces se conoce como un Verdadero Positivo (VP); sin embargo si el valor real es n entonces se conoce como un Falso Positivo (FP). De igual modo, tenemos un Verdadero Negativo (VN) cuando tanto el experimento como el valor dado son n, y un Falso Negativo (FN) cuando el resultado de la predicción es n pero el valor real es p.

En este proyecto se quiere determinar si un pixel pertenece a una manzana (p) o no pertenece a una manzana (n). Un falso positivo en este caso ocurre cuando se predice que el resultado es manzana cuando en realidad no lo es. Un falso negativo, por el contrario, ocurre cuando el resultado de la predicción es negativo, sugiriendo que no es manzana pero en realidad si lo es.

Un verdadero negativo es cuando se predice que no es manzana y la predicción es correcta, y del mismo modo, un verdadero positivo es cuando se predice que el píxel pertenece a una manzana y de verdad pertenece.

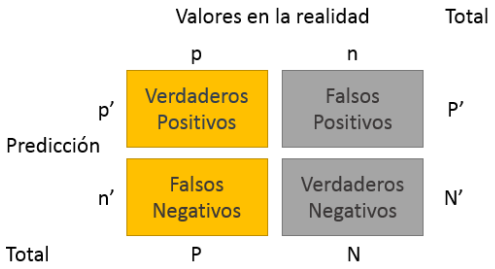


Figura 5.1: Matriz de confusión

La matriz de confusión (ver figura 5.1) puede proporcionar varias medidas de evaluación. Para dibujar una curva ROC sólo son necesarias las razones de Verdaderos Positivos (VPR) y de falsos positivos (FPR). La VPR mide hasta qué punto un clasificador es capaz de clasificar los casos positivos correctamente, de entre todos los casos positivos disponibles durante la prueba. La FPR define cuántos resultados positivos son incorrectos de entre todos los casos negativos disponibles durante la prueba. Lo recién mencionado se puede observar en las siguientes ecuaciones:

$$Sensibilidad = VPR = \frac{VP}{VP+FN}$$

$$1 - especificidad = FPR = 1 - \frac{VN}{VN+FP}$$

Un espacio ROC se define por FPR y VPR como ejes x e y respectivamente, y representa los intercambios entre verdaderos positivos (en principio, beneficios) y falsos positivos (en principio, costos).

El mejor método posible de predicción se situaría en un punto en la esquina superior izquierda, o coordenada (0,1) del espacio ROC, representando un 100 % de sensibilidad (ningún falso negativo) y un 100 % también de especificidad (ningún falso positivo). A este punto (0,1) también se le llama una clasificación perfecta. Por el contrario, una clasificación totalmente aleatoria daría un punto a lo largo de la línea diagonal, que se llama también línea de no-discriminación, desde el extremo inferior izquierdo hasta la esquina superior derecha (independientemente de los tipos de base positiva y negativa).

La diagonal divide el espacio ROC. Los puntos por encima de la diagonal representan los buenos resultados de clasificación (mejor que el azar), puntos por debajo de la línea de los resultados pobres (peor que al azar).

Otro valor interesante a tener en cuenta para el análisis de los datos es la exactitud, conocida en inglés como *accuracy*. Refiere a la exactitud de los resultados obtenidos, donde se mide los valores acertados frente a la cantidad total de muestras:

$$accuracy = ACC = \frac{\#VP+\#VN}{TotalPoblación}$$

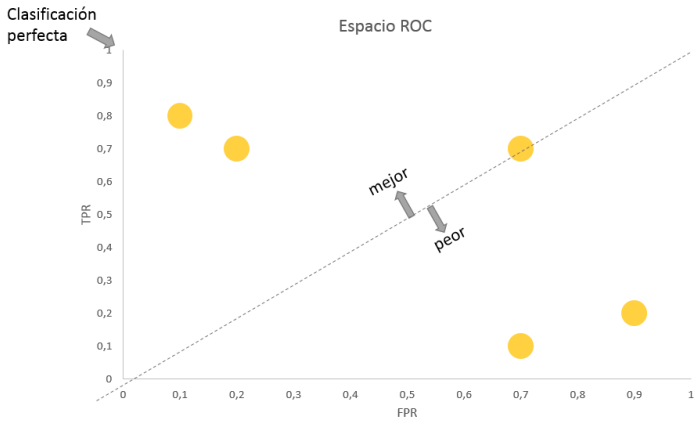


Figura 5.2: *Curva ROC*

5.1.2. FV

El valor FV es la medida de precisión de un test, se considera como una media armónica que combina los valores de precisión y de sensibilidad. De tal forma que:

$$FV = 2\frac{Precisión.Sensibilidad}{Precisión+Sensibilidad} = 2\frac{VP}{2VP+FP+FN}$$

$$Precisión = \frac{VP}{VP+FP} \ , \ Sensibilidad = \frac{VP}{VP+FN}$$

Utilizando como ejemplo la clasificación de píxeles en manzana o no manzana, precisión es el número de resultados correctos, es decir píxeles clasificados como manzana que efectivamente son manzana sobre el número total de píxeles clasificados como manzanas. Por otr lado, Sensibilidad es el número de resultados correctos sobre el total de píxeles que pertenecen efectivamente a manzanas.

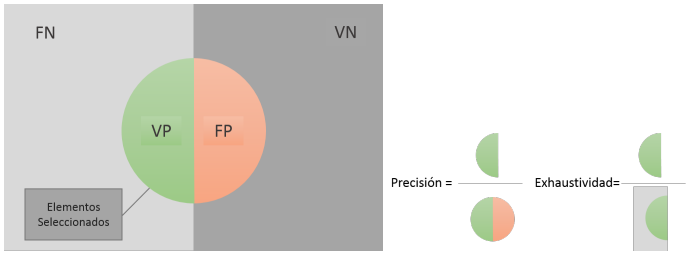


Figura 5.3: *Precisión y sensibilidad.*

Cuando las clases están desbalanceadas, si se utilizará como medida de rendimiento la precisión (Acc), tendríamos resultados incorrectos, dado que en el caso de que la clase

de negativos sea mucho mayor que la de positivos, al reconocer todo como negativo obtendríamos un Acc muy bueno. Por ejemplo, en el caso de reconocer manzanas en un árbol, la cantidad de pixeles que pertenecen al fondo es mucho mayor que los que pertenecen a manzanas. Por lo tanto si clasificamos todo como fondo, la Acc va a ser muy buena, pero si embargo no reconoceríamos nada. Por tal motivo es que en clases desbalanceadas se utilizan otros indicadores, como el FV. En la figura 5.3 se encuentra una ilustración de lo descrito.

5.1.3. Validación cruzada (Cross Validation)

La validación cruzada es una técnica utilizada para evaluar los resultados de un análisis estadístico y garantizar que son independientes de la partición entre datos de entrenamiento y prueba. Consiste en no utilizar todo el conjunto de datos para el entrenamiento, removiendo algunos datos antes de que el mismo comience. Luego, cuando el entrenamiento finaliza, los datos que fueron removidos pueden ser utilizados para medir la performance del modelo aprendido.

Un tipo de validación cruzada es el de K iteraciones o K-fold cross-validation donde los datos de muestra se dividen en K subconjuntos. Uno de los subconjuntos se utiliza como datos de prueba y el resto (K-1) como datos de entrenamiento. El proceso de validación cruzada es repetido durante K iteraciones, con cada uno de los posibles subconjuntos de datos de prueba. Finalmente se utilizan todos los resultados de cada iteración para obtener un único resultado. Este método es muy preciso puesto que evaluamos a partir de K combinaciones de datos de entrenamiento y de prueba, pero aun así tiene una desventaja, y es que, es lento desde el punto de vista computacional. En la práctica, la elección del número de iteraciones depende de la medida del conjunto de datos. Lo más común es utilizar validación cruzada de 10 iteraciones (10-fold cross-validation). Lo descrito se puede visualizar en la figura 5.4.

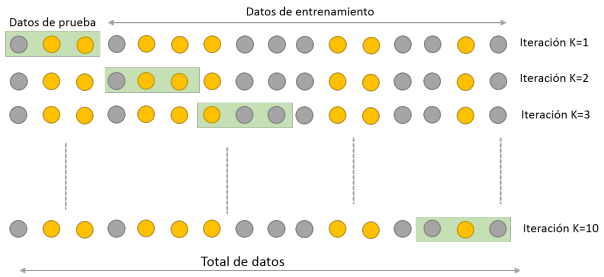


Figura 5.4: *Validación cruzada de K iteraciones con K=10.*

5.2. Base de datos

En proyectos de esta índole, la base de datos es unos de los pilares fundamentales sobre el cuál se basa el desarrollo de todo el proyecto. Es a partir de la base de datos que se imponen las condiciones para el estudio de efectividad y eficacia de las técnicas estudiadas. En este caso la base de datos se ha construido a partir de imágenes de manzanos con manzanas rojas adquiridas en huertas, no en laboratorios ni en ambientes controlados.

Se ve como un beneficio que la adquisición de las imágenes se haya realizado en ambientes no controlados, ya que la efectividad de las técnicas estudiadas no queda

condicionada a que las fotografías sean adquiridas en laboratorios. Sí es un requerimiento que los elementos presentes tengan un tono (Hue) distinto al que presentan las manzanas rojas.

Además de las imágenes de manzanos, la base de datos cuenta también con una máscara, imagen binaria, por cada imagen, indicando los píxeles manzana y no manzana. Por último también se cuenta con un tercer archivo que indica la ubicación de las manzanas. Se cuenta entonces con un total de tres archivos por fotografía: imagen, máscara binaria y ubicación de las manzanas.

Por otro lado, además de la base de datos en un sistema de archivos, para el estudio de la efectividad de las técnicas de reconocimiento de píxeles se implementó una estructura de memoria a partir de la del sistema de archivos. A continuación se detallará en qué consisten ambos tipos.

5.2.1. Sistema de archivos

Imágenes

Por las características del proyecto las imágenes pueden ser tomadas únicamente en un momento específico, cuando las manzanas están en su árbol.

Para este proyecto las imágenes fueron tomadas en una visita al campo del INIA Las Brujas al mismo momento que estaba comenzando el proyecto, por lo que las mismas no fueron tomadas con este fin. Cuando se estudiaron diferentes técnicas y se decidió cómo iba a ser la implementación, ya no se contaba con manzanas en los árboles, teniéndose entonces que adaptar a estas imágenes.

Se cuenta entonces con un total de 266 imágenes, de las cuales 20 fueron adquiridas en la visita recién mencionada y otras extraídas de diversas fuentes de sitios en Internet. Esto es un desafío debido a que al provenir de distintas fuentes, hay variabilidades en las cámaras utilizadas para la adquisición, condiciones de luz y especies de manzanas.

En la figura 5.5 se visualizan algunas de las imágenes que componen la base de datos, donde se puede apreciar entre otras cosas diferencias de iluminación, tiempo de exposición, distintos balances de blancos, distintas especies de manzana y adquisición realizada a distintas distancias.

Los tamaños de las imágenes varían también en resolución y número de bytes. En cuanto a resolución, el valor promedio es de aproximadamente 600x400 píxeles, y el tamaño de bytes varían entre 200Kb y 1.7Mb.

Imagen binaria

Además de las imágenes especificadas en la sección anterior, se deben indicar por cada una, los píxeles manzanas que se encuentran allí. Esto es necesario para el entrenamiento de los métodos de clasificación de píxeles.

Como se detalló en el capítulo Marco teórico, los píxeles se clasifican en 2 clases: manzana y no manzana, la forma de indicar esto se resolvió a través de una imagen binaria, donde cada píxel de la misma indica la clasificación respecto al píxel de la

imagen original: negro para clase no manzana, blanco para clase manzana. En la figura 5.6 se puede apreciar lo descripto.

Figura 5.5: Muestra de imágenes que se encuentran presentes en la base de datos. *Se observa la variabilidad entre: especies de manzanas, balance de blancos, iluminación y tiempo de exposición.*

Para llevar esto a cabo, se implementó un programa que se fue iterando para cada imagen de la base de datos, donde se genera una imagen binaria en negro con las mismas dimensiones que la imagen entrante, y se va formando las regiones (en color blanco) arrastrando el cursor sobre las manzanas en la imagen de entrada, similar como cuando se utiliza la herramienta pincel en un editor gráfico.

Figura 5.6: Imagen binaria. *Imagen de manzanas con su respectiva imagen binaria. Los píxeles blancos corresponden a píxeles clasificados como manzana, los negros como no manzanas.*

Ubicación de manzanas

Además de la imagen binaria que se utiliza para el entrenamiento y evaluación de las técnicas de reconocimiento de píxeles manzana, es necesario también conocer la ubicación de las manzanas presentes en las imágenes y así evaluar las técnicas de cuantificación de manzanas. Para realizar esto, de forma manual se reconocen las manzanas presentes y se almacenan las coordenadas del centro de cada una. Se toma como punto de origen el vértice superior izquierdo, y las coordenadas x, y como la cantidad de píxeles a la que se encuentra el centro de forma horizontal y vertical respectivamente.

5.2.2. Estructuras en memoria

Los métodos de reconocimiento de píxeles Knn y SVM , tienen en común que utilizan conjuntos de puntos junto con sus características y clasificación, donde no es relevante en qué parte de la imagen se encuentra cada punto. En base a esto es que tanto para el entrenamiento como para la clasificación, estos dos métodos modelan su dominio como un conjunto de puntos.

El dominio mencionado, se implementa a partir de la base de datos en un sistema de archivos, haciendo lectura de cada píxel de la fotografía y de la imagen binaria. De la fotografía se extraerá los correspondientes valores de las características que los métodos de clasificación utilicen, y de la imagen binaria se extraerá su clasificación.

Estos valores extraídos se almacenan en una matriz de tamaño $N \times 4$, donde N es la cantidad de píxeles leídos a lo largo de toda las imágenes de la base de datos y 4 son las tres características más la etiqueta del píxel. Esta matriz será la base de datos cargada en memoria, se pierde el concepto de base de datos como conjuntos de imágenes y se introduce el concepto de la base de datos como un conjunto de puntos (con sus valores de características y clasificación) en estructuras en memoria. En la figura 5.7 se visualiza lo descripto.

5.2.3. Porcentaje de muestreo de la base de datos

La base de datos se encuentra compuesta por un total de 266 imágenes, con un promedio de resolución de 600x400 px, es un total de 266 * 600 * 400 = 63.840.000 píxeles. Es decir que si se tomara en cuenta el total de píxeles de las distintas fotografías de la base de datos, se necesitaría en memoria una estructura con casi 64 millones de lugares. A esto se lo debe multiplicar por tres debido a que son tres características que se estudian, y además multiplicarlo también por dos ya que se debe tomar en cuenta la clasificación del píxel a partir de la imagen binaria. Si se tomaran todos los píxeles, se necesitarían estructuras de datos muy grandes que implican costes muy altos de procesamiento y tiempo.

Por lo descripto en el párrafo anterior, es de suma importancia realizar un estudio del porcentaje mínimo de píxeles necesarios que serán tomados de tal forma que sea una representación aceptable del total de los píxeles en la base de datos. Para esto se variarán distintos porcentajes y se clasificarán los píxeles (cómo manzana o no manzana) utilizando Árbol de decisión, ya que es la técnica que requiere menos tiempos de ejecución.

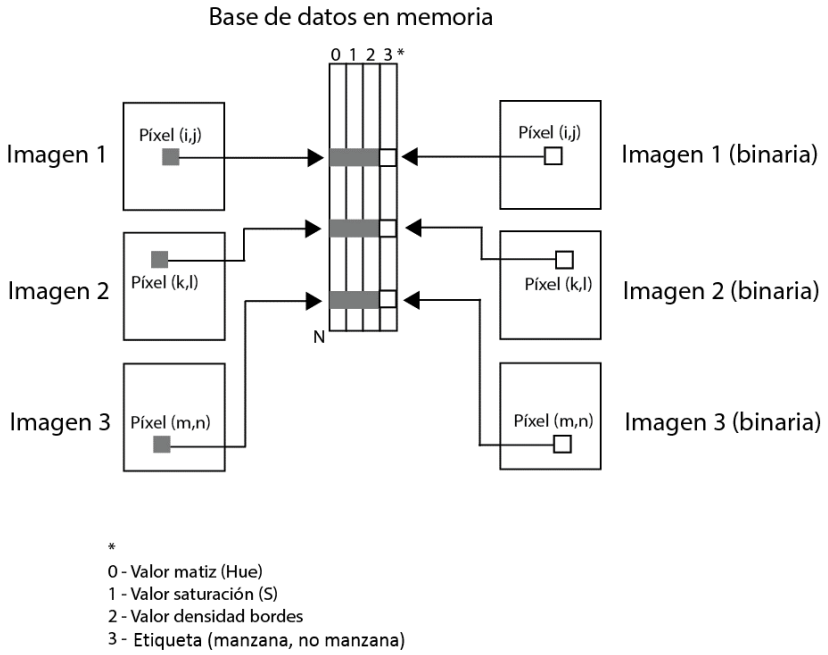


Figura 5.7: Estructura de la base de datos en memoria. *La base de datos en memoria está compuesta a partir de la lectura de píxeles de las fotografías a lo largo de toda la base de datos, junto con su respectiva imagen binaria para obtener la clasificación del píxel. Se tiene como resultado una matriz de* $N \times 4$*, siendo* N *la cantidad de píxeles leídos.*

Se iterará variando el porcentaje de píxeles y clasificando. El bucle se detendrá cuando luego de varias iteraciones se concluya que no hay diferencia significativa entre el resultado de una iteración y las anteriores, esto quiere decir que el muestreo de píxeles que se toma, por más grande que fuera, tiende a un mismo resultado, por tanto con tomar un porcentaje menor donde el resultado no varíe, se considerará que es un buen muestreo de la base de datos en su totalidad.

Para poder abarcar la mayor cantidad de variaciones de las características de píxeles, se propone tomar un porcentaje de píxeles del total por imagen, y en posiciones aleatorias. Es decir, para cada imagen de la base de datos se tomará un porcentaje de píxeles del total, y se tomarán de forma aleatoria en el alto y ancho de la imagen, sin criterio alguno en la elección.

Un ejemplo de lo que sucede al no tomar de forma aleatoria la elección, se puede ver si se considera una imagen en la cual la mitad superior es cielo. Si se seleccionan los píxeles comenzando desde la esquina superior izquierda y con barrido hacia la derecha, el 10 % de los píxeles no presentarían gran variación en el valor de sus características, ya que todos pertenecerían a cielo.

En la figura 5.8 se puede ver como varía el valor de FV al aumentar el porcentaje de la base de datos hasta que queda estacionario a partir del 0,4 %. Se utiliza 0,5 % para dar un margen.

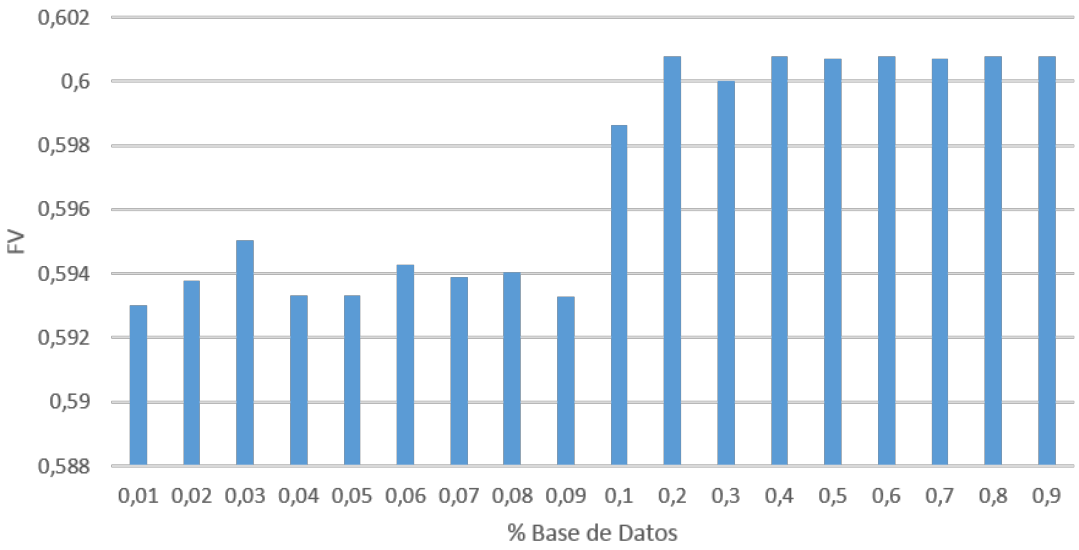


Figura 5.8: *FV de la clasificación de los pixeles utilizando árbol de decisión al variar el porcentaje de la base de datos.*

5.3. Reconocimiento de píxeles

A continuación ser presentan los resultados para las diferentes técnicas de reconocimiento de pixeles, la comparación de las mismas y los resultados de utilizar operaciones morfológicas.

5.3.1. Árbol de decisión

A continuación se describen los resultados obtenidos estudiando la efectividad de la técnica Árbol de decisión, variando los parámetros que la misma involucra.

Para el estudio de la técnica se utilizará un porcentaje de puntos extraídos de la base de datos, que cómo ya se estudió, es del 0.05 %. Los puntos contenidos en este subconjunto, se reordenarán de forma aleatoria, de tal forma que en cada reordenamiento las particiones de Cross validation que se formarán para evaluación y entrenamiento serán distintas. A cada reordenamiento, se le llamará *muestreo*, realizando un total de diez muestreos, donde en cada uno se utiliza Cross Validation de diez iteraciones.

Ajuste de Matiz (H), Saturación (S) y Densidad de Bordes

Para realizar el estudio de los valores óptimos de los tres parámetros, se realiza una búsqueda exhaustiva de los mismos, descartando de la búsqueda los valores que se puede prever que no serán óptimos. Estos valores que se descartan para la búsqueda, son por ejemplo los valores que representan los tonos azules o amarillos, en el estudio de valores óptimos de la matiz.

Como primer paso se determinó el rango aproximado para los valores de Matiz (H). Sabiendo que el tono de las manzanas es rojizo, se utilizan para la evaluación valores cercanos a los mismos.

En la figura 5.9 se observa el histograma del matiz de las manzanas de la base de datos, así como los del fondo.

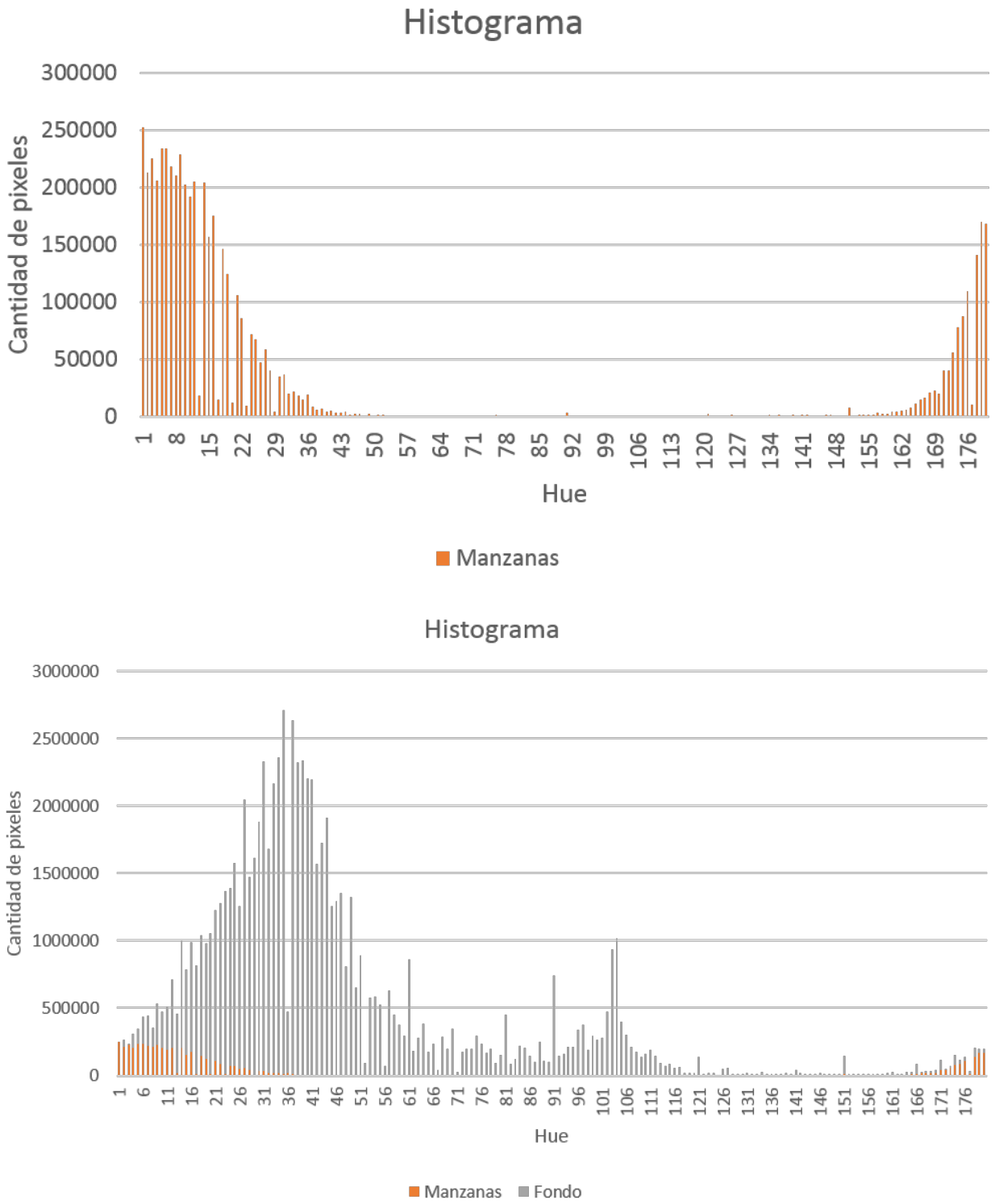


Figura 5.9: Histograma Hue. *La imagen de arriba muestra el histograma para el matiz de las manzanas, abajo se observa el histograma de las manzanas y el fondo*

Por definición de saturación, los valores que se quieren descartar son los más bajos.

Para densidad de bordes, al contrario que saturación, se quieren descartar los valores más altos, debido a que los mismos se asocian al pasto u hojas.

Sabiendo entonces en qué rangos aproximados varían los tres parámetros, se realiza una búsqueda exhaustiva sobre los mismos, utilizando como ya se mencionó Cross

Validation en diez iteraciones, repetido en diez muestreos.

Los resultados de FV para los valores óptimos son en promedio $0,62 \pm 6,75e-5$.

En la tabla 5.2 se observan el promedio y la desviación estándar de cada características.

H rango menor	H rango mayor	S	Densidad de Bordes	FV
13 ± 0	$161,2 \pm 1,47$	47 ± 0	76	$0,62 \pm 6,75E-05$

Cuadro 5.1: *Valores óptimos de H, S y Densidad de Bordes.*

Como conclusión, en la figura 5.10 se observa como mejoran los resultados al ir agregando características.

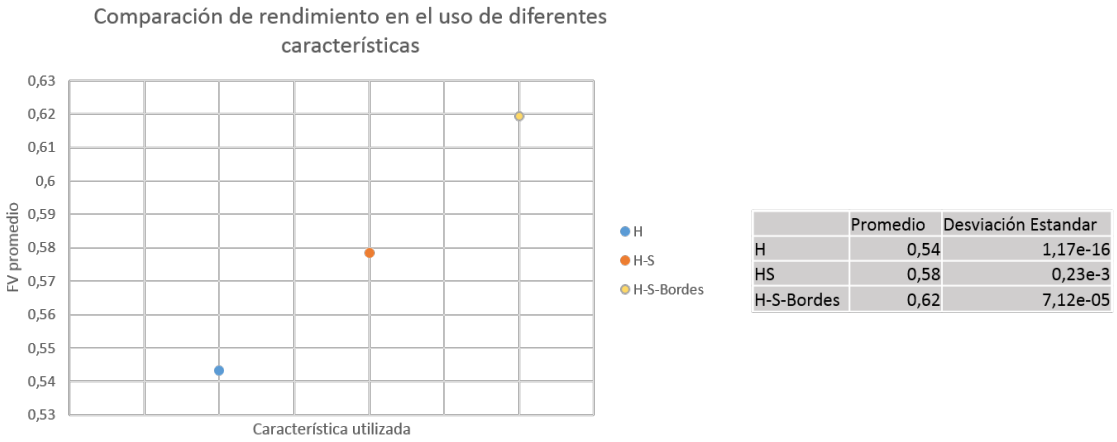


Figura 5.10: *Resultados al utilizar distintas características.*

5.3.2. Knn

Como se detalló en el capítulo Solución Propuesta, el estudio de efectividad y eficacia de la técnica de Knn se realizará variando:

1. Porcentaje de píxeles clasificados como no manzanas.

   En una fotografía común de la base de datos, el área total que abarca las superficies de las manzanas en la fotografía, es una proporción muy pequeña en comparación con el área total de la imagen. Esto conlleva a que por cada imagen siempre se tendrá una cantidad mucho menor de píxeles clasificados como manzanas, en comparación con los clasificados como no manzanas (llamados también píxeles de fondo). Esto provoca que en el dominio de las características la nube de píxeles de fondo tenga mucha mayor densidad que la nube de píxeles manzanas, es decir, en una misma proporción de espacio del dominio, habrá mucho más píxeles de fondo que manzanas. Esto genera un problema, ya que puede haber una mala clasificación para aquellos píxeles entrantes que se encuentren próximo a las frontera de ambas nubes, ya que se tomarán siempre mayor cantidad de píxeles de fondo que de manzana, como se visualiza en la figura 5.11.

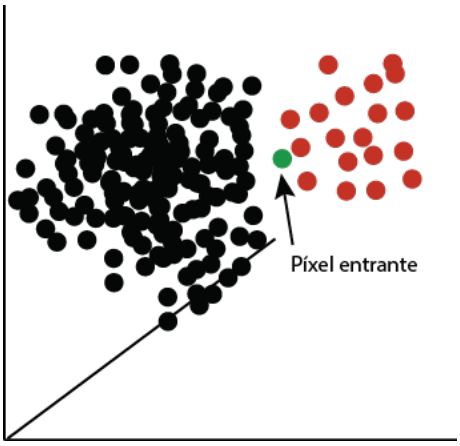


Figura 5.11: *Píxeles distribuidos en el dominio de características para Knn. La nube izquierda está conformada por píxeles clasificados como no manzanas (de fondo), y la nube a la derecha con píxeles clasificados como manzana. Esta última nube tiene una menor densidad, provocando una cantidad menor de vecinos más cercanos al píxel entrante en comparación con los vecinos más cercanos de la nube izquierda, aunque el mismo se encuentra más próximo a la nube de la derecha.*

Una posible solución a lo detallado anteriormente, es disminuir la densidad de la nube de píxeles de fondo en el entrenamiento, lo que se conoce como undersampling [9, 17], dejándola lo más próxima a la densidad de la nube de píxeles manzanas. Se evaluará entonces variaciones en el porcentaje de píxeles de fondo a retener para el entrenamiento de Knn.

2. Cantidad de vecinos más próximos a considerar para la clasificación (valor de K).

   Como se ha detallado en el Marco teórico, la selección del valor de K es uno de los puntos más críticos. Es el principal parámetro del cuál dependerá la efectividad de la técnica. Tener en cuenta que cuanto mayor es K, mayores serán los costes de tiempo y procesamiento, por tanto se seleccionará el menor K posible para el cuál se haya tenido los mejores resultados.

Las pruebas se realizarán utilizando Cross validation de 10 particiones, donde a su vez, se utilizo Cross Validation de 10 particiones sobre el conjunto de entrenamiento. El único parámetro a variar de la técnica Knn, es el valor de K, aún así, como ya se mencionó, se debe evaluar que se entrene con distintos porcentajes de píxeles clasificados como no manzana.

Se realizarán las pruebas con los siguientes posibles valores:

- **% píxeles de fondo**: 1 %, 5 %, 10 %, 15 %, 20 %, 25 %, 30 %, 35 %, 40 %, 45 %, 50 %, 55 %, 60 %, 65 %, 70 %, 75 %.
- **K**: 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33.

Se entrenará y evaluará tomando una muestra de porcentaje mínimo de una buena representación de la base en su totalidad, que cómo ya se estudió en la sección 4.1,

el porcentaje mínimo aceptable es del 0.05 %. Debido al bajo porcentaje, resulta ser una cantidad de píxeles no muy elevada con el que se entrena y evalúa, por tanto los tiempos de ejecución no son tan altos, y por tal motivo se optó por no utilizar Hill climbing en estas pruebas. Se reordena de forma aleatoria la base de datos y se repiten los experimentos 10 veces, por tanto se realizará un total de 10 muestreos. En la figura 5.12, se visualiza los resultados.

Muestreo	K	% píxeles de fondo	TP	FP	TN	FN	Recall	Precision	Fv
1	15-27	45-60%	1589	812	38849	910	0.64	0.66	0.65
2	19-31	45%-55%	1607	803	38858	892	0.64	0.67	0.65
3	19-31	40%-55%	1558	777	38884	941	0.62	0.67	0.64
4	19-31	35%-55%	1565	800	38861	934	0.63	0.66	0.64
5	17-31	40%-50%	1605	829	38832	894	0.64	0.66	0.65
6	17-33	35%-55%	1610	869	38792	889	0.64	0.65	0.65
7	13-29	40%-50%	1594	810	38851	905	0.64	0.66	0.65
8	17-27	40%-50%	1577	805	38856	922	0.63	0.66	0.65
9	17-31	40%-50%	1569	790	38871	930	0.63	0.67	0.65
10	17-33	40%-50%	1590	812	38849	909	0.64	0.66	0.65

Figura 5.12: *Valores de FV para Knn durante los 10 muestreos.*

Se observa que los resultados se encuentran aproximadamente entre 0.64 y 0.65, con una media de $0,65 \pm 0{,}003$. En la figura 5.13 se observan los valores de parámetros óptimos que se obtuvieron en cada muestreo. Se puede observar también la cantidad de píxeles que se evaluó, con un total de más de 42.000 píxeles.

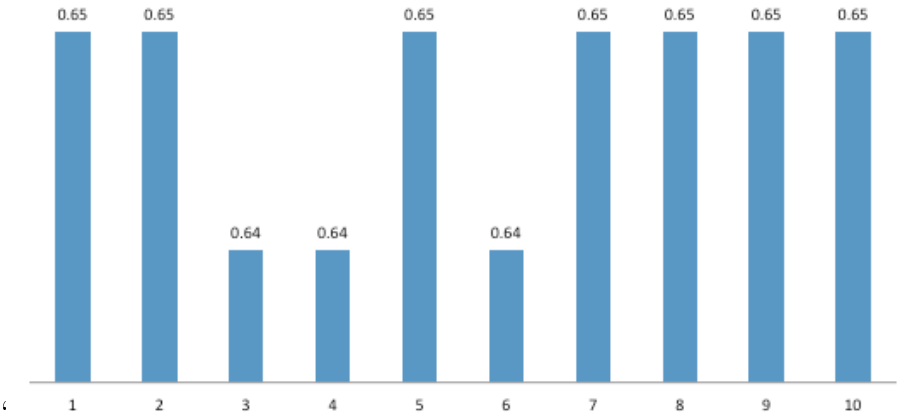


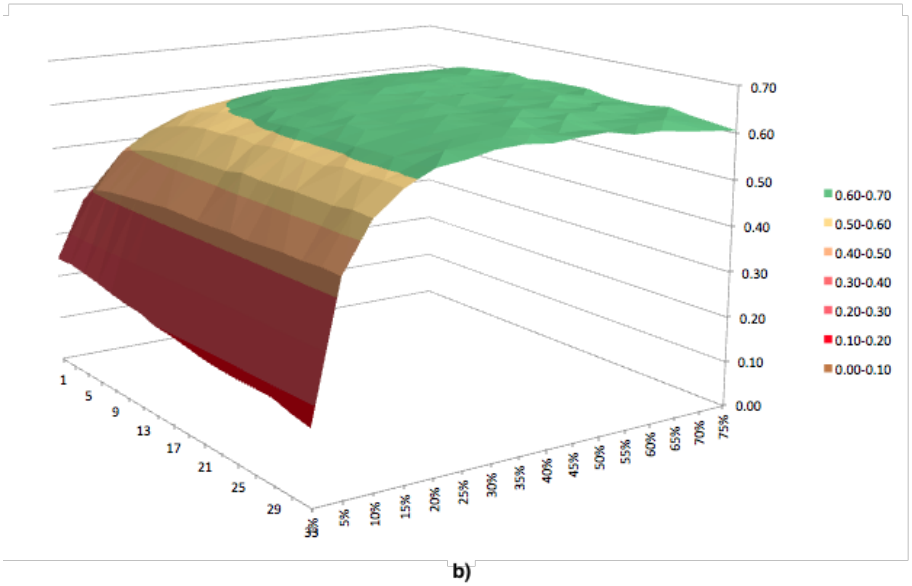
Figura 5.13: *Gráfica de valores de FV para Knn en los 10 muestreos*

En cuanto a los valores de parámetros óptimos, en la figura 5.13 se puede observar que el rango predominante de K es entre 19-31, y 35 %-55 % para porcentaje píxeles de fondo.

Es posible modelar la efectividad del método frente a la variabilidad de los parámetros, como se muestra en la figura 5.14. Allí se visualiza en una iteración de entrenamiento para el muestreo 5 de Cross validation, como varía el FV según los valores de los parámetros. El comportamiento es similar en los distintos muestreos.

	1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33
1%	0.24	0.24	0.24	0.23	0.22	0.22	0.21	0.20	0.20	0.19	0.18	0.18	0.18	0.18	0.18	0.17	0.16
5%	0.37	0.41	0.42	0.42	0.42	0.43	0.43	0.43	0.43	0.43	0.44	0.43	0.44	0.44	0.44	0.44	0.44
10%	0.43	0.49	0.50	0.51	0.52	0.52	0.53	0.53	0.53	0.54	0.54	0.54	0.54	0.54	0.54	0.54	0.54
15%	0.46	0.53	0.55	0.56	0.57	0.58	0.58	0.58	0.58	0.58	0.58	0.58	0.58	0.59	0.59	0.59	0.59
20%	0.48	0.55	0.58	0.59	0.60	0.60	0.60	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.62
25%	0.49	0.57	0.59	0.60	0.61	0.62	0.62	0.62	0.62	0.62	0.63	0.63	0.62	0.62	0.63	0.63	0.62
30%	0.50	0.58	0.60	0.62	0.62	0.62	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.64	0.64
35%	0.51	0.58	0.61	0.62	0.62	0.63	0.63	0.63	0.64	0.63	0.64	0.64	0.64	0.64	0.64	0.64	0.64
40%	0.51	0.59	0.61	0.62	0.63	0.63	0.63	0.64	0.64	0.64	0.64	0.64	0.64	0.64	0.64	0.64	0.63
45%	0.52	0.59	0.61	0.62	0.63	0.64	0.64	0.63	0.64	0.64	0.63	0.64	0.64	0.64	0.63	0.63	0.64
50%	0.52	0.59	0.61	0.63	0.63	0.64	0.64	0.64	0.63	0.64	0.64	0.64	0.65	0.64	0.64	0.63	0.64
55%	0.52	0.60	0.61	0.63	0.63	0.64	0.64	0.64	0.63	0.63	0.64	0.63	0.64	0.64	0.63	0.63	0.63
60%	0.53	0.59	0.62	0.62	0.63	0.63	0.63	0.63	0.64	0.63	0.64	0.64	0.63	0.64	0.63	0.63	0.63
65%	0.53	0.59	0.61	0.62	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.62	0.63	0.62	0.63	0.63	0.62
70%	0.53	0.59	0.61	0.63	0.63	0.62	0.63	0.63	0.62	0.62	0.63	0.62	0.63	0.62	0.62	0.62	0.61
75%	0.53	0.59	0.61	0.62	0.62	0.63	0.62	0.63	0.63	0.63	0.62	0.62	0.62	0.62	0.62	0.61	0.61

a)



b)

Figura 5.14: *a) Resultado de FV frente a la variabilidad de K y porcentajes de píxeles de fondo en único muestreo, b) Representación gráfica en 3D de la tabla a).*

Visualmente se observa que el método es más efectivo en la zona de color verde, determinado por el rango de valores 13-31 para K, y el rango 35 %-55 % para porcentaje de píxeles de fondo, rango bastante similar al obtenido en todas los muestreos.

5.3.3. Support Vector Machine

Cómo se detalló en Solución propuesta, el estudio de efectividad y eficacia de la técnica de SVM se realizará variando Gamma y C

Las pruebas se realizarán utilizando Cross validation de diez particiones, donde a su vez, cada partición se entrena utilizando Cross validation de diez particiones.

Se realizarán las pruebas con los siguientes posibles valores:

- **Gamma**:1,3,7 y 10
- **C**: 0.1,1,10,70 y 100,150,200

Se entrenará y evaluará tomando una muestra de la base de datos con el porcentaje mencionado en la sección de resultados base de datos, es decir 0.05 % de la base de datos, valor cuya representación de la base de datos retorna valores estables.

Se observa que el FV promedio es $0,642 \pm 0,8e-3$.

Para nueve de las 10 iteraciones, los valores que retornaron mejores resultados fueron C=100 y gamma=1.

5.3.4. Comparación de técnicas

En la figura 5.15 se visualiza una comparación entre las tres técnicas de reconocimiento de píxeles, con sus respectivos valores de FV.

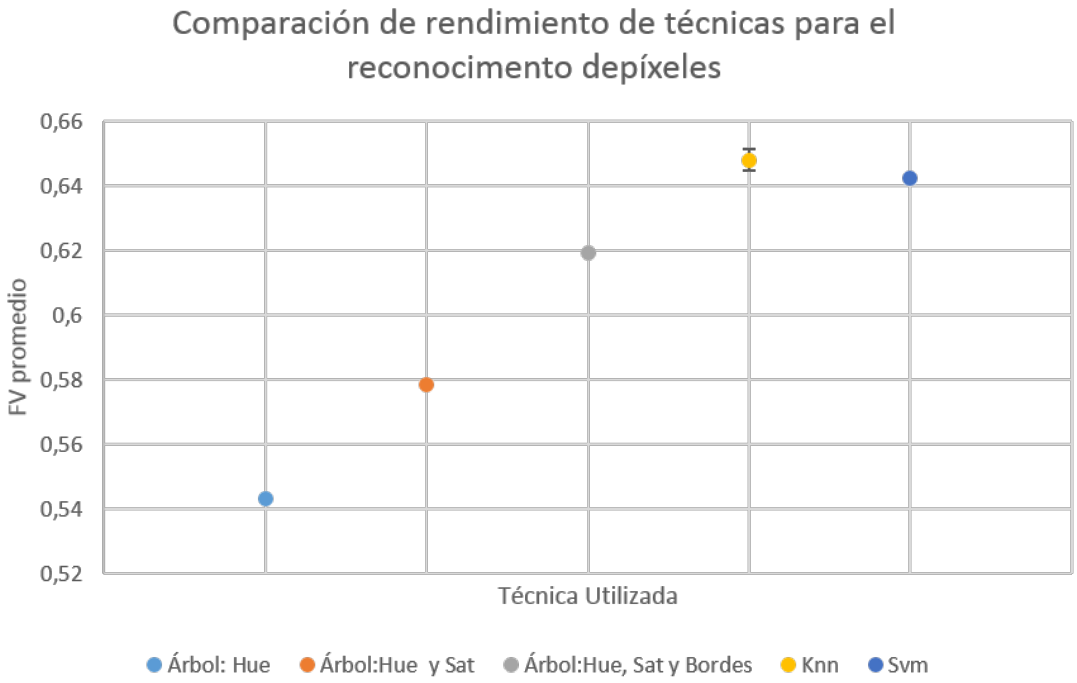


Figura 5.15: *Comparación entre las técnicas de reconocimiento de píxeles.*

Debido a que las 3 técnicas utilizaron el mismo dominio de características para su clasificación, resulta interesante una comparación de la clasificación de cada uno de los métodos para un conjunto de puntos finitos normalizados de 0 a 1 (un cubo) de las 3 características, se puede visualizar en 3D, como se muestra en la figura 5.16

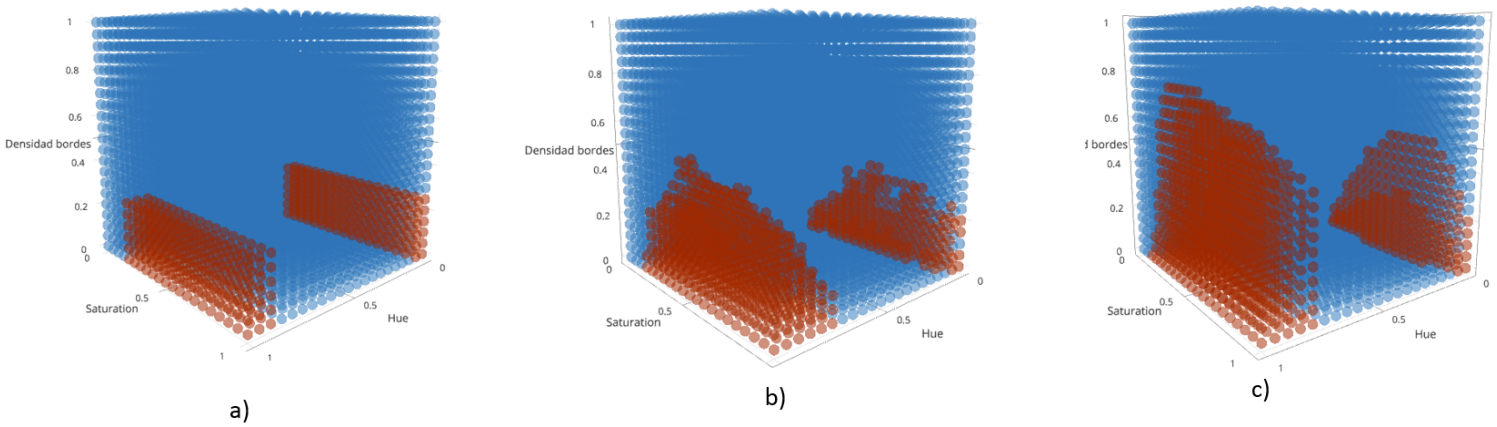


Figura 5.16: *a) Espacio de puntos clasificados por el método Árbol de decisión, b) Espacio de puntos clasificados por Knn, c) Espacio de puntos clasificados por SVM. Los puntos rojos son los clasificados como manzanas, los azules como fondo (no manzana).*

Se observa cómo si bien la nube de píxeles manzanas en los tres métodos se aproxima, Knn y SVM tienen una mejor aproximación, evitando fronteras rectas como sí sucede con Árbol de decisión.

5.3.5. Operaciones morfológicas

Los objetivos de aplicar operaciones morfológicas en las imágenes binarias se tratan fundamentalmente para la disminución de ruido y mejora del alisado de las regiones consideradas como manzanas, así como también la reducción de huecos dentro de las mismas. En el marco de la etapa de procesamiento de este proyecto, las operaciones morfológicas se realizan luego del reconocimiento de píxeles y antes de comenzar con la detección de manzanas, y se consideran cómo una mejora dentro del contexto de reconocimiento de píxeles.

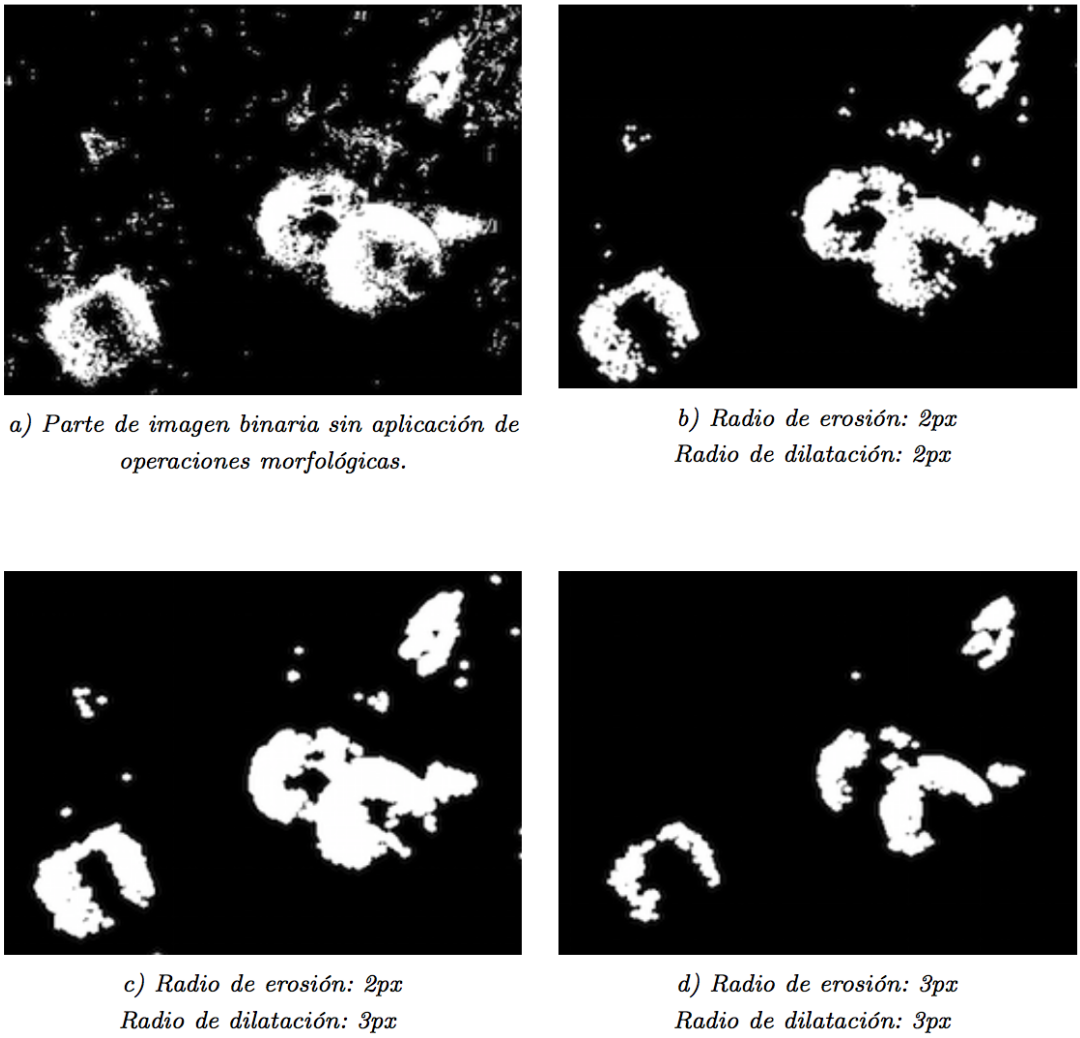
Se propone aplicar primero erosión y dilatación, y luego dilatación y erosión. En la primer caso se busca la eliminación de ruido sin la pérdida de área de las regiones que interesan, y en el segundo caso se busca el alisado de las regiones y cierre de huecos sin perder la forma externa de la región.

Erosión y dilatación como eliminación de ruido

Las posibles variaciones para la eliminación de ruido de las imágenes binarias (resultado de ejecutar algún método de reconocimiento de píxeles), es variando la elección del tipo de elemento estructurante y el tamaño del mismo. Debido a que las manzanas presentan una forma geométrica bastante circular, se opta por utilizar únicamente como tipo de elemento estructurante un círculo, ya que de esta forma la dilatación y erosión en los bordes externos de las regiones se mantendrán con curvatura.

A modo de ejemplo, a continuación en la figura 5.17 se visualiza la reducción de ruido (aplicando erosión y dilatación) de una sección de una imagen de la base de

datos, variando el radio del círculo.



a) Parte de imagen binaria sin aplicación de operaciones morfológicas.

b) Radio de erosión: 2px
Radio de dilatación: 2px

c) Radio de erosión: 2px
Radio de dilatación: 3px

d) Radio de erosión: 3px
Radio de dilatación: 3px

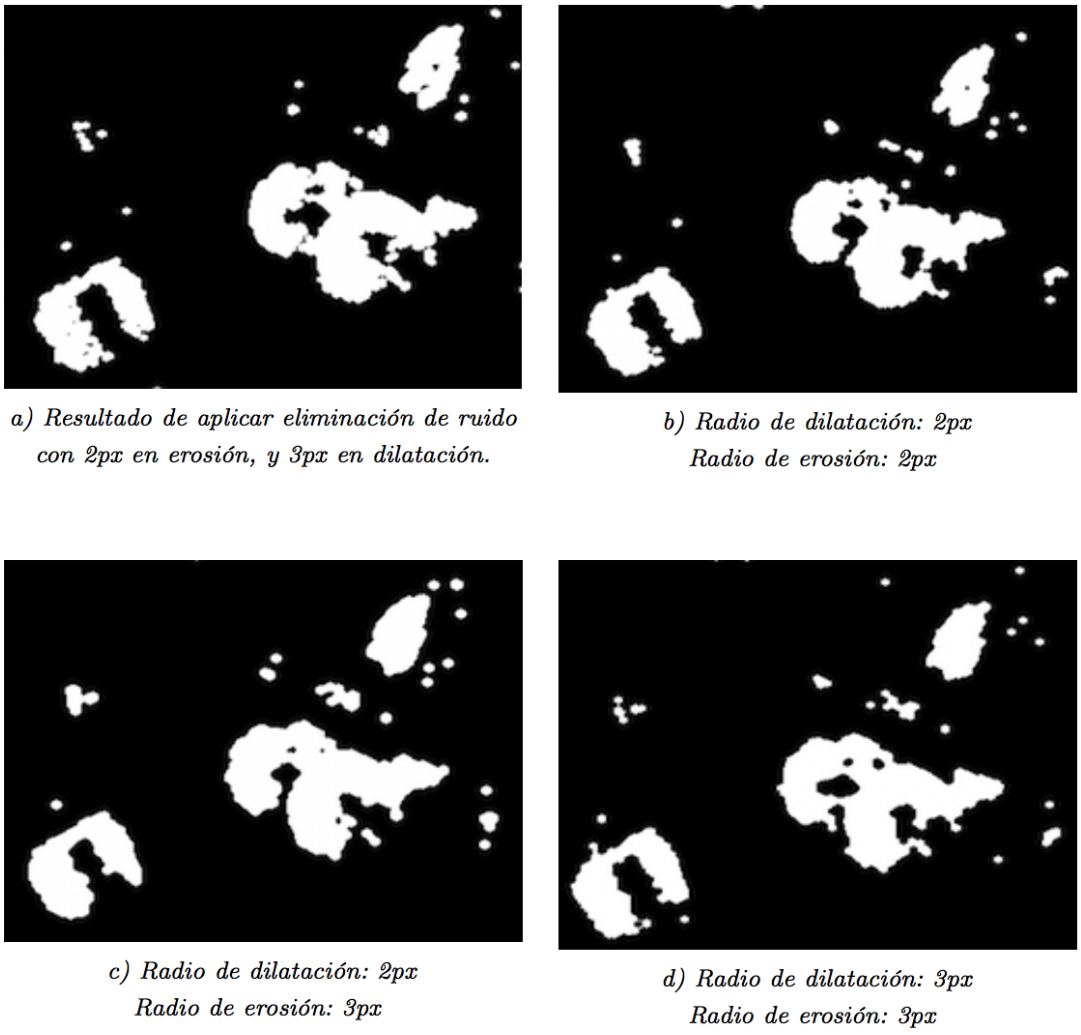
Figura 5.17: *Tabla de ejemplos aplicando operaciones morfológicas para eliminar ruido.*

Dilatación y erosión como alisado de las regiones

Una vez realizado la eliminación se ruido, se prosigue con el alisado de regiones y disminución de huecos internos. En la figura 5.18se visualizan ejemplos.

Resultados

Para obtener los valores óptimos de radio para la disminución de ruido y alisado de regiones, se iterará por los posibles valores de 1 a 10 píxeles de tamaño de radio del elemento estructurante. El indicador de los mejores resultados, será cómo ya se ha detallado, el valor FV.



a) Resultado de aplicar eliminación de ruido con 2px en erosión, y 3px en dilatación.

b) Radio de dilatación: 2px Radio de erosión: 2px

c) Radio de dilatación: 2px Radio de erosión: 3px

d) Radio de dilatación: 3px Radio de erosión: 3px

Figura 5.18: *Tabla de ejemplos aplicando operaciones morfológicas para alisado de regiones.*

Para esto se realizan los siguientes pasos:

1. Se obtiene la imagen binaria que representa la clasificación de los píxeles, y la imagen de la fotografía.
2. Se aplica a la imagen de la fotografía, la técnica de clasificación Árbol de decisión con valores óptimos. Se opta por esta técnica ya que es la que requiere menor tiempo de ejecución.
3. Se aplican las operaciones morfológicas a la imagen binaria resultado del reconocimiento de píxeles.
4. Se cuentan la cantidad de píxeles distintos e iguales que hay entre la imagen binaria de la base de datos y la imagen binaria resultado de aplicar las operaciones morfológicas, para calcular luego el valor de FV.

A continuación se visualiza cómo ejemplo en la figura 5.19, las distintas imágenes involucrando los pasos mencionados.

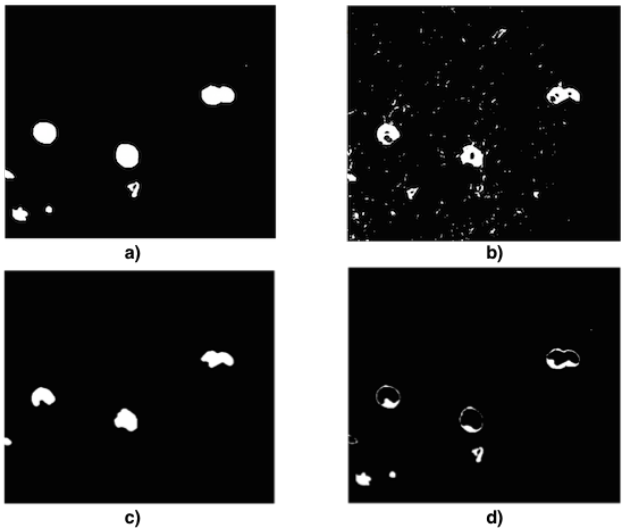


Figura 5.19: *a) Imagen binaria de la base de datos, b) Imagen binaria cómo resultado de reconocimiento de píxeles, c) Operaciones morfológicas aplicadas a la imagen b), d) imagen diferencia entre a) y c).*

Se evalúa utilizando Cross validation. Para realizar las evaluaciones, se iterará los radios de erosión y dilatación en el rango de 1-10 píxeles, y aplicando Árbol de decisión cómo técnica de clasificación con los valores de la tabla 5.2 de parámetros que fueron considerados como óptimos.

H rango menor	H rango mayor	S	Densidad de Bordes	FV
13 ± 0	$161,20 \pm 1,47$	47 ± 0	76	$0,62 \pm 6,76E-05$

Cuadro 5.2: Valores óptimos de H, S y Densidad de Bordes

En la figura 5.20 a) se visualiza para una iteración aleatoria de Cross validation, los resultados FV, se puede observar que los mejores resultados se encuentran en la diagonal (pequeña diferencia de radio entre erosión y dilatación), lo cual es razonable ya que si los radios difirieran mucho entre ellos, se presentaría un aumento o reducción de área de las regiones, ya que se estaría realizando más/menos erosión que dilatación. En 5.22 b) se presenta el porcentaje de mejora frente al óptimo valor FV de Árbol de decisión, y 5.22 b) una representación gráfica. Se puede deducir que en esta iteración, la mejora a Árbol de decisión en su máximo punto es del 10 %, un porcentaje significativo tomando en cuenta que se aplicó dicha mejora al valor óptimo de Árbol de decisión.

En la figura 5.21 se visualiza el resultado de FV para las 10 iteraciones de Cross validation evaluadas. Para las 10 iteraciones los mejores valores de radio fue 5px de erosión y 7px de dilatación.

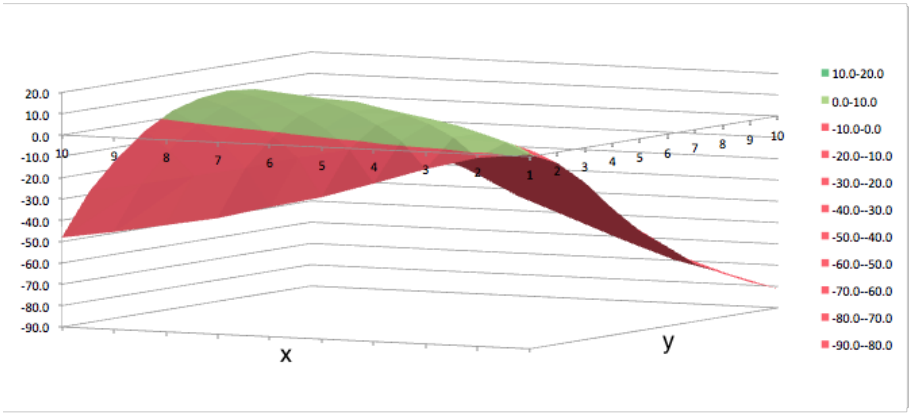
En la figura 5.22 se realiza una comparación visualizando la mejora gradual a la técnica Árbol de decisión, aplicando finalmente operaciones morfológicas.

	1	2	3	4	5	6	7	8	9	10
1	0.63	0.59	0.52	0.43	0.35	0.29	0.23	0.19	0.15	0.12
2	0.61	0.64	0.60	0.52	0.44	0.37	0.29	0.24	0.19	0.15
3	0.57	0.65	0.65	0.59	0.51	0.44	0.35	0.29	0.24	0.19
4	0.52	0.63	0.67	0.65	0.59	0.52	0.43	0.37	0.29	0.24
5	0.47	0.60	0.66	0.67	0.65	0.59	0.51	0.44	0.36	0.30
6	0.44	0.57	0.64	0.67	0.67	0.64	0.57	0.50	0.42	0.35
7	0.40	0.53	0.61	0.66	0.68	0.67	0.63	0.57	0.50	0.42
8	0.37	0.50	0.58	0.64	0.67	0.68	0.66	0.62	0.55	0.48
9	0.34	0.47	0.55	0.62	0.66	0.67	0.67	0.65	0.60	0.53
10	0.32	0.44	0.53	0.59	0.64	0.66	0.67	0.66	0.63	0.58

a)

	1	2	3	4	5	6	7	8	9	10
1	1.0	-5.1	-16.4	-30.7	-43.4	-53.0	-63.0	-69.7	-75.9	-81.1
2	-1.7	3.3	-2.8	-15.7	-29.6	-40.5	-52.7	-60.9	-68.9	-75.2
3	-8.1	4.6	4.5	-4.6	-17.5	-29.7	-43.1	-52.7	-61.9	-69.7
4	-16.0	1.8	7.7	5.0	-4.4	-15.3	-30.0	-40.8	-52.5	-61.4
5	-23.5	-3.4	6.2	8.9	4.6	-4.0	-17.8	-29.1	-41.4	-52.3
6	-29.3	-8.3	3.1	9.0	8.8	3.4	-8.3	-19.0	-31.7	-43.3
7	-35.6	-14.5	-1.8	6.5	10.0	8.6	1.4	-7.7	-19.6	-31.5
8	-40.0	-19.1	-5.9	3.5	8.8	9.6	6.0	-0.6	-11.4	-22.7
9	-44.3	-24.0	-10.6	-0.5	6.3	8.8	8.2	4.6	-3.1	-13.7
10	-48.0	-28.5	-15.1	-4.4	3.0	6.8	8.2	6.8	2.0	-5.9

b)



c)

Figura 5.20: *a) Tabla de FV aplicando operaciones morfológicas (radio de dilatación en filas, radio de erosión en columnas), b) Porcentaje de mejora (con respecto al valor FV de Árbol de decisión) luego de aplicar operaciones morfológicas, c) representación gráfica en 3D de la tabla b), con radio de dilatación en el eje y, radio de erosión en el eje x.*

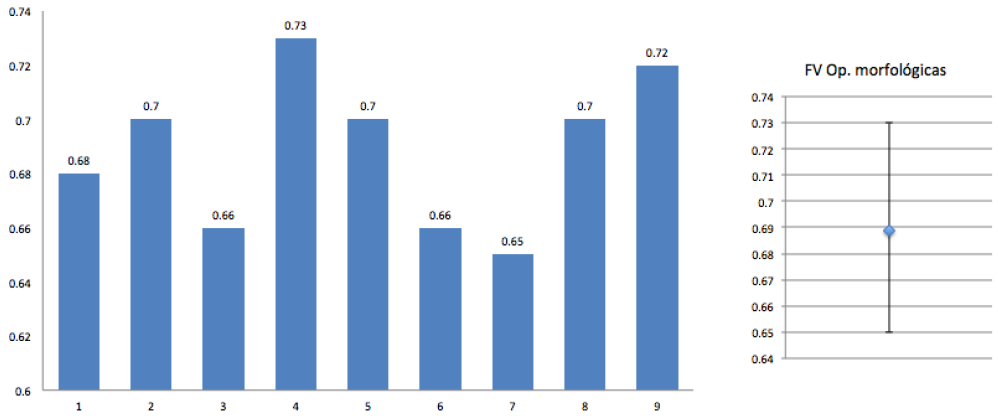


Figure 5.21: *Valores de FV para operaciones morfológicas aplicadas en las 10 etapas de Cross validation.*

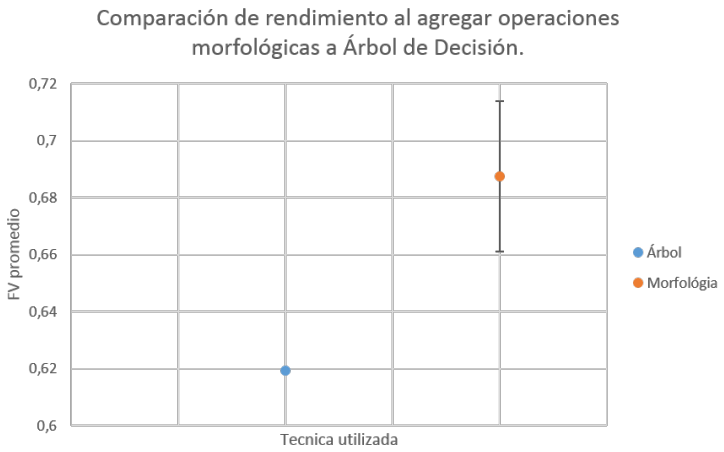


Figura 5.22: *Gráfica de FV comparando la efectividad de la técnica de Árbol de decisión al aplicar operaciones morfológicas.*

5.4. Detección de manzanas

A continuación ser presentan los resultados para la detección de manzanas, primero se presentan los resultados de detección de círculos de Hough y luego el procesamiento que se le hace a los mismos.

Detección de círculos de Hough

Como se detalló en la solución propuesta, el estudio de la detección de círculos se realiza variando:

1) Distancia mínima entre círculos detectados

2) Umbral del acumulador, cuanto menor sea este número, mayor será la sensibilidad de reconocimiento de círculos.

El radio mínimo y máximo se determinó observando la base de datos, los valores son 20 y 75 respectivamente.

Si bien la transformada de Hough puede encontrar círculos utilizando tanto la máscara en RGB como la máscara binaria, se probó que la máscara RGB generaba círculos dentro de las manzanas por las sombras o textura de las mismas. Por tal motivo se decidió utilizar la máscara binaria, un ejemplo de esto se puede observar en la figura 5.23.

Dado que en la base de datos se cuenta con el centro de las manzanas, para determinar si hay un acierto o no se debe definir una tolerancia entre los centros para considerar si es el mismo círculo. Para no ser muy restrictivos con el tamaño del radio de las manzanas, es que se utiliza para la tolerancia el valor del radio del círculo encontrado.

El FV obtenido para la detección fue $0,50 \pm 0,057$.

En la figura 5.24 se observan los valores óptimos encontrados, y sus respectivos valores de distancia mínima y acumulador.

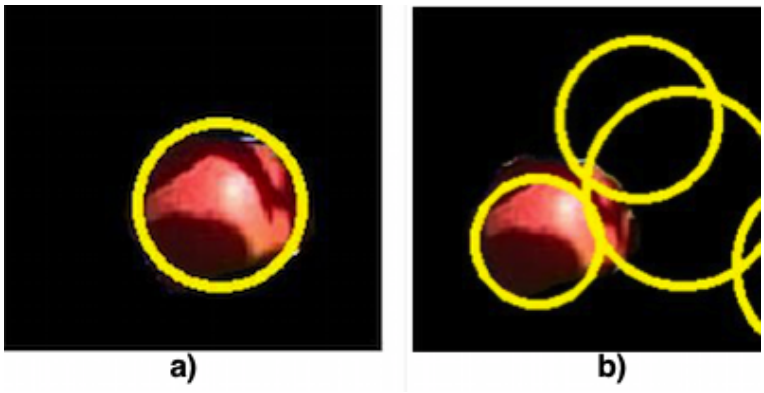
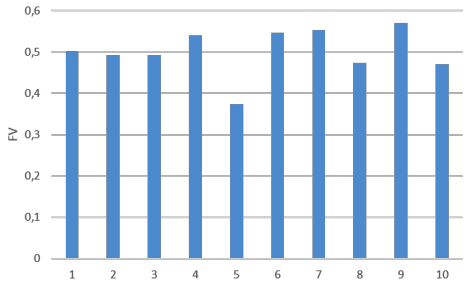


Figura 5.23: *a) Detección de círculos a partir de la imagen binaria b) detección de círculos a partir de la máscara RGB, donde se puede observar las detecciones erróneas de círculos debido a las sombras de las manzanas.*



Etapa	Distancia mínima	Acumulador	FV
1	75	7	0.50
2	66	8	0.49
3	66	8	0.49
4	60	7	0.54
5	69	7	0.37
6	69	7	0.55
7	66	8	0.55
8	66	8	0.47
9	66	8	0.57
10	66	8	0.47

Figura 5.24: *Valores óptimos obtenidos para la detección de círculos.*

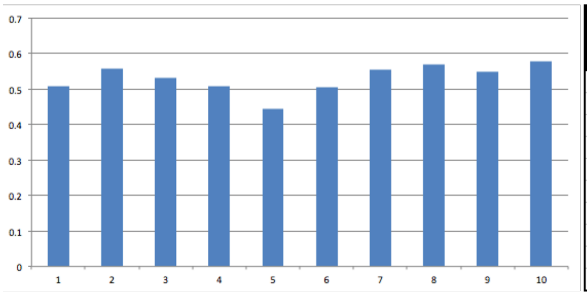
Pos-procesamiento en círculos

Una vez evaluado la detección de círculos utilizando la transformada de Hough para círculos, se planteó agregar una mejora a esto. Esta mejora consiste en descartar aquellos círculos que se considera se detectaron más de uno para la representación de una misma manzana. La forma de realizar esto, cómo ya se explicó en la solución propuesta, consiste en descartar los círculos que se intersectan, y que quitando su área intersectada, no incluyan una área restante significativa de región de manzana. Para los círculos no intersectados, también serán estudiados y se conservarán únicamente los círculos cuya área de la región también sea significativa. Para estas pruebas, el parámetro en estudio a variar es el porcentaje de área que se considera significativa.

Se realizarán evaluaciones utilizando Cross validation para imágenes, iterando por distintos valores de porcentaje de área mínima, iterando de a 5 %, dentro del rango de 15-50 %. Se realizarán pruebas conservando únicamente los círculos cuyas áreas de región manzana incluidas (área no intersectadas con otros círculos) sea mayor al porcentaje de la iteración actual.

En la gráfica de la figura 5.25 se visualizan los resultados según valor de FV. Con una media de $0{,}53 \pm 0{,}04$, donde el porcentaje de área mínima, siempre se mantuvo

en el 20 %. En términos de cantidad de manzanas detectadas, se obtuvo una media de 98 manzanas encontradas (frente a un total de 196 manzanas presentes) y tuvo 72 equivocaciones



Etapa	Distancia mínima	Acumulador	Porcentaje área mínima	FV
1	48	8	20	0.51
2	48	8	20	0.56
3	48	8	20	0.53
4	48	8	20	0.51
5	48	8	20	0.45
6	48	8	20	0.51
7	48	8	20	0.55
8	48	8	20	0.57
9	48	8	20	0.55
10	48	8	20	0.58

Figura 5.25: *Gráfica FV para procesamiento pos círculos, para 10 muestreos de la base de datos, evaluando con Cross validation.*

Comparando los resultados con la detección de círculos, se concluye que se presenta una leve mejora, cómo se visualiza en la gráfica de la figura 5.26.

Algo que se debe tener en cuenta, es que el pos procesamiento de círculos, puede traer cómo consecuencias eliminar detecciones que son válidas, pero debido al bajo porcentaje de área de región, son descartadas. En las imágenes de la figura 5.27, se presentan 2 ejemplos donde en un caso se obtienen resultados favorables, y en el otro no.

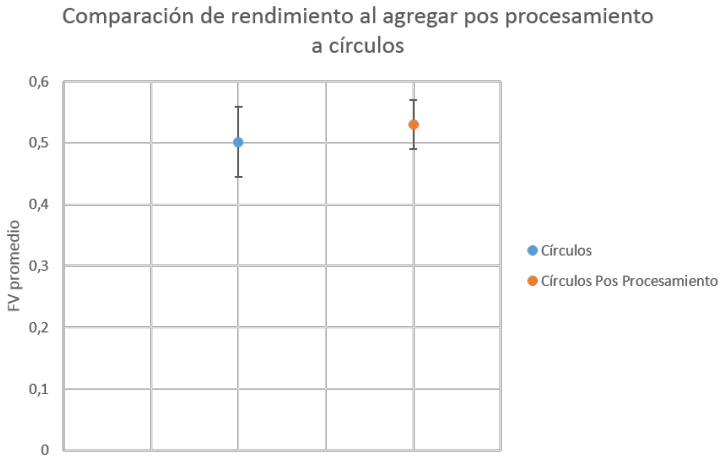


Figura 5.26: *Comparación de efectividad al agregar pos procesamiento de círculos.*

Figura 5.27: *Estudio del área de la región dentro del círculo. En la imagen a) se visualizan los círculos sin pos procesamiento, mientras que en la imagen b) se puede ver como se eliminó un circulo cuya área era nula. En la imagen c) y d) se puede ver como esto puede traer consigo eliminar manzanas válidas.*

5.5. Resultados en imágenes

Luego del estudio de las técnicas de reconocimiento de píxeles y detección de manzanas, se visualizan a continuación algunas imágenes procesadas por algunas de las técnicas estudiadas (entrenando las mismas sobre otro conjunto de imágenes), con el fin de observar los resultados de forma visual en las imágenes.

Se utilizará la técnica Árbol de decisión cómo técnica para el reconocimiento, se aplicará operaciones morfológicas y en cuanto a la detección se aplicará el reconocimiento de círculos con la transformada de Hough y finalmente con pos procesamiento de estos círculo para quitar intersecciones y regiones contenidas con área despreciable. Todas las técnicas serán ejecutadas con los valores óptimos estudiados.

Figura 5.28: *Imágenes resultados, se puede observar una buena detección frente a variabilidad de balance de blancos, distintas especies de manzanas, y distintos tamaños de las mismas. También se encuentra presente pequeñas falsas detecciones (regiones oscuras).*

Figura 5.29: *Imágenes resultados, se puede observar una buena detección frente a variabilidad de balance de blancos, distintas especies de manzanas, y distintos tamaños de las mismas. También se encuentra presente pequeñas falsas detecciones (regiones oscuras).*

Capítulo 6

Conclusiones y Trabajos futuros

En este último capítulo se presentan las conclusiones y trabajos a futuro.

6.1. Conclusiones

Base de datos

Unos de los grandes desafíos que se logró superar fue lograr componer una base de datos de fotografías de manzanas en una huerta. Para ello se realizó el etiquetado de cada píxel por fotografía, así como el etiquetado de los centros de las manzanas presentes en cada imagen. Se contó entonces para la ejecución a lo largo de todo el proyecto, con un total de 266 imágenes con sus respectivas etiquetas de píxeles y centros de manzanas.

Gracias a la base de datos se pudieron evaluar las distintas técnicas de clasificación y detección de manzanas utilizando fotografías adquiridas en ambientes no controlados, por distintas cámaras, a distintas distancias, con distintos balances de blancos y distinta sobre-exposición. Se estudió que tomando en cuenta todas estas variabilidades, los resultados fueron prometedores, lo cual demuestra una cierta flexibilidad de adaptación por parte de las técnicas a cualquier huerta.

Es de destacar que queda disponible para futuras investigaciones una base de datos completa y abierta.

Clasificación de píxeles

Para la clasificación de píxeles, se realizó análisis y pruebas de las técnicas: Árbol de decisión, Knn (K-Vecinos más cercanos) y SVM (Support Vector Machine). En las tres técnicas se utilizaron las mismas características determinantes que permiten identificar los píxeles manzanas, estas fueron: Matiz (H), Saturación (S), y Densidad de bordes. Debido a que las fotografías fueron adquiridas directamente en huertas, hay presentes pocos elementos con matiz rojo, exceptuando las manzanas rojas, es por esto que la característica matiz, fue determinante para la clasificación de píxeles de manzanas rojas. A su vez, la saturación permite descartar zonas muy claras, donde el matiz no alcanza para determinar si es rojo o no. Otra característica importante es la densidad de bordes, ya que las manzanas presentan una textura lisa, frente a una textura rugosa como lo son las hojas, ramas, etc.

También cómo mejora del reconocimiento de píxeles, se optó por aplicar operaciones morfológicas para mejorar la definición en las regiones clasificadas como manzanas, y en la eliminación del ruido.

Por otra parte, se concluyó que la técnica Árbol de decisión presenta una efectividad levemente menor a la de Knn y SVM, si bien esto es una desventaja para dicha técnica, su principal ventaja es que los costos de procesamiento son menores en comparación con Knn y SVM, ya que no requiere búsqueda en ejecución (como si lo requiere Knn) ni de entrenamiento previo (como sí lo requiere Knn y SVM). SVM presenta las ventajas de las técnicas de Knn y Árbol de decisión, ya que una vez realizado el entrenamiento, no presenta grandes costos de procesamiento, y la efectividad de dicha técnica es superior a la de Árbol de decisión, siendo prácticamente la misma a la de Knn.

En cuanto a la precisión y sensibilidad de cada técnica estudiada:

- Árbol de decisión presentó que en el 69 % de los casos, de los píxeles que clasificó como manzana, efectivamente fueron píxeles manzanas, mientras que con Knn y SVM fue de 66 % y 81 % respectivamente.

- En cuanto a la cantidad de píxeles clasificados correctamente como manzana, frente al total de píxeles manzanas, Árbol de decisión tuvo una media del 56 %, mientras que Knn y SVM fue de 64 % y 53 % respectivamente.

Detección de manzanas

Para la detección de manzanas, se utilizó la transformada de Hough para círculos, ya que las manzanas en las fotografías presentan una forma muy próxima a la circular. Luego, se realizó procesamiento para descartar aquellas detecciones de círculos conteniendo poca cantidad de píxeles manzanas, y descartar también círculos detectados repetidas veces en una misma manzana, obteniendo una leve mejora. En cuanto a las técnica de reconocimiento de círculos y el pos procesamiento de los mismos para quitar círculos intersectados y con regiones con área mínima, se obtuvo una media de 98 manzanas encontradas (frente a un total de 196 manzanas presentes) y tuvo 72 equivocaciones.

6.2. Trabajos a futuro

Cómo trabajo a futuro se podría implementar una alternativa a la clasificación de píxeles haciendo uso de las tres técnicas de clasificación ya estudiadas. Esta mejora consistiría en obtener la clasificación de cada una de las tres técnicas (Árbol de decisión, Knn y SVM) de forma simultánea para cada píxel entrante, y finalmente clasificar el mismo dependiendo del resultado devuelto de los 3 clasificadores, a esto se lo conoce cómo "Sistema de múltiples clasificadores" [18, Página100]. Este método consiste en clasificar el píxel a partir de la votación de cada clasificador, donde es posible ponderar el voto de cada uno, según el margen de error de cada técnica. Un aspecto importante a destacar, es que mediante la combinación de clasificadores se obtienen mejores resultados que con los clasificadores utilizados de forma individual, así como también permite resaltar las bondades de los clasificadores al mismo tiempo que se atenúan o desaparecen las desventajas [13].

Otro trabajo a futuro sería la detección de manzanas a partir de Template matching. Esta línea de trabajo difiere a la propuesta en cuanto a la clasificación de píxeles para luego el estudio de los mismos para la detección de manzanas. Template matching es el nombre genérico para un conjunto de técnicas que cuantifica similitudes de dos imágenes digitales, o una porción de las mismas, con el objetivo de establecer si son iguales o no [11]. Bajo el contexto de detección de manzanas, se podría aplicar Template matching partiendo de una nueva base de datos compuesta por las regiones que

contienen únicamente a una manzana, extráyendolas a partir de las imágenes de la base de datos ya ofrecida. Con esto se lograría entonces distintos templates de lo que es una manzana, para luego buscar una región similar contenida en una imagen entrante, y en caso que encuentre dicha región, marcar como presencia de manzana allí. Unos de los posibles desafíos, es la elaboración de un conjunto apropiado de templates que abarque las distintas especies de manzanas, en distintas posiciones en las que se podrían encontrar, y tomando en cuenta también otras variaciones.

Todo el proyecto se ha basado a partir de imágenes, y el enfoque ha sido la detección de imágenes. Si se deseara el reconocimiento en tiempo real a partir de vídeo, un trabajo a futuro podría ser aplicar técnicas que involucra el procesamiento entre una imagen (o cuadro) y la siguiente, de tal forma que se preserve la ubicación de la manzana en un cuadro, y para el siguiente buscar la detección de la misma manzana en una ubicación próxima.

Glosario

Visión por computadora: Consiste en la adquisición, procesamiento, clasificación y reconocimiento de imágenes digitales.

Píxel: Elemento básico de una imagen.

Imagen: Arreglo bidimensional de píxeles con diferente intensidad luminosa, escala de gris. Si la intensidad luminosa de cada píxel se representa por n bits, entonces existirán 2^n escalas de gris diferentes. Matemáticamente, una imagen se representa por $r = f(x, y)$, donde r es la intensidad luminosa del píxel cuyas coordenadas son (x, y).

Para obtener una imagen a color se deben transformar primero los parámetros cromáticos en eléctricos y representar los colores, lo cual puede realizarse de diferentes maneras, dando lugar a diferentes espacios de colores o mapas de color.

Brillo: Indica si un área está más o menos iluminada.

Tono: Indica si un área parece similar al rojo, amarillo, verde o azul o a una proporción de ellos.

Luminosidad: Brillo de una zona respecto a otra zona blanca en la imagen.

Saturación: Se basa en la pureza del color, un color muy saturado tiene un color vivo e intenso, mientras que un color menos saturado parece más descolorido y gris. Sin saturación, un color se convierte en un tono de gris.

Histograma: Representación gráfica de una variable en forma de barras, donde la superficie de cada barra es proporcional a la frecuencia de los valores representados, ya sea en forma diferencial o acumulada.

HSV: Espacio de color donde sus componentes son Matiz(Hue), Saturación(Saturation) y Valor (value).

RGB: Espacio de color donde sus componentes son Rojo, Verde y Azul.

SVM: Support Vector Machine.

KNN: K - Nearest Neighbor.

CV: Cross Validation.

FV: El valor FV es la medida de precisión de un test, se considera como una media armónica que combina los valores de precisión y de sensibilidad.

OpenCV: Biblioteca para el procesamiento de imágenes.

Canny: Algoritmo para la detección de bordes.

Anexo 1: Manual de usuario

6.3. Etiquetado de la base de datos

Para realizar el etiquetado de la base de datos se deben ubicar todas las imágenes a etiquetar en una misma carpeta y luego ejecutar el programa indicando la opción 1 del menú principal del programa: *Etiquetado de la Base de Datos.*

Se visualizará cada imagen que se encuentre en la carpeta, siendo necesario seguir los siguientes pasos:

1. Marcar el contorno de las manzanas.
2. Presionar la tecla 's' y seleccionar el área para pintar en el interior del contorno.
3. Presionar la tecla 'o' para confirmar la selección.
4. Presionar la tecla 'g' para guardar.
5. Presionar la tecla 'p' si se quiere visualizar la mascara.
6. Presionar la tecla 'c' para marcar el centro de cada manzana.
7. Presionar la tecla 'f' para indicar que se terminaron de marcar los centros de las manzanas.

6.4. Ajuste de Parámetros Árbol de Decisión

Si se desea testear los valores de Hue, Saturation y Densidad de bordes para los cuales se obtienen mejores resultados en la clasificación con una nueva base de datos etiquetada, se debe seleccionar la opción 2 del menú principal: *Ajuste de parámetros del Árbol de Decisión.*

Como resultado se obtiene un archivo de texto llamado resultado_ajuste_paramatros_arbol.csv con los mejores parámetros.

6.5. Ajuste de Parámetros Knn

Si se desea elegir el K para el cual se obtienen mejores resultados en la clasificación utilizando Knn con una nueva base de datos etiquetada, se debe seleccionar la opción 3 del menú principal: *Ajuste de k para Knn.*

Como resultado se obtiene un archivo de texto llamado resultado_ajuste_k_Knn.csv con los mejores parámetros.

6.6. Ajuste de Parámetros SVM

Si se desea elegir el gamma y c para los cuales se obtienen mejores resultados en la clasificación utilizando SVM con una nueva base de datos etiquetada, se debe seleccionar la opción 4 del menú principal: *Ajuste de c y gamma para SVM.*

Como resultado se obtiene un archivo de texto llamado resultado_ajuste_paramatros_svm.csv con los mejores parámetros.

6.7. Clasificar los píxeles de una imagen

Si se desea clasificar los píxeles de una imagen se debe seleccionar la opción 5 del menú principal: *Clasificar píxeles de imagen.* A continuación debe indicar el número de la técnica a utilizar según el menú desplegado y el nombre de la imagen a clasificar. Se mostrará en pantalla el resultado.

6.8. Ajustar parámetros operaciones morfológicas

Si se desea ajustar el radio para las operaciones morfológicas debe seleccionar la opción 6 del menú principal: *Ajuste de parámetros de las operaciones morfológicas.*

Como resultado se obtiene un archivo de texto llamado resultado_ajuste_paramatros_opMorf.csv con los mejores parámetros.

6.9. Ajuste de Parámetro para la detección de círculos

Si se desea ajustar la distancia mínima entre círculos y el umbral del acumulador para la detección de círculos debe seleccionar la opción 8 del menú principal: *Ajuste de parámetros para la detección de círculos.*

Como resultado se obtiene un archivo de texto llamado resultado_ajuste_paramatros_círculos.csv con los mejores parámetros.

6.10. Guardar nuevos parámetros

Si se desea cambiar los parámetros de algunas de las técnicas, se debe abrir el archivo de configuración: «configuration.conf» y cambiar los parámetros que desee.

6.11. Detección de manzanas en una imagen

Si se desea detectar manzanas en una imagen se debe seleccionar la opción 9 del menú principal: *Detectar manzanas en una imagen.* A continuación debe indicar el número de la técnica a utilizar para la clasificación de pixeles según el menú desplegado y el nombre de la imagen. Se mostrará en pantalla el resultado.

6.12. Detección de manzanas en una vídeo

Si se desea detectar manzanas en un vídeo se debe seleccionar la opción 10 del menú principal: Detectar manzanas en un vídeo. A continuación debe indicar el número de la

técnica a utilizar para la clasificación de pixeles según el menú desplegado y el nombre del vídeo. Se mostrará en pantalla el resultado.

Bibliografía

[1] Circles Hough. http://docs.opencv.org/modules/imgproc/doc/feature_detection.html?highlight=houghcircles#houghcircles. Accedido: 2015-06-21.

[2] FLANN (Fast Approximate Nearest Neighbour Search. http://docs.opencv.org/modules/flann/doc/flann_fast_approximate_nearest_neighbor_search.html. Accedido: 2015-06-13.

[3] Hough iris image. http://answers.opencv.org/upfiles/1370738567862134.jpg. Accedido: 2015-06-02.

[4] Hough Tennis Image. http://i.stack.imgur.com/sqyEx.jpg. Accedido: 2015-06-03.

[5] HSL and HSV. https://en.wikipedia.org/wiki/HSL_and_HSV. Accedido: 2015-06-13.

[6] OpenCV. http://opencv.org/. Accedido: 2015-06-12.

[7] AD Aggelopoulou, Dionysis Bochtis, S Fountas, Kishore Chandra Swain, TA Gemtos, and GD Nanos. Yield prediction in apple orchards based on image processing. *Precision Agriculture*, 12(3):448–456, 2011.

[8] D. Ballard and C. Brown. Computer vision. 1982.

[9] Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM Sigkdd Explorations Newsletter*, 6(1):20–29, 2004.

[10] Christopher M Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[11] Roberto Brunelli. *Template matching techniques in computer vision: theory and practice*. John Wiley & Sons, 2009.

[12] Edward R Dougherty, Roberto A Lotufo, and The International Society for Optical Engineering SPIE. *Hands-on morphological image processing*, volume 71. SPIE press Bellingham, 2003.

[13] María Guijarro Mata-García. Combinación de clasificadores para identificación de texturas en imágenes naturales: nuevas estrategias locales y globales. 2010.

[14] Anil K Jain, Robert PW Duin, and Jianchang Mao. Statistical pattern recognition: A review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(1):4–37, 2000.

[15] Wei Ji, Dean Zhao, Fengyi Cheng, Bo Xu, Ying Zhang, and Jinjing Wang. Automatic recognition vision system guided for apple harvesting robot. *Computers & Electrical Engineering*, 38(5):1186–1195, 2012.

[16] Antonio Ramón Jiménez, Anil K Jain, R Ceres, and JL Pons. Automatic fruit recognition: a survey and new results using range/attenuation images. *Pattern recognition*, 32(10):1719–1736, 1999.

[17] Miroslav Kubat, Stan Matwin, et al. Addressing the curse of imbalanced training sets: one-sided selection. In *ICML*, volume 97, pages 179–186. Nashville, USA, 1997.

[18] Ludmila I Kuncheva. *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons, 2004.

[19] Raphael Linker, Oded Cohen, and Amos Naor. Determination of the number of green apples in rgb images recorded in orchards. *Computers and Electronics in Agriculture*, 81:45–57, 2012.

[20] Thomas B Moeslund. *Introduction to video and image processing: Building real systems and applications*. Springer Science & Business Media, 2012.

[21] Andrew W Moore. An intoductory tutorial on kd-trees. 1991.

[22] J Rakun, D Stajnko, and D Zazula. Detecting fruits in natural scenes by using spatial-frequency based texture analysis and multiview geometry. *Computers and Electronics in Agriculture*, 76(1):80–88, 2011.

[23] Omri Safren, Victor Alchanatis, Viacheslav Ostrovsky, and Ofer Levi. Detection of green apples in hyperspectral images of apple-tree foliage using machine vision. *Transactions of the ASABE*, 50(6):2303–2313, 2007.

[24] Denis Stajnko, Miran Lakota, and Marko Hočevar. Estimation of number and diameter of apple fruits in an orchard during the growing season by thermal imaging. *Computers and Electronics in Agriculture*, 42(1):31–42, 2004.

[25] Amy L Tabb, Donald L Peterson, and Johnny Park. Segmentation of apple fruit from video via background modeling. *ASABE paper*, (063060), 2006.

[26] Mario Miranda Terence Robinson. Una vision para huertos futuros de manzana y pera. *Department of Horticulture Cornell University Geneva NY 14456*, 2005.

[27] Qi Wang, Stephen Nuske, Marcel Bergerman, and Sanjiv Singh. Automated crop yield estimation for apple orchards. In *Experimental Robotics*, pages 745–758. Springer, 2013.

[28] Andrew R Webb. *Statistical pattern recognition*. John Wiley & Sons, 2003.

[29] Linghe Yang, J Dickinson, QMJ Wu, and S Lang. A fruit recognition method for automatic harvesting. In *Mechatronics and Machine Vision in Practice, 2007. M2VIP 2007. 14th International Conference on*, pages 152–157. IEEE, 2007.

[30] Jun Zhao, Joel Tow, and Jayantha Katupitiya. On-tree fruit recognition using texture properties and color data. In *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 263–268. IEEE, 2005.