UNIVERSIDAD DEL PAPALOAPAN

Campus Loma Bonita

INGENIERÍA EN COMPUTACIÓN

DETECCIÓN DEL ÁREA FOLIAR DE UNA HOJA DE JITOMATE MEDIANTE CÓDIGOS DE CADENA Y PROCESAMIENTO DE IMÁGENES

TESIS

QUE PARA OBTENER EL TÍTULO DE:
INGENIERO EN COMPUTACIÓN

PRESENTA:

GENARO DE JESÚS MENDOZA RASCÓN

ASESOR DE TESIS:

M.C. EDUARDO ORTIZ HERNÁNDEZ

LOMA BONITA, OAXACA. 2016

UNIVERSIDAD DEL PAPALOAPAN

INGENIERÍA EN COMPUTACIÓN

LA PRESENTE TESIS TITULADA **"DETECCIÓN DEL ÁREA FOLIAR DE UNA HOJA DE JITOMATE MEDIANTE CÓDIGOS DE CADENA Y PROCESAMIENTO DE IMÁGENES"** PRESENTADA POR EL SUSTENTANTE DE LICENCIATURA **C. GENARO DE JESÚS MENDOZA RASCÓN** BAJO LA DIRECCIÓN DEL **M.C. EDUARDO ORTIZ HERNÁNDEZ**, HA SIDO REVISADA Y ACEPTADA POR EL COMITÉ EXAMINADOR PARA SER DEFENDIDA EN EL EXAMEN PROFESIONAL Y OBTENER EL TÍTULO DE INGENIERO EN COMPUTACIÓN.

M.C. EDUARDO ORTIZ HERNÁNDEZ
ASESOR

Dr. MIGUEL ÁNGEL SÁNCHEZ HERNÁNDEZ	**Dr. EDUARDO SÁNCHEZ SOTO**	**M.C. ALMA ALHELÍ PEDRO PÉREZ**
PRESIDENTE	SECRETARIO	VOCAL

LOMA BONITA, OAXACA. 2016

Dedicatoria

La presente tesis se la dedico especialmente a mi familia por que gracias a su apoyo y ejemplo que me brindaron cada segundo de mi vida, he llegado a realizar esta meta. Principalmente a mi padre y a mi abuelita por brindarme todo lo necesario para salir adelante.

Agradecimientos

A mi padre Florencio, a mi abuelita Manuela, a mis hermanos Martin y Javier, a mis tías Rosa, Rica, Avelina, a mi tío José y a mis primos por darme su apoyo y motivación para seguir adelante. A Dios por permitirme llegar a esta etapa de mi vida.

A mi asesor de tesis el M.C. Eduardo Ortiz Hernández por su apoyo incondicional, su paciencia y por compartir sus conocimientos conmigo para concluir mi tesis.

A mis revisores M.C Alma Alhelí Pedro Pérez, Dr. Eduardo Sánchez Soto, Dr. Miguel Ángel Sánchez Hernández por el tiempo que me brindaron en la revisión y corrección de mi tesis.

A mis profesores por compartir sus conocimientos durante estos cinco años de la carrera.

A mis compañeros Leydy, Carmen, Zuleima, Marleny, Luis Enrique, José Luis, Ezequiel, Arturo, Alex, Nieto, Christian por compartir grandes momentos en fiestas, en desveladas realizando nuestros proyectos y por el apoyo que siempre nos brindamos como grupo.

A mis amigos Guadalupe, Abigail, Berenice, Salome, Norma, Lalo, Julio, Julian, Raul, Ripol, Pupe por brindarme su amistad, ayudarme a desestresarme un rato los fines de semana. A Daniela, Cindy, Osiris, Marielva, Juan, Jonas por brindarme su amistad y apoyo. A mi equipo de beisbol Edgar, Americo, Juan, Jair, Cheo, Cesar por hacer con ellos un poco de deporte.

A mis patrones y compañeros de trabajo por apoyarme durante estos años y darme consejos para ser mejor persona.

Resumen

El área foliar de una planta se refiere a la cantidad de superficie de hoja que posee, y para su determinación se utilizan diferentes métodos tanto destructivos como no destructivos. El objetivo de este trabajo fue aplicar un método no destructivo para el cálculo del área foliar de una hoja de jitomate a partir de una fotografía tomada con la cámara del dispositivo, creando así una aplicación móvil.

Posteriormente ésta se binariza a partir de un umbral adecuado mediante el método de dos picos y a su vez la imagen es rotada hasta obtener la vista con mayor área. Debido a que dos foliolos de una hoja pueden encontrarse sobrepuestos (ocultando parte del área de uno), es necesario separar y reconstruir cada uno para obtener mejor su área. Para la reconstrucción de los foliolos primero se crearon dos modelos a partir de 100 foliolos de diferentes hojas en base al borde, estos modelos son ajustados de manera vertical y reducidos a una escala para las comparativas posteriores.

A la imagen capturada se le detecta si dos foliolos se encuentran uno encima del otro en forma vertical comparando el código de cadena del foliolo con los modelos generados, y alineando las dos cadenas buscando la parte que hace falta para reconstruir el foliolo. Por último se calcula el área foliar contando el número de pixeles en color negro existentes en la imagen digitalizada.

Palabras clave: procesamiento de imágenes, área foliar, foliolos, códigos de cadena.

Abstract

The foliar area of a plant refers to the quantity of leaf surface that it possess, and to make this determination various methods can be used, such as destructive and non destructive methods. The objective of this work was to use a non destructive method to calculate the size of the area of a tomato leaf, by creating a mobile application, starting with a photograph taken with a cellular phone camera.
The next step was to binarize the image using a threshold obtained by the two peaks method and rotated until the view with the highest area was obtained. Due to the fact two leaflets can overlay (hiding part of the area of one of them), it was necessary to separate and reconstruct each one of them in order to better calculate the size of its area. For the reconstruction process of the leaflets, 100 leaflets divided into two models of different types of tomato leaves based on the edge of their boarder were created, these models were adjusted in a vertical form and reduced in scale for further comparisons.
A captured image is detected if two leaflets overlay vertically comparing the leaf chain code with generated models, and aligning the two chains looking for the part it takes to reconstruct the leaflet. Finally the foliar area containing the number of black pixels in the digital image was calculated.

Keywords: image processing, foliar area, leaflets, chain code.

Índice general

Índice de figuras

Capítulo 1

Introducción

El jitomate es una especie de la familia de las solanáceas originaria de América de gran importancia debido a que se utiliza para el consumo diario y por el alto contenido de proteínas que contiene. Por tal motivo, los investigadores de la carrera de Ingeniería Agrícola Tropical pretenden obtener un fruto de excelente calidad con la mínima utilización de recursos, como pueden ser la cantidad de agua, de fertilizantes, etc. Para calcular los recursos antes mencionados se recurrirá a la obtención del área foliar por ser un factor representativo del desarrollo del jitomate.

El área foliar de una planta se refiere a la cantidad de superficie de hoja que ella posee [10], la determinación es fundamental en estudios de nutrición y crecimiento vegetal, con ésta se puede determinar la acumulación de materia seca, el metabolismo de carbohidratos, el rendimiento y calidad de la cosecha [15].

En la actualidad existen diversos procedimientos para la determinación del área foliar mediante métodos destructivos. Algunos de los cuales consisten en cortar varias hojas para colocarlas sobre una lámina y así medir su altura y anchura. Estas dos mediciones se ponderan mediante un factor específico para cada tipo de hoja y planta para determinar así el área foliar en cm^2. Como prodrá entenderse inmediatamente este método además tiene la desventanja de ser aproximado debido a que no se obtiene el área real de las hojas.

Para evitar la destrucción de las hojas se propone implementar un método no destructivo, en el cual se toma una fotografía digital a cada hoja de la planta mediante una aplicación realizada en la plataforma de Android. La obtención del área foliar se realizará utilizando métodos de procesamiento de imágenes directamente sobre la fotografía.

1.1. Planteamiento del problema

En la binarización de la imagen es necesario calcular un umbral adecuado para cada captura, debido a que la cámara fotográfica en los dipositivos moviles no cuentan con la misma resolución y la misma calidad. Existen dos problemas que ocasionan que el área no sea la adecuada: la imagen es capturada en diferente posición, se modifica la imagen ajustando el ángulo, es decir, se tiene que rotar la imagen hasta encontrar un área mayor. Cuando dos foliolos se encuentran uno encima de otro, se procede a separar y reconstruir cada uno.

1.2. Justificación

La aplicación se desarrolló con los siguientes propósitos fundamentales.

1. Crear una herramienta no destructiva para el cálculo del área foliar del jitomate.
2. Reducir el tiempo para realizar el cálculo, lo que conlleva beneficios como reducir la exposición de las personas a las altas temperaturas que existen en la región de la baja cuenca del Papaloapan.
3. El cálculo con la aplicación es más exacto que medir hoja por hoja de la planta como se realiza en la actualidad.
4. Los resultados obtenidos serán útiles para estimar que cantidad de agua, fertilizante, etc. serán utilizados para un adecuado crecimento de las plantas de jitomate.

1.3. Alcances y Limitaciones

1.3.1. Alcances

1. En el presente trabajo se buscarán algoritmos que obtengan el umbral adecuado y el alineamiento de secuencias en un menor tiempo de ejecución.
2. El cálculo del área foliar de la hoja de jitomate será obtenido en unidades cuadradas.
3. La reconstrucción de los foliolos se hará únicamente a una forma, donde dos foliolos se encuentran sobrepuestos y alineados de manera vertical, demostrando que se pueden reconstruir foliolos apartir de los códigos de cadena Freeman y el alineamiento de secuencias.

1.3.2. Limitaciones

1. En un trabajo a futuro se pretende calcular el área foliar de la hoja de jitomate en cm^2 utilizando la luz del autofocus que tienen las cámaras en algunos dispositivos móviles.
2. Existe muchas formas de encontrar dos o más foliolos sobrepuestos por lo cual se pretende realizar un algoritmo para detectar estos casos y así reconstruirlos.

1.4. Objetivo General

Determinar mediante una fotografía el área foliar que existe en una hoja de jitomate.

1.4.1. Objetivos Particulares

1. Revisar el estado del arte en relación con la estimación de área foliar.
2. Aprender a utilizar la plataforma Android.
3. Investigar algoritmos para realizar transformaciones lineales en imágenes.

4. Realizar un algoritmo para corregir en una fotografía los ángulos de inclinación mediante transformaciones lineales.
5. Realizar un algoritmo para calcula el área foliar en hojas de jitomate.
6. Implementar todos los métodos en Android.

1.5. Hipótesis

La aplicación tomará una fotografía en diferentes ángulos la cual permitirá no cortar ninguna hoja de jitomate para su análisis, podrá ser capturada mediante un dispositivo móvil para obtener el área foliar de una planta de jitomate optimizando el tiempo de cálculo.

Capítulo 2

Estado del arte

2.1. Método de Montgomery

El método propuesto por Montgomery [10] sugiere, que el área de las hojas se calcula a partir de mediciones lineales, como se observa en Figura 2.1, donde el área foliar (LA) puede estimarse utilizando la siguiente ecuación (2.1).

$$LA = bLW \tag{2.1}$$

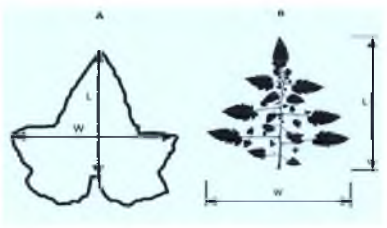


Figura 2.1: Hoja de jitomate medida mediante el método de Montgomery.

Donde:

b es un valor de una hoja empírica de forma que difiere entre especies.

L es la logitud de la hoja.

W es la anchura de la hoja.

2.2. Estimación del área foliar en distintos cultivares de jitomate (*Lycopersicon esculentum Mill*) utilizando medidas foliares lineales

El objetivo fue obtener una ecuación alométrica entre las dimensiones lineares y de superficie que permitiera estimar rápidamente el área foliar de un cultivo. Para tal fin se seleccionaron al azar ocho hojas de diferentes tamaños en plantas de cultivares antiguos y modernos de jitomate. Los cultivares antiguos utilizados fueron: Platense, Marmande, 7718 y Carmelo, cultivados al aire libre durante la primavera con una densidad de $23{,}400$ plantas por hectárea. Los cultivares modernos fueron Splendid, Salvador, Bonanza, FA 144, Agroflora 4, BHN 9086 y BHN CO128, cultivados en invernadero con una densidad de $25{,}000$ plantas por hectárea. En las hojas muestreadas se estudiaron las relaciones alométricas entre el largo y ancho máximo y el área foliar individual [2].

2.3. Estimación del área foliar a partir de observaciones morfológicas convencionales en *Morus alba var. Acorazonada*

El objetivo fue establecer un modelo de regresión para estimar el área de las hojas a partir de la medición del largo y el ancho de la lámina, para lo cual se realizó un estudio con plantas de morera provenientes del banco de semilla de la EEPF "Indio Hatue". Se midió el largo y el ancho de las hojas que se formaron hasta los ocho meses de edad de la plantación. Para la determinación del área, la silueta de cada hoja se calcó sobre un papel, sin causar daños ni destrucción de tejido. Las mediciones se realizaron a través de un planímetro óptico. La información obtenida fue procesada estadísticamente a través de análisis de regresiones simples y múltiples mediante el programa estadístico SPSS versión 10.0.

Se obtuvieron coeficientes de correlación altamente significativos para casi todos los ajustes efectuados a partir del largo y el ancho de las hojas. A través de la ecuación 2.2 [7]:

$$AF = 0.003 + 0.0073X \tag{2.2}$$

Donde: X= largo por ancho de las hojas.

2.4. Mediciones lineales en la hoja para la estimación no destructiva del área foliar en albahaca (*Ocimun basilicum L*)

Se obtuvieron modelos estadísticos para estimar y predecir el área foliar AF basado en el largo L y ancho A de la hoja de albahaca bajo condiciones de clima desértico, régimen de riego y manejo orgánico en la porción meridional de la península de Baja California, México. Se tomaron 500 muestras $n = 500$ aleatorizadas de hojas de tres edades (jóvenes, intermedias y maduras), en una superficie de una hectárea. Las mediciones obtenidas se correlacionaron para la generación de ecuaciones de regresión lineales simples (área foliar en función de largo y de ancho) y lineal múltiple (área foliar en función de largo por ancho). El área foliar resultó correlacionada positivamente con largo $r = 0.89$ y con ancho de la hoja $r = 0.86$ y con ancho por largo $r = 0.97$. Todas las ecuaciones calculadas explicaron significativamente $P < 0.0001$ el AF (amplitud de variación entre modelos de $R^2 = 0.70 - 0.93$). Estos resultados muestran la posibilidad de estimar en forma predictiva el AF de manera confiable a partir de medidas fácilmente obtenibles sin destruir la planta, además, valores altos de R^2 no necesariamente indican las bondades del modelo para estos fines. Integralmente, el mejor modelo fue lineal múltiple de largo por ancho $R^2 = 0.93$ [16].

Capítulo 3

Marco conceptual

3.1. Procesamiento Digital de Imágenes

El procesamiento digital de imágenes es aquel conjunto de técnicas matemáticas para el diseño y formulación de los operadores necesarios en una o varias tareas específicas de realce de patrones, es decir que el procesamiento de imágenes tiene como finalidad última el realce de una o varias clases de patrones, ya sean espectrales o espaciales [4].

3.2. Binarización

La binarización de una imagen [5] consiste en comparar los niveles de gris presentes en la imagen con un valor (umbral) predeterminado. Si el nivel de gris de la imagen es menos que el umbral, al pixel de la imagen binarizada se le asigna el valor 0 (negro), y si es mayor, se le asigna un 1 (blanco); de esta forma se obtiene una imagen en blanco y negro. Para lograr que la binarización de la imagen sea la adecuada se necesita calcular un umbral diferente en cada imagen procesada, debido a que la resolución o calidad de una cámara fotográfica no es la misma en todas.

En la Figura 3.1-a se puede ver la imagen original, mientras en la Figura 3.1-b se puede ver la imagen binarizada de acuerdo al umbral calculado por el método de umbralización (dos picos).

Figura 3.1: (a) Hoja original; (b) Hoja binarizada.

3.3. Histograma

Una manera fácil de encontrar un umbral de este tipo es mediante el histograma de los niveles de gris de la imagen. Un histograma en este contexto es un vector que tiene el mismo número de dimensiones que la imagen en niveles de gris. El valor asignado a cada componente en el histograma es el número de píxeles con el i nivel de gris, la suma de todos los componentes en el histograma es igual al número de pixeles. En la Figura 3.2 se muestra el histograma de la hoja de jitomate en escala a grises.

3.4. Umbralización

El umbral es un método utilizado para diferenciar un objeto del fondo en una imagen, donde todos los pixeles con valores de gris igual o por encima de un umbral T se consideran como pixeles ”objeto”, todos los pixeles con valores de gris por debajo del umbral son considerados como ”fondo” [18].

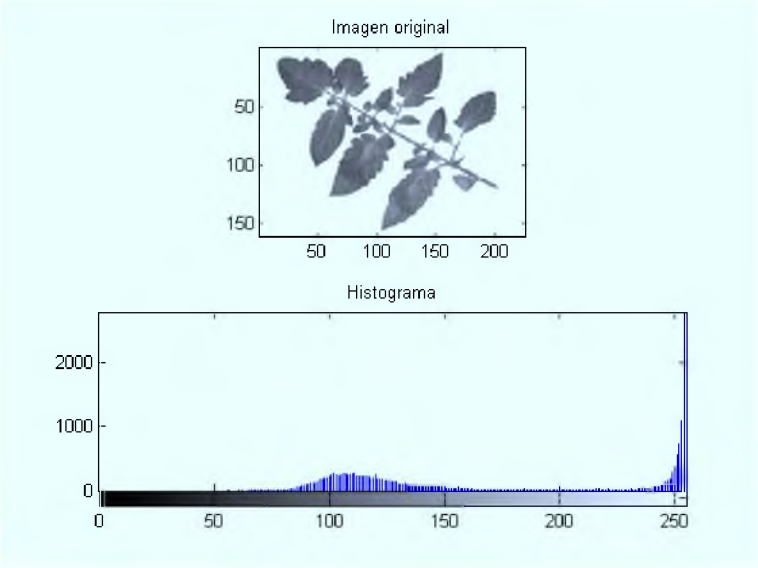


Figura 3.2: Histograma de la hoja de jitomate.

3.4.1. Método de Umbralización (Dos Picos)

La selección de un umbral consta de dos pasos: la localización de los dos picos, y encontrar el punto más bajo entre ellos [14]. Encontrar el primer pico en el histograma es simple: se trata de encontrar el valor más grande. Sin embargo, el valor del segundo mayor es probable que se encuentre al lado del más grande. Una forma simple que con frecuencia funciona bastante bien es buscar el segundo pico multiplicando los valores del histograma por el cuadrado de la distancia desde el primer pico. Esto da preferencia a los picos que no están cerca del máximo. Por lo tanto, si el pico más grande está en el nivel j en el histograma, selecciona el segundo pico como lo muestra la ecuación 3.1:

$$Max = (k-j)^2 h(k) | (0 <= k <= 255) \tag{3.1}$$

3.5. Teorema de Pick

El teorema establece que dado un polígono relaciona su área A con el número de puntos en la red L en su interior y el número B en su límite [8], expresado mediante la ecuación 3.2:

$$A = L + B/2 - 1. \tag{3.2}$$

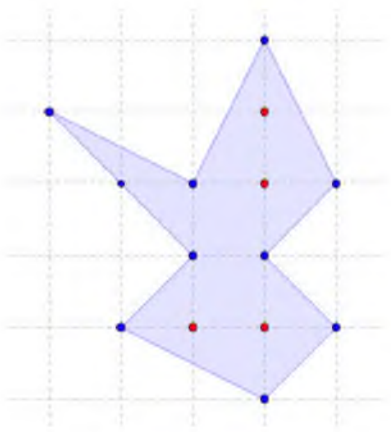


Figura 3.3: Polígono para obtención del área. Tomada de [8].

3.6. Transformaciones Lineales

Las transformaciones geométricas modifican la relación espacial entre pixeles. En términos del procesamiento de imágenes digitales una transformación geométrica consiste de dos operaciones básicas:

1. Una transformación espacial que define la reubicación de los pixeles en el plano imagen.
2. Interpolación de los niveles de grises, los cuales tienen que ver con la asignación de los valores de intensidad de los pixeles en la imagen transformada.

En términos matemáticos las transformaciones afines son las más usadas en imágenes digitales 2D por su representación y manejo matricial. Una transformación afín es aquella en la que las coordenadas x', y' del punto imagen son expresadas linealmente en términos de las del punto original x, y. (ecuación 3.3) [12]:

$$x' = ax + by + my' = cx + dy + n \tag{3.3}$$

Las transformaciones afines tal como la de rotar una región geométrica en un ángulo θ alrededor de un punto fijo, se define como lo muestra la ecuación 3.4:

$$T\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} cos(\theta) & -sen(\theta) \\ sen(\theta) & cos(\theta) \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} \tag{3.4}$$

3.7. Operaciones morfológicas

El procesamiento morfológico de imágenes constituye una clase importante de transformaciones en el cual la forma y estructura de los objetos (patrones espaciales) es modificada para dilucidar su naturaleza morfológica [9]. Se dice que un objeto está mínimamente conectado si la eliminación de cualquiera de los pixeles que lo componen resulta en una pérdida de conectividad para los pixeles restantes. Además se dice que un pixel es crítico si al eliminarlo se pierde conectividad del objeto correspondiente.

Existen dos tipos de transformaciones básicas en la morfología matemática que son la: dilatación y la erosión. En la primera de estas, un objeto crece uniformemente en su extensión espacial, mientras que en la segunda el objeto reduce su extensión espacial uniformemente [4].

3.7.1. Dilatación

Se considera un templete de tamaño impar, el cual recorre todas las posiciones definidas por los pixeles de la imagen; si el patrón binario del templete coincide con el estado (0 o 1) de los pixeles de la imagen cubiertos por éste, entonces en la imagen de salida el pixel en correspondencia espacial con el centro del pixel del templete se pone en su estado binario predeterminado [4].

3.7.2. Erosión

Se considera un templete de tamaño impar, el cual recorre todas las posiciones definidas por los pixeles de la imagen; si el templete no coincide con el estado de los pixeles de la imagen, el pixel de salida se fija en el estado binario opuesto [4].

3.8. Código de Cadena

Los códigos de cadena se utilizan para la representación de formas a partir del cambio en la dirección de los pixeles que forman cada objeto. Cada cambio de dirección en los pixeles corresponde a un elemento del código. Al final del recorrido de todo el objeto se tiene una secuencia de palabras del código que conforman la cadena asociados a números reales. El primer enfoque para la representación de curvas digitales utilizando código de cadena se introdujo por Freeman en 1961, y establece que "en un esquema general para la codificación estructuras de la línea deben cumplir tres objetivos " [6]:

1. Debe fielmente preservar la información de su interés.
2. Permite almacenamiento compacto y conveniente para la exhibición.
3. Facilita cualquier proceso requerido.

Los códigos de cadena [17] se pueden representar mediante los siguientes métodos: tortuga de Papert o PRT, F4 (Freeman cadena de cuatro direcciones), F8 (cadena de Freeman ocho direcciones), VCC2 (código de cadena vértice de dos símbolos), VCC3 (Código de vértice cadena de tres símbolos), 3OT (ortogonal direcciones cadena de tres símbolos), y AF8 (ángulo Freeman código de cadena de ocho direcciones). Estos métodos se pueden aplicar a diferentes objetos o formas irregulares, estós pueden ser a: forma de animales, letras, humanos, plantas, etc.

3.8.1. Código de Cadena Freeman.

En el enfoque de Freeman, una curva arbitraria está representada por una secuencia de vectores de unidad de longitud y un conjunto de ocho direcciones posibles. En la cuadrícula digital, la codificación se basa en el hecho de que los puntos sucesivos en el contorno son adyacentes entre sí.

Si se utiliza la conectividad 4 o conectividad 8, el código de cadena se define como los dígitos de 1 al 4 (figura (3.4)-a) o 1 al 8 (figura (3.4)-b), asignado en el sentido de las agujas del reloj.

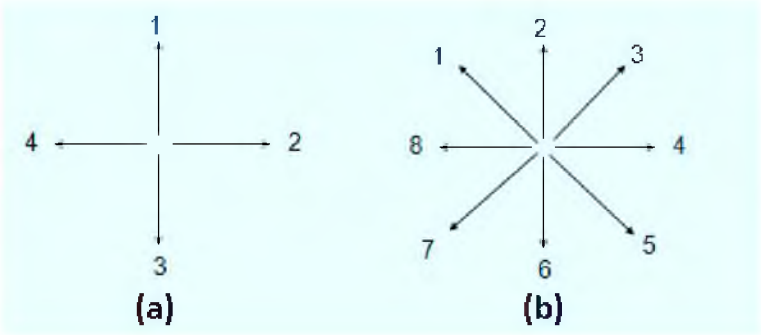


Figura 3.4: (a) Direcciones para vecindad de cuatro; (b) Direcciones para vecindad de ocho.

3.8.2. Código de Cadena Vértice

En 1999, Bribiesca [17] propuso un código de cadena basado en el número de vértices de células que están en contacto con el contorno de la figura. Con el fin de comprimir la información de la cadena de VCC, cada elemento (vértice de células) de la cadena se redujo por uno. Los códigos pueden consistir en dos conjuntos formados por dos o tres símbolos, dependiendo si la resolución de la celda es un cuadrado o un hexágono. Si la resolución de la celda es un cuadrado (Figura 3.5-a) su conjunto de símbolos es VCC3 = 0, 1, 2, donde cada símbolo representa el número de vértices celda de resolución que están en contacto a lo largo de la forma del contorno (Figura 3.5-b).

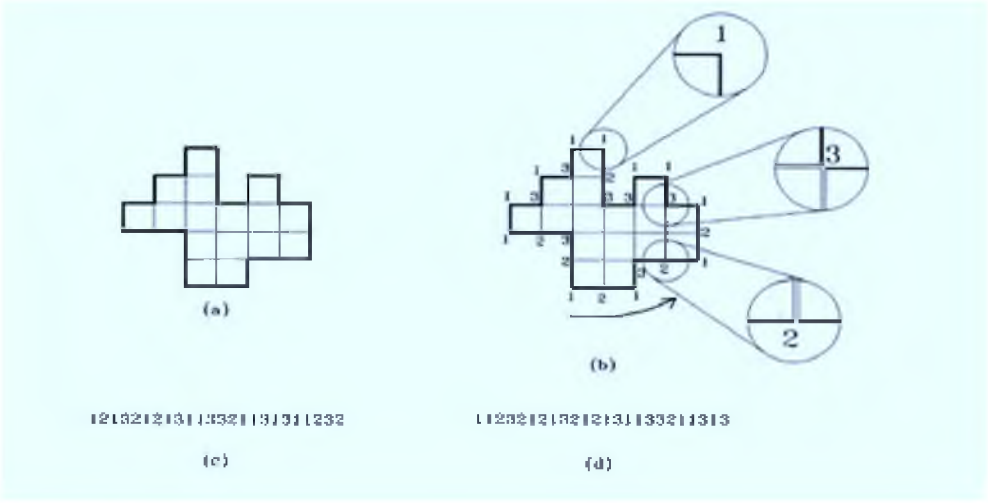


Figura 3.5: El código de cadena vértice: (a) una forma compuesta de píxeles; (b) los elementos de la VCC; (c) la cadena en sentido antihorario de la forma se muestra en (a); (d) la cadena de la izquierda es invariante bajo punto de partida. Tomada de [17].

El método utilizado para la obtención de la cadena es el VCC (Código de cadena de Vértices) [3]; algunas características importantes son:

1. Es invariante bajo traslación, rotación, y opcionalmente puede ser invariante bajo punto de partida y de reflejo transformación como se muestra con los códigos en las Figuras 3.5 c y d.

2. Es posible representar formas compuestas de células triangulares (Figura 3.6-a), rectangulares (Figura 3.6-b), y hexagonales (Figura 3.6-c).

3. Es posible obtener las relaciones entre el contorno delimitador y el interior de la forma.

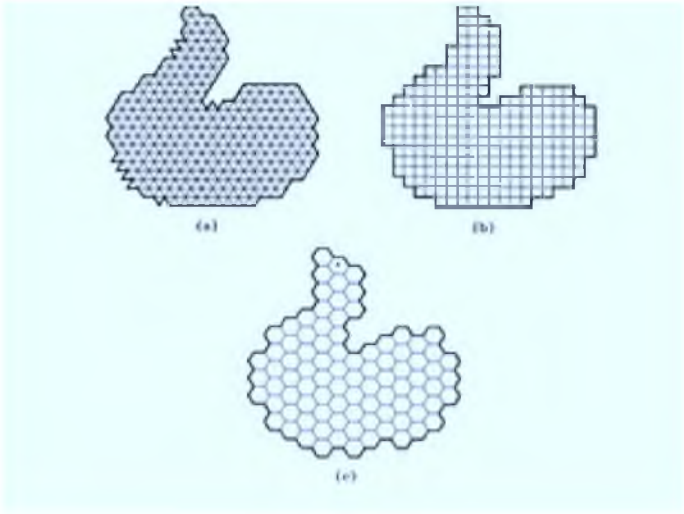


Figura 3.6: Ejemplos de formas discretas compuestas de diferentes formas de células: (a) una forma compuesta de células triangulares; (b) una forma compuesta de células rectangulares; (c) una forma compuesta de células hexagonales. Tomada de [17].

3.9. Alineamiento de Secuencias

Un alineamiento de secuencias [11] es una forma de representar y comparar dos o más secuencias o cadenas, y resaltar sus zonas de similitud. El alineamiento de secuencias normalmente se representa escribiendo las 2 secuencias que queremos comparar una encima de la otra. Si en una posición nos encontramos la misma letra para ambas secuencias, significa que esa posición se ha conservado durante la evolución. En cambio, si las letras no coinciden o nos encontramos un gap (hueco), esto se interpreta como una mutación puntual. Esto lo podemos apreciar en la Figura 3.7 donde se muestran dos secuencias alineadas.

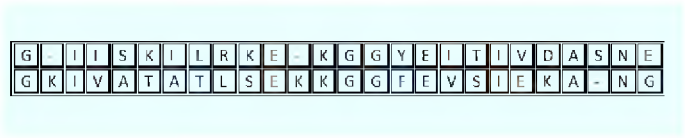


Figura 3.7: Ejemplo de alineamiento de dos secuencias.

Existen dos formas de alinear secuencias: alineamiento local es cuando se comparan las dos secuencias y ninguna de las dos secuencias no se parecen; alineamiento global es cuando las dos secuencias se parecen y cuentan con la misma longitud.

3.9.1. Algoritmo de Needleman - Wunsch.

Es un método de programación dinámica fue propuesto por primera vez en 1970, por Saúl Needleman y Christian Wunsch [13], con la finalidad de obtener el alineamiento global de dos secuencias. El algoritmo se divide en tres partes:

1. **Inicialización de la matriz de puntuaciones.**

   Generar la matriz M (Figura 3.8) en la cual se colocan todas las posibles combinaciones de las dos secuencias $s1$ y $s2$, añadiéndole una fila y columna más de ceros para la generación de la recursividad. Definimos $n = \left|s1\right|$ y $m = \left|s2\right|$ entonces la matriz M será de $m+1*(n+1)$. Por lo que en la posición i se encuentra el $i-1$ y en la posición j el $j-1$.

		G	A	A	T	T	C	A	G	T	T	A
	0	0	0	0	0	0	0	0	0	0	0	0
G	0											
G	0											
A	0											
T	0											
C	0											
G	0											
A	0											

Figura 3.8: Matriz Mde $n * m$.

2. **Cálculo de la matriz de puntuaciones.**

   El cálculo de la matriz de puntaciones, consiste en realizar el cálculo de cada elemento de la matriz aplicando la ecuación (3.5):

$$M_{i,j} = max(M_{i-1,j-1} + S_{i,j}; M_{i,j-1} + W; M_{i-1,j} + W). \tag{3.5}$$

   Donde:

   $M_{i-1,j-1} + S_{i,j}$: Indica si existe o no coincidencia de los caracteres de la secuencia, si los caracteres coinciden $S_{i,j}$ será un valor positivo, sino será un valor negativo.

   $M_{i,j-1} + W$: Indica la suma en horizontal más la penalización por gap (W).

   $M_{i-1,j} + W$: Indica la suma vertical más la penalización por gap (W).

   En la Figura 3.9 se muestran las posiciones de cada uno de los elementos para ir rellenando la matriz $M(i, j)$ aplicando la ecuación (3.5), en la Figura 3.10 se muestra la matriz de puntuaciones terminada.

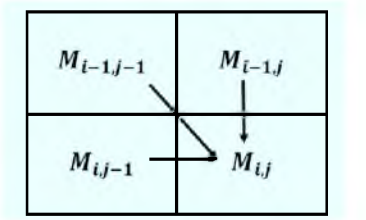


Figura 3.9: Dependencia de datos para el cálculo del elemento $M_{i,j}$.

		G	A	A	T	T	C	A	G	T	T	A
	0	0	0	0	0	0	0	0	0	0	0	0
G	0	1	1	1	1	1	1	1	1	1	1	1
G	0	1	1	1	1	1	1	1	2	2	2	2
A	0	1	1	2	2	2	2	2	2	2	2	3
T	0	1	2	3	3	3	3	3	3	3	3	3
C	0	1	2	3	3	3	4	4	4	4	4	4
G	0	1	2	3	3	3	4	4	5	5	5	5
A	0	1	4	3	3	3	4	5	5	5	5	3

Figura 3.10: Matriz de puntaciones terminada.

3. **Realización del traceback (rastreo) y generación del alineamiento.**

   Este procedimiento se realiza de forma inversa, es decir, se comienza en la parte inferior derecha y concluye en la superior izquierda de la matriz M. Para avanzar se toma en cuenta las condiciones antes mencionadas en la ecuación, se avanza hacia arriba o hacia la diagonal izquierda o diagonal derecha dependiendo cual condición genera el valor en $M_{i,j}$. En el alineamiento de cadenas es necesario tomar en cuenta las siguientes reglas para trazar un camino, como se muestra en la Figura 3.11 de color amarillo.

   - Si el avance es diagonal, alineamos:

     $a1 = a1 + s1_{i-1}$ y $a2 = a2 + s1_{i-1}$

   - Si el avance es horizontal, alineamos :

     $a1 = a1 + s1_{i-1}$ y $a2 = a2 + " - "$

   - Si el avance es vertical, alineamos:

     $a1 = a1 + " - "$ y $a2 = a2 + s1_{i-1}$

		G	A	A	T	T	C	A	G	T	T	A
	0	0	0	0	0	0	0	0	0	0	0	0
G	0	1	1	1	1	1	1	1	1	1	1	1
G	0	1	1	1	1	1	1	1	2	2	2	2
A	0	1	1	2	2	2	2	2	2	2	2	3
T	0	1	2	3	3	3	3	3	3	3	3	3
C	0	1	2	3	3	3	4	4	4	4	4	4
G	0	1	2	3	3	3	4	4	5	5	5	5
A	0	1	4	3	3	3	4	5	5	5	5	3

Figura 3.11: Generación del alineamiento.

3.10. Programación concurrente

Java permite la creación de programas con múltiples hilos de ejecución (multithreading); es decir, programas con varias partes denomidas hilos, que realizan sus actividades en paralelo. La creacion de varios programas con varios hilos de ejecución resulta muy útil en algunas ocasiones; por ejemplo, para imprimir un documento largo mientras el usuario continúa con el uso de la IGU, para consegir una eficiente implementación de las animaciones.

Los hilos de ejecución en java están implementados en la clase "Thread", que forma el paquete de "java.lang". Cada hilo (*thread*) tiene un principio, un flujo de ejecución y un fin definidos, pero no es una entidad independiente sino que debe ejecutarse en el contexto de un programa. La clase Thread implementa el concepto de ejecución multihilo de una forma independiente de la plataforma donde se produce la ejecución [1].

Utilizar programación cocurrente hace que la aplicacion sea más rapida para realizar el procesamiento digital de las imágenes, en esta ocasión se utiliza para reestructurar cada foliolo en un hilo, para que el cálculo del área sea en menor tiempo. El tiempo de ejecución depende de las características del celular donde se esté corriendo la aplicación, entre mejor sea su procesador y su memoria RAM menos tiempo tarda en ejecutarse.

Capítulo 4

Diseño e implementación

4.1. Creación de los modelos

Consiste en crear dos modelos generales de dos foliolos, para la reconstrucción de otros foliolos mediante los códigos de cadena y alineamiento de secuencias. En la creación de los modelos se toma una muestra de 100 foliolos de diferentes hojas, separándolas en dos clases (A,B), a las cuales se aplicaron los siguientes pasos:

1. Binarizar.

   Cada una de las imágenes de los foliolos es binarizada, el umbral para binarización es dinámico, es decir su valor cambia de acuerdo a la imagen. El valor del umbral se calcula mediante el método de dos picos. En la Figura 4.1 se muestran las imágenes de los foliolos binarizadas (blanco y negro).

2. Cambio de orientación.

   Debido a que los foliolos se encuentran en diferente orientación, se procede a colocar manualmente todas las imágenes de manera vertical como se muestra la Figura 4.1, y así obtener un mejor modelo.

Figura 4.1: Imágenes de los foliolos ajustados de manera vertical.

3. Redimensionar.

   De la misma manera cada una de las imágenes es ajustada en una escala de $100 * 150$ pixeles, en este proceso se busca en donde inicia y donde termina el foliolo para ser recortado, así se crea una nueva imagen. Posteriormente esta nueva imagen se redimensiona a la escala mencionada. Este proceso es importante, por que todas las imágenes se encuentran en un solo tamaño y así es más facil crear un modelo adecuado.

4. Erosionar.

   Como se puede ver en la Figura 4.1 los foliolos tienen ruido, esto ocasiona que los códigos de cadena no sean los adecuados. Para quitar el ruido se erosiona la imagen, es decir, se eliminan todos aquellos pixeles en negro, donde alguno de sus vecinos sea de color blanco (en una imagen digital su valor es -1). Por lo cual se toma en cuenta la vecindad de ocho como lo muestra la Figura 4.2-a, en donde las posiciones (x, y) representan al pixel en negro (en una imagen digital su valor es -16777216).

5. Obtención de los códigos de cadena.

   La obtención de los códigos de cadena se realizan en cada una de las imágenes de los 100 foliolos mediante los código de Cadena Freeman (vecindad de ocho), la codificación empieza en el primer pixel que se encuentra en la parte inferior de la imagen, recorriendo todo el borde hasta terminar en el mismo pixel. El recorrido se realiza en el sentido de las manecillas del reloj, se obtiene el código (Figura 4.2-b) del siguiente pixel (vecino que se encuentre en borde).

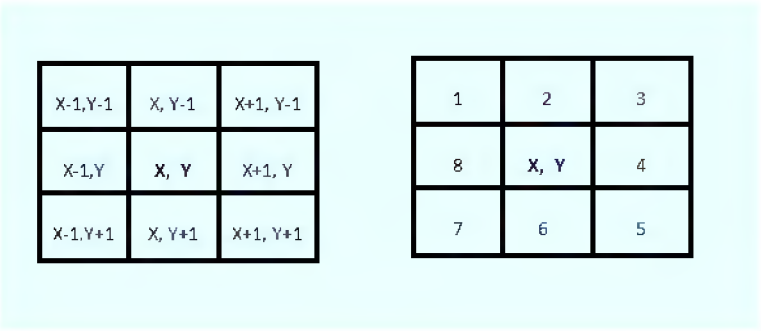


Figura 4.2: Vecindad de ocho: (a) Posición de los vecinos del pixel central (x, y) en la matriz de la imagen; (b) Valores de los vecinos del pixel central (x, y) para el código de cadena.

6. Alineamiento de los códigos de cadena.

   El alineamiento de los códigos se realiza mediante la programación dinámica y el algoritmo de NeedlemanWunsch, primero se alinean los dos primeros códigos en cada clase, el código de cadena resultante se alinea con el siguiente, así sucesivamente hasta terminar con todas las imágenes de cada clase. Como resultado de estas alineaciones obtenemos un modelo para cada clase (A, B).

   Modelos generales:

   Código de cadena de la clase A.

   $MODELO1$="188181881818818818818818818818881818211221111221121118822122
   232212121112133344443332222221211221212222223232223332221222323333223333
   322232322223323333224433322323332232345555545555665655545555566655554
   565566565555665565556666665666565665666666655554554666656767766666767666
   666576766667777777666677777877878777777687778".

Código de cadena de la clase B.

$MODELO2$="188188888188888818881888881888888181122111888122333332222111123221122324444333322221121211222132323221444433332322222221223224433322121232322222122233233332232332233323233322332345555456556555455555655656566555555467666656565655554444466665666666766666655544446687777767676657665554444446666555676666666677677787887878878888787888777676666665687778878878888".

7. Reconstrucción de los modelos.

   La reconstrucción de los modelos se realiza con ayuda de códigos Freeman (vecindad de ocho). Se recorre cada uno de los elementos del modelo, colocando en negro el pixel que corresponde al valor del elemento del modelo. En la Figura 4.2-b se observa el valor, mientras en la Figura 4.2-a se observa el pixel correpondiente a dicho valor.

   En la Figura 4.3-a,c se muestran los dos modelos reconstruidos, en el caso que no se encuentren totalmente cerrados, se traza una línea recta desde el punto inicial hasta el final. La recta es trazada mediante el método DDA (Analizador Diferencial Digital), este mismo método es utilizado para llenar los modelos como se aprecia en la Figura 4.3-b,d.

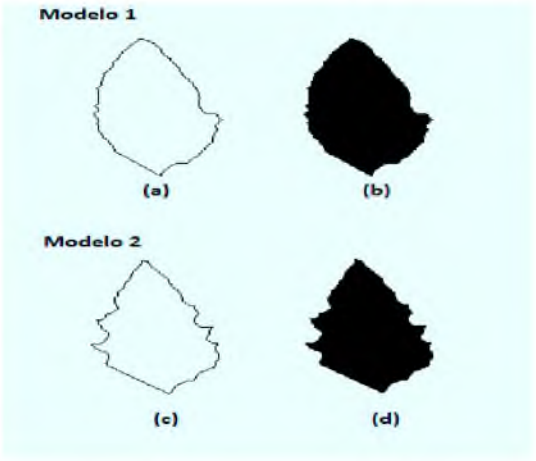


Figura 4.3: (a,c) Reconstrucción de los modelos; (b,d) Relleno de los modelos.

4.2. Binarización y tranformación lineal de la imagen

El primer paso es la captura de la hoja de jitomate en la cual es obtenida el área foliar, la fotografía es capturada con la cámara del celular donde se está ejecutando la aplicación. Al tomar la fotografía es necesario colocar la hoja de papel en blanco con la referencia detrás de la hoja del jitomate. La referencia es un cuadro de color rojo con una medida de $3cm^2$, la cual se utiliza para el cálculo del área.

No todas las cámaras fotográficas cuentan con la misma resolución, por lo que es necesario obtener un umbral difierente en cada captura, y así lograr una mejor binarización de la imagen. El umbral es calculado mediante el método de dos picos, el cual se elaboró en el lenguaje Java bajo la plataforma de Android; la función realizada se describe a continuación:

4.2.1. Pseudocódigos de la Binarización y el método de dos picos

```
1: calculoDosPicos(entrada)
2: hk=0
3: k=0
4: for (i=0: entrada.longitud) do
5:     if (entrada(i) es menor que k)  then
6:         k=i
7:         hk=entrada(i)
8:     end if
9: end for
10: hi=0
11: j=0
12: for (i=0: entrada.longitud) do
13:     aux=(i-k)^2 * entrada(i)
```

```
14:     if (aux es mayor que hi) then
15:         j=i
16:         hi=aux
17:     end if
18: end for
19: menor=k
20: mayor=j
21: if (k es mayor que j) then
22:     mayor = k
23:     menor = j
24: end if
25: t=j
26: ht=hk
27: for (i = menor+1 : menor que mayor-1) do
28:     if (entrada(i) es menor que ht) then
29:         t=i
30:         ht=entrada(i)
31:     end if
32: end for
33: retorna t
```

A continuación se describe la función que binariza la imagen de acuerdo con el umbral obtenido por el método de dos picos.

```
1: binarizarImagen(umbral)
2: pixel2=0
3: promedio=0
4: bmp2(w,h)
5: for (i = 0: i menor a w) do
6:     for (j = 0: i menor a h) do
7:         pixel2 = valor del bmp(i, j)
8:         red = valor rojo de pixel2
9:         blue= valor azul de pixel2
10:        green = valor verde de pixel2
11:        promedio = promedio del rojo, verde y azul
12:        if ( promedio es menor que umbral) then
13:            bmp2(i, j) se convierte en black
14:        else
15:            bmp2(i, j) se convierte en white
16:        end if
17:    end for
18: end for
```

Después de tener la imagen binarizada se realiza una tranformación lineal, es decir, se rota en un ángulo de 0° a 30°, con la finalidad de encontrar la mayor área foliar posible y tener una mejor vista de la hoja de jitomate. Antes de empezar la rotación es necesario calcular primero el área foliar, en el momento de incrementar el ángulo de rotación de la imagen se calcula de nuevo el área, si aumenta el área se sigue incrementando el ángulo hasta los 30° y el área obtenida es la final. En caso de que disminuya, el área final es la obtenida en la posición anterior.

4.3. Separación y reconstrucción de los foliolos

En la Figura 4.4 se observan dos ejemplos donde dos foliolos se encuentran uno encima de otro, esto impide saber con exactitud cuál es el área foliar de la imagen, por lo cual se tienen que separar en cada ejemplo los dos foliolos y reconstruirlos utilizando alguno de los dos modelos generados y así obtener mejor el área foliar.

En la reconstrucción de los foliolos es necesario ajustar la imagen en cada ejemplo "F1 y F2"; es decir, se realizan algunos de los procedimientos realizados para generar los modelos (binarizar, cambio de orientación, redimensionar, erosionar), en la Figura 4.5 se muestran dos ejemplos de los foliolos erosionados. Para separacion de los foliolos se realizan los siguientes pasos:

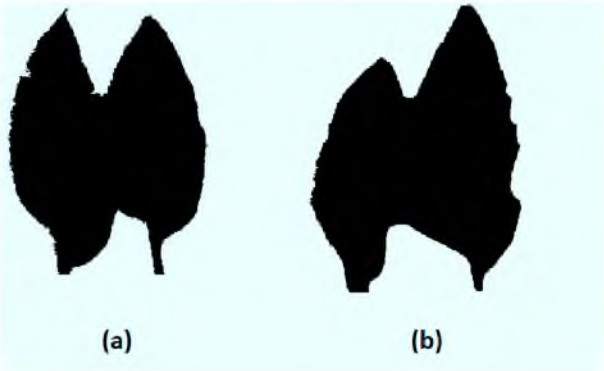


Figura 4.4: (a) Dos foliolos que se encuentran uno sobre otro "F1" ; (b) Dos foliolos que se encuentran uno sobre otro "F2".

1. Ubicación de los puntos de intersección. Como se puede observar en los dos ejemplos de la Figura 4.5 los foliolos se intersectan en dos puntos (superior e inferior), estos puntos se localizan de la siguiente forma. La localización del punto superior, el recorrido comienza en el primer pixel del borde derecho del primer foliolo, el objetivo es encontrar el borde izquierdo del segundo foliolo (otro pixel en negro), si se encuentra a una distancia mayor de 20 pixeles sus coordenadas son guardadas y continúa con el siguiente pixel en borde del primer foliolo.

Las coordenadas guardadas son: $x1, y1$ que corresponde al pixel del borde en primer foliolo, $x3, y3$ corresponde al pixel del borde en segundo foliolo. En caso de que la distancia sea menor las coordenadas de intercección en la parte superior son las últimas almacenadas.

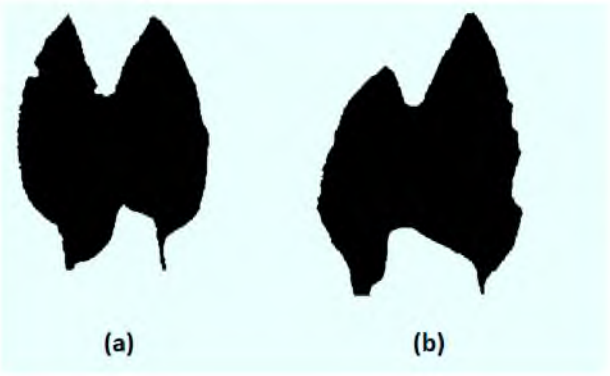


Figura 4.5: (a) Erosión de los foliolos que se encuentran uno sobre otro "F1" ; (b) Erosión de los foliolos que se encuentran uno sobre otro "F2".

La localización del punto inferior es igual que la anterior solo que en este caso el recorrido comienza a partir del último pixel en el borde derecho del primer foliolo hasta encontrar las coordenadas de la intercepción. Las coordenadas guardadas son: $x2, y2$ que corresponde al pixel del borde en primer foliolo, $x4, y4$ corresponde al pixel del borde en segundo foliolo. En caso de que la distancia sea menor las coordenadas de intercepción en la parte superior son las últimas almacenadas.

2. Obtención de los código de cadena.

   En cada ejemplo se localiza primero el primer pixel $incialX, inicialY$ que se encuentra en la última fila en el primer foliolo, la codificación de la primera parte comienza en este pixel recorriendo el borde hasta el pixel que se encuentre en la posición $x1, y1$. La segunda parte comienza del pixel que se encuentre en la posición $x2, y2$ recorriendo el borde hasta el pixel que encuentre la posición $incialX, inicialY$; las partes son separadas por un "-".

A continuación se muestra el código de cadena obtenido en el primer foliolo en cada ejemplo.

Código de cadena del primer foliolo en la Figura 4.5-a (F1):

$Cadena1$=

"888887887876788888-2212222221122213221181212118818181818111818181111
8122118211222122222118243112222221133122231133222323124318322433221232
3222333444454433313182181243324324332332243433344343243343243334465655
5654666545656565666556655554665656546546668 76".

Código de cadena del primer foliolo en la Figura 4.5-c (F2):

$Cadena1$=

"27877767666666666766676666767767788787887777775688888888888888-11222
1212222212121182218111111818218182221812111218182212112111322312243243
2222183323223212223433223223334324322322223233332233223323432432343334
3434333434334343344434343343344545454455565556655465666556682 2".

La obtención del código de cadena del segundo folio, comienza por encontrar la fila que tiene el primer pixel $incialX, inicialY$ en negro, la búsqueda comienza en la columna (x1). La codificación de la primera parte del código de cadena comienza en el pixel $x3, y3$ recorriendo el borde hasta el pixel que se encuentre en la posición $inicialX, inicialY$. La segunda parte comienza apartir del pixel $inicialX, inicialY$ recorriendo el borde hasta el pixel que encuentre la posición $x4, y4$. Las partes son separadas por un "-", a continuación se muestran el código de cadena obtenido el segundo foliolo en cada ejemplo.

Código de cadena del segundo foliolo en la Figura 4.5-b (F1):

$Cadena2$=

"22212222122212222112223118222181888118881888-33223324332223224323333244322243343433224343323433322434444544545555445546565546554565555655656545665546555566565665665666566566555555566666666766667576666666677666666767676677666777877787887878878878878878778776666666665755666888".

Código de cadena del segundo foliolo en la Figura 4.5-d (F2):

$Cadena2$=

"22211222122211111818188181888818-33232333343223332324323324332434323332432432333233333444444455554556555556555654556566556665655566556766655454666667666656545566666566546667767677666776666676575555545457766676776677667777877787876778788877777776666688".

La reconstrucción de cada cadena de los dos foliolos en los dos ejemplos se observa en la Figura 4.6, donde además se observa que el borde de cada foliolo no se encuentra cerrado, por lo que el siguiente paso es cerrar cada borde para obtener el área. Este proceso se lleva a cabo mediante el alineamiento de cadena entre el código del foliolo y los códigos de cadena de los dos modelos, obteniendo de alguno de los dos modelos la parte de código que se necesita para cerrar el foliolo.

3. Alineamiento, reconstrucción y relleno de los foliolos.

   En los códigos de cadena de los foliolos se muestra con "-" la parte de código que falta para cerrarlo, por lo cual se busca en los códigos de los modelos cual es la parte que mejor se ajusta. Este procedimiento es tardado por lo cual es necesario ocupar la programación concurrente, donde para la reconstrucción del primer foliolo se realiza en un hilo y los del segundo se realiza en otro hilo.

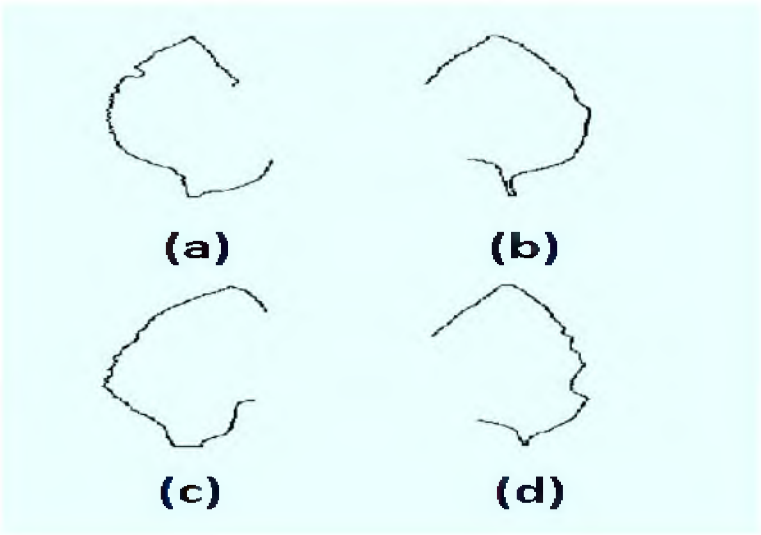


Figura 4.6: Código regenerado del Ejemplo 1 (a) primer foliolo, (b) segundo foliolo; Código regenerado del Ejemplo 2 (a) primer foliolo, (b) segundo foliolo.

Pasos realizados en el primer foliolo (figura 4.6-a,c):

a) Crear una sub-cadena de cada uno de los modelos:

La sub-cadena en el primer foliolo comienza en la posición de la longitud de la segunda parte del código del foliolo, termina en la posición que resultante entre la resta de la longitud del modelo y la longitud de la primera parte del código, como se muestra a continuación :

$$SubCadena1 = MODELO1.subcadena(Cadena1(parte2).longitud - n,$$
$$MODELO1.longitud - Cadena1(parte1).longitud - n)$$
$$SubCadena2 = MODELO2.subcadena(Cadena1(parte2).longitud - n,$$
$$MODELO2.longitud - Cadena1(parte1).longitud - n)$$

El valor de ”n” va de cero hasta treinta, se repite treinta veces hasta encontrar cual es la mejor sub-cadena.

b) Se sustituye en la $cadena1$ el "-" por la $subCadena1$ obtenida del modelo 1 y la $SubCadena2$ obtenida en $modelo2$ formando una cadena nueva en cada sustitución.

$$CadenaN1 = Cadena1(parte2) + "" + Subcadena1 + "" + Cadena1(parte1)$$

$$CadenaN2 = Cadena1(parte2) + "" + Subcadena2 + "" + Cadena1(parte1)$$

c) Alinear las dos nuevas cadenas $CadenaN1, CadenaN2$ con los modelos $MODELO1$, $MODELO2$, se alinea la $CadenaN1$ con $MODELO1$, el resultado es guardado en $alineacion1$ y la $CadenaN2$ con $MODELO2$, el resultado guardado en $alineacion2$.

d) Buscar el número de coincidencias entre $CadenaN1, alineacion1$, el resultado es guardado en $m1$ y $CadenaN2yalineacion2$, el resultado es guardado en $m2$, quien tiene el mayor número de coincidencias es nuestro modelo final.

Si el $m1$ es mayor que $m2$, el modelo general es de la siguiente manera:

$$modelo\_general = CadenaN1(parte1) + "" + CadenaN1(parte2) + "" + SubCadena1$$

En caso de que $m2$ sea mayor, el modelo general es de la siguiente manera:

$$modelo\_general = CadenaN1(parte1) + "" + CadenaN1(parte2) + "" + SubCadena2$$

e) Se reconstruye el código del modelo general con los códigos de cadena de Freeman (Figuras 4.7-a, 4.9a), si el foliolo aún no está cerrado se traza una línea recta entre los dos puntos (Figuras 4.7-b, 4.7b). Por último se rellena el esqueleto del foliolo (Figuras 4.7-c, 4.7c).

Pasos realizados en el segundo foliolo (figura 4.6-b,d):

a) Crear una sub-cadena de cada uno de los modelos. La sub-cadena en el segundo foliolo comienza en la posición de la longitud de la primera parte del código del foliolo, termina en la posición que da cómo resultado la resta entre la longitud del modelo y la longitud de la segunda parte del código. Como a continuación se muestra:

$SubCadena1 = MODELO1.subcadena(Cadena2(parte1).longitud - n,$
$MODELO1.longitud - Cadena2(parte2).longitud - n)$
$SubCadena2 = MODELO2.subcadena(Cadena2(parte1).longitud - n,$
$MODELO2.longitud - Cadena2(parte2).longitud - n)$

El valor de "n" va de cero hasta treinta, se repite treinta veces hasta encontrar cual es la mejor sub-cadena.

b) Se sustituye en la $cadena1$ el «-» por la $SubCadena1$ obtenida del modelo 1 y la $SubCadena2$ obtenida en $modelo2$ formando una cadena nueva en cada sustitución.

$CadenaN1 = Cadena2(parte1) + "" + Subcadena1 + "" + Cadena2(parte2)$
$CadenaN2 = Cadena2(parte1) + "" + Subcadena2 + "" + Cadena2(parte2)$

c) Alinear las dos nuevas cadenas $CadenaN1, CadenaN2$ con los modelos $MODELO1$, $MODELO2$, se alinea la $CadenaN1$ con $MODELO1$, el resultado es guardado en $alineacion1$ y la $CadenaN2$ con $MODELO2$, el resultado es guardado en $alineacion2$.

d) Buscar el número de coincidencias entre $CadenaN1, alineacion1$, el resultado es guardado en $m1$ y $CadenaN2$ y $alineacion2$, el resultado es guardado en $m2$, quien tiene el mayor número de coincidencias es nuestro modelo final.
Si el $m1$ es mayor que $m2$, el modelo general es de la siguiente manera:

$$modelo\_general = CadenaN1(parte2) + "" + CadenaN1(parte1) + "" + SubCadena1$$

En caso de que $m2$ sea mayor, el modelo general es de la siguiente manera:

$$modelo\_general = cadenaN2(parte2) + "" + cadenaN2(parte1) + "" + Subcadena2$$

e) Se reconstruye el código del modelo general con los código de cadena de Freeman (Figuras 4.8-a, 4.9-a), si el foliolo aun no está cerrado se traza una línea (Figuras 4.8-b, 4.9-b). Por último se rellena el esqueleto del foliolo (Figuras 4.8-c, 4.9-c).

4.3.1. Resultados aplicados a los ejemplos F1 y F2

Pasos aplicados a la figura 4.6a:

$SubCadena1$="666656665656656666666555545546666567677666667676666657676666677

$SubCadena2$="565555444446666566666676666665554444668777776767665766555444444

666655567666666667767778788787878878888787".

Alineamiento de la $CadenaN1$ y el $MODELO1$.

$cadena1$="221222222112221322118121211818818818811118181811118122118211222122

22221182431122222211331222311332222321243183224332212323322233344445443331318

82181243324324332332243433344343243343243334465655565466654565656656655655

55466565654654666876666656665656566666665555455466665676776666676766666576766

666778888887887876788888".

Resultado del alineamiento entre las dos cadenas.

$alineacion1$="122222218122213221181212118188181881811181818211221181221182111

11822212223221218243112222221133122231133222232124318321243322123233222333444

44544323313182331812432324324332333322434333443432433432433344656554565466565

54565656566655665545654665655654654666876666656665656566666665555455466665676776

76776666676766666576766667777777666678888887878767688888".

Alineamiento de la $CadenaN2$ y el $MODELO2$.

$CadenaN2$="22122222211222132211812121188181881811181818111181221182112221
22222118243112222221133122231133222321243183224332212323322233344445443331318218124332432433233322434333443432433432433344656555654666545656566566556655546656565465466687656555544444666656666667666666555444466877776767665765544444466665556766666666776777878878878878888787888887887876788888".

Resultado del alineamiento entre las dos cadenas.

$alineacion2$="12221322118121211818818818811118181811118122111821122333212222211822432211223222211331222311332122321324318322144443322123233222333444454433321318231812433243243323333224343322344343243343243323455554565565556546665456565665566556555467656565465466687656555544444666656666667666666555444466877776767665766555444444666655567666666667767778788787887888878878887778767666656888788787678888888".

El número de coincidencias entre la $CadenaN1$ y $alineacion1$ es de 100, $Cadena2$ y $alineacion2$ es de 101, por lo tanto el modelo general en este foliolo es:

$modelo\_general = CadenaN1(parte1) + "" + CadenaN1(parte2) + "" + SubCadena2$

$Modelo\_general$ ="88888878878767888882212222221122213221181212118188181881
81118181811118122118211222122222118243112222221133122231133222321243183224332212323322233344445443331318218124332432433233322434333443432433432433344656555654666545656566566556655546656565465466687656555544444666656666667666666555444466877776767665765544444466665556766666666776777878878788788887887887767888888".

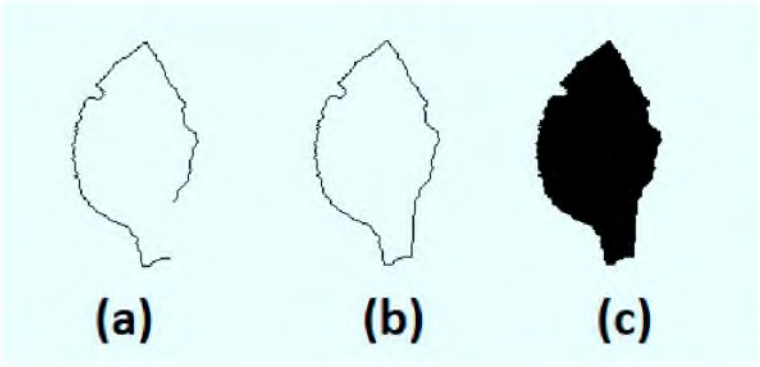


Figura 4.7: (a) Modelo general del primer foliolo; (b) Foliolo cerrado; (c) Foliolo rellenado.

Pasos aplicados a la Figura 4.6b:

$SubCadena1$=”188188181881818881818211221111221121118 8”

$SubCadena2$=”188818888818888881811221118881223333322221111232211223244443333222211212112221323”

Alineamiento de la $CadenaN1$ y el $MODELO1$.

$CadenaN1$=”22212222122212222112223118222181888118881888188188181888181888181821122111221121111883322332433222322432333324432224334343322434332343332243444544545555454554656555465545665555565566544555544554656554655455655565565456655465555665656656566566566556555566666666766667576666666677666666767676677666778777878878778878878878878787787776666666665755666888”.

Resultado del alineamiento entre las dos cadenas.

$alineamiento1$=”18222181881811888188818818818818818881818211221111221121118822122322123211121333442433322222212112212122322432322233322443223234332243322234322243323333224433322324344445445455554545546565554655455665555565565654566554655556565665656656665566555555566666666766667576666666677666667676767657676666777877787887877887888787887787776666666665755666888”.

Alineamiento de la $CadenaN2$ y el $MODELO2$.

$CadenaN2$="22212222122212222112223118222181888118881888188818888818888888181
12211188812233333222211112322112232444433332222112121122213233322332432232232
24323333244322243343433224343323433322434444544545555445546565546554565555655
56565456655465555665656656565666565665665555555666666667666675766666666776666
67676766776667778777878878778878878887878877877666666666575566686

Resultado del alineamiento entre las dos cadenas.

$alineamiento2$=18222181888118881888188818888818888881811221118881223333322
21111232211223244443333222211212112221323332214444333324322222321223224432332
22212323224432122433434332232432243323433322432444454454556554454656554656565
45655556554656545665656465555446656566565666566566655555556686666667666675766
65666666776666667676766776667778777878878778878878887878877877676666665686
5788855666888".

El número de coincidencias entre la $CadenaN1$ y $alineacion1$ es de 93, $Cadena2$ y $alineacion2$ es de 138, por lo tanto el modelo general en este foliolo es:

$modelo\_general = cadenaN2(parte2) + "" + cadenaN2(parte1) + "" + Subcadena2$

$Modelo\_general$=3322332433222322432333324432224334343322434332343332243444
54454555544554656554655455655556556565456655465555665656656566565665665555555
56666666676666757666666666776666667676766776667778777878878778878878878778877
87776666666665755666888222122221222122221122231182221818881188818881881888188
81888888181122111888122333332222111123221122324444333322221121211222132 3".

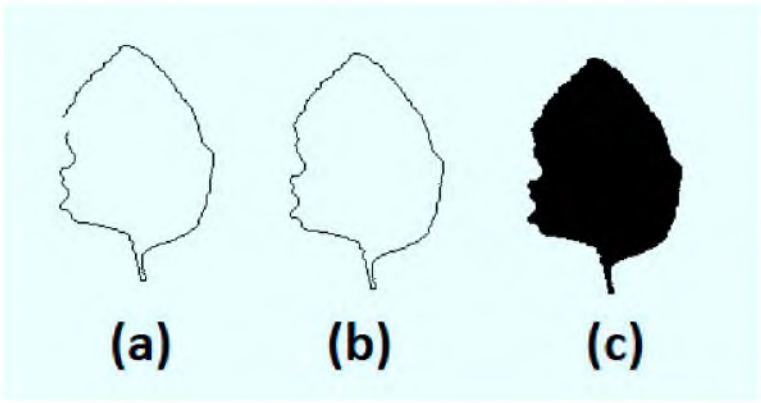


Figura 4.8: (a) Modelo general del primer foliolo; (b) Foliolo cerrado; (c) Foliolo rellenado.

Pasos aplicados a la Figura 4.6c:

$SubCadena1$="555554565566565555665565556666656665656656".

$SubCadena2$="655565656665555554676666565656555544444666656666676666665554
4446687777767676657665 5".

Alineamiento de la $CadenaN1$ y el $MODELO1$.

$CadenaN1$="1122212122222212121182218111111818218182221812111218182212112111
3223122432432222183323223212223433223223334324322322223233332233223323432432
3433343343433343343343344434343343343344545454455565556656546566656682255555
4565566565555665565556666665666565665627877767666666666766676666767767788787
7887777775688888888888888".

Resultado del alineamiento entre las dos cadenas.

$alineacion1$="1222121222222182121182218811111181882181821122118121112181882
2122232212121112132231224324322221833213223212223433223223334324322322223233
3322332233234324323433343343343343343343343344434343343343344545454455565556655
4656665566822555554565566565555665565556666665666565665627877765555455476665
6767766666767667666576767767788787688777775688888888888888".

Alineamiento de la $CadenaN2$ y el $MODELO2$.

$CadenaN2$="11222121222222121211822181111118182181822218121112181822121121113223122432444433332222183323223212221343232214322333432432232222323333
32244333223323432432343334334343332434332434334444343433243432344545445545556555665546566655668226555
55665465666556682265556565666555555467666656565655554444466665666676666665554444668777776767665761
65527877767666666666766676666767767788787887777756888888888888888".

Resultado del alineamiento entre las dos cadenas.

$alineacion2$="18812221212222221212118221811111181821818221218121112318182212111322312243244444333322221833232232122213432322143223334324322322223233
32244333223323432432343334334343332434332434334444343433243432344545445545556555665546566655668226555
55665465666556682265556565666555555467666656565655554444466665666676666665554444668777776767665761
65527877767666666666766676666767767788787887777756888888888888888".

El número de coincidencias entre la $CadenaN1$ y $alineacion1$ es de 119, $Cadena2$ y $alineacion$ es de 138, por lo tanto el modelo general en este foliolo es:

$$modelo\_general = CadenaN1(parte1) + "" + CadenaN1(parte2) + "" + SubCadena2$$

$Modelo\_general$="27877767666666666676667666676776778878788777775688888888888888111222121222222121211822181111118182181822218121112181822121121113223122432444433332222183323223212221343232214322333432432232222323333224433322332343243234333433434333243433243433444343433243432344545445545556555665546566655668226555655665465666556682265556565666555555467666656565655554444466665666667666666555444466877777676766576555554676666565656555544444666656666667666666555444466877777676766576655"

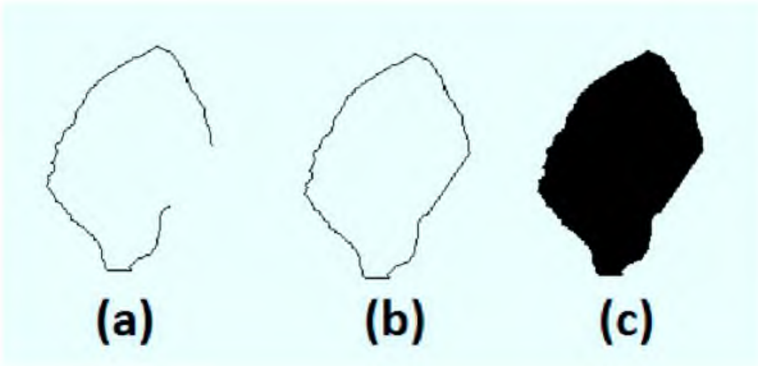


Figura 4.9: (a) Modelo general del primer foliolo; (b) Foliolo cerrado; (c) Foliolo rellenado

Pasos aplicados a la Figura 4.6d:

$SubCadena1$=”18188188188188181881818881818211221111221121118822122232212121”.
$SubCadena2$=”818888881888188888188888818112211188812233333222211112322112232444433332222112121122213232322144443332”.

Alineamiento de la $CadenaN1$ y el $MODELO1$.
$cadena1$=”2221122212221111181818818188881811818818818818818188181888181821122111122112111882212223221212133232333343223332324323324332434323333243243233332333344444455554556555556555654556566556665655566555676665545466666766665654556666656654666776767766677666667657555554545657766676776677666777778777878767787888777777766666688”.

Resultado del alineamiento entre las dos cadenas.
$alineacion1$=”1881818888181181881881881881818818188818182112211112211211188221222322121211121332344423332222221211221212222223432223332324323323433224343222333243243233332244333223233444444555545565555565655654556566556665655546655665655766655645466666765666565455666665666555455466677676776666776766667657555554545657766676776677666677777877787876778788877777776666688”.

Alineamiento de la $CadenaN2$ y el $MODELO2$.

$CadenaN2$="2221122212221111181818818188881818188888818881888881888888181122 1118888122333332222111123221122324444333322221121211222132323221444433333323323 2333343223332324323324332434323332432432333323333444444455554556555556555654 556566556665655566555676665545466666766665654556666656654666776767766677666 6765755555454565776667677667766677777877787876778788877777776666688".

Resultado del alineamiento entre las dos cadenas.

$alineacion2$="1881818888181818888888188818888818888881811221118888122333333222 2111123221122324444333322221121211222132323221444433332323233334322443332 3 2143232223243324343233332432432333323233344444445555455656555465556545565 66 5566655655546766665556766655445466666766665654556666656654444666776767766677 66666765755555454565776665576776677666776777878878788787677878887777777766665 6877788877888888".

El número de coincidencias entre la $CadenaN1$ y $alineacion1$ es de 119, $Cadena2$ y $alineacion$ es de 132, por lo tanto el modelo general en este foliolo es:

$$modelo\_general = CadenaN1(parte1) + "" + CadenaN1(parte2) + "" + SubCadena2$$

$Modelo\_General$="3323233334322333232432332433243432333243243233332333344444 4455554556555556555654556566556665655566555676665545466666766665654556666 65 6654666776767766677666667657555554545657766676776677666777778777878767787888 7777777666668822211222122211111818188181888818181888888188818888818888881811 22111888122333332222111123221122324444333322221121211222132323221444433332".

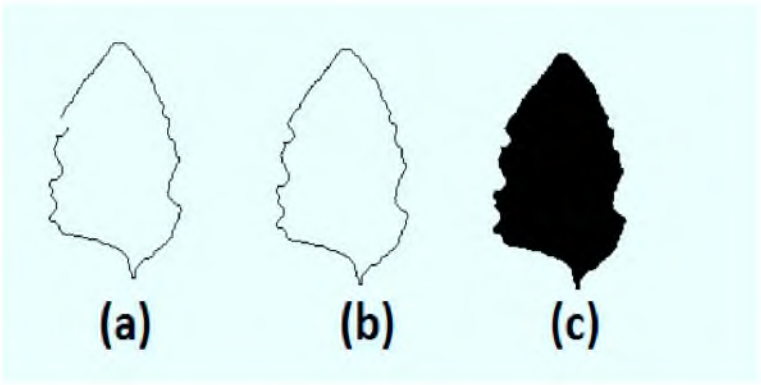


Figura 4.10: (a) Modelo general del primer foliolo; (b) Foliolo cerrado; (c) Foliolo rellenado.

4.3.2. Pseudocódigo del alineamiento

```
1: alineamiento(c1, c2)
2: j=long1= longitud de c1
3: i=long2= longitud de c2
4: caso1,caso2,caso3,score
5: f, cA1,cA2
6: ca1=c1 convertido en arreglo,ca2=c2 convertido en arreglo
7: for (i=1:i menor que long2+1) do
8:     f(i)(0)=d*i
9: end for
10: for (j=1:j menor que long1+1) do
11:     f(0)(j)=d*j
12: end for
13: for (int i=1:i menor que long2+1) do
14:     for (int j=1;j menor que long1+1) do
15:         a=coincidencia(i-1, j-1)
16:         if (a==true) then
17:             S=coin
```

```
18:         else
19:             S=dif
20:         end if
21:         caso1=f(i-1)(j-1)+S
22:         caso2=f(i)(j-1)+d
23:         caso3=f(i-1)(j)+d
24:         f(i)(j)=mayor(caso1,caso2,caso3)
25:     end for
26: end for
27: while (i mayor que 0 and j mayor que 0) do
28:     score=f(i)(j)
29:     if (coincidencia(i-1, j-1)==true) then
30:         S=coin
31:     else
32:         S=dif
33:     end if
34:     if (score==(f(i-1)(j-1)+S)) then
35:         cA1=ca1(j-1)++cA1
36:         cA2=ca2(i-1)++cA2
37:         disminuir j y i
38:     else if score==(f(i)(j-1)+d)) then
39:         cA1=ca1[j-1]++cA1
40:         cA2=ca1[j-1]++cA2
41:         disminuir j
42:     else if (score==(f(i-1)(j)+d)) then
43:         cA1=ca2[i-1]++cA1
44:         cA2=ca2[i-1]++cA2
```

45: disminuir i

46: **end if**

47: **end while**

48: return cA1

4.4. Área foliar

Este proceso se calcula utilizando dos métodos, el teorema de Pick y el conteo de los pixeles en una imagen digital. En la Figura 4.11 se muestra como encontrar los puntos en el interior y exterior del polígono, aplicando la ecuación L es el número de color blanco (82) y B es el número de puntos en color rojo (54) y el resultado es 108 unidades. La figura 4.12 muestra el conteo de los pixeles en negro dando como resultado 108 unidades cuadradas (u^2). En ambos casos el resultado es el mismo; sin embargo, el tiempo en realizar el cálculo es diferente, por lo que el método que se utiliza es el conteo de pixeles, debido que el teorema de Pick tiene un mayor tiempo de ejecución. A partir del número de pixeles de la imagen de referencia, se obtiene el área foliar aproximada en cm^2.

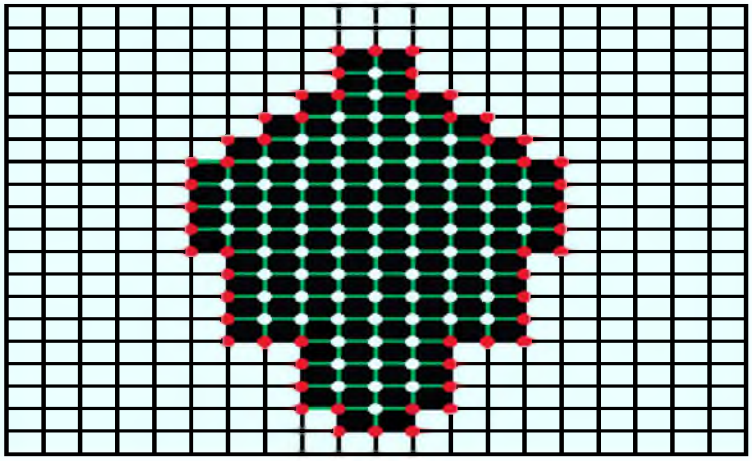


Figura 4.11: Imagen pixeleada para el teorema de Pick, donde los puntos en color rojo indican el número de puntos en su limite y los puntos en color blanco el número de puntos en su interior.

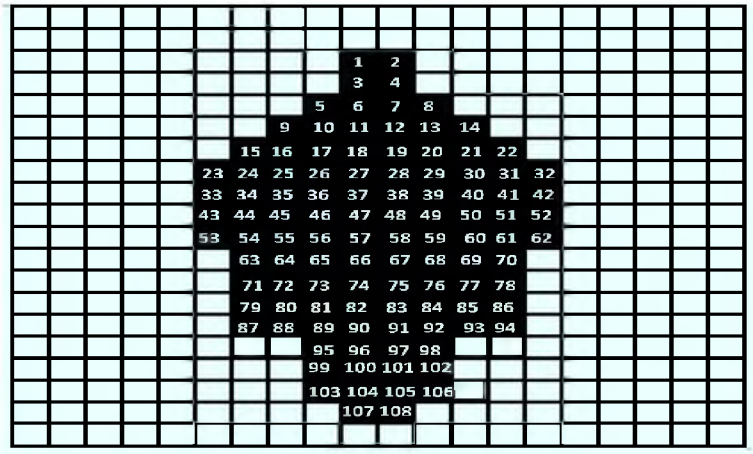


Figura 4.12: Imagen pixeleada, conteo de los pixeles en negro.

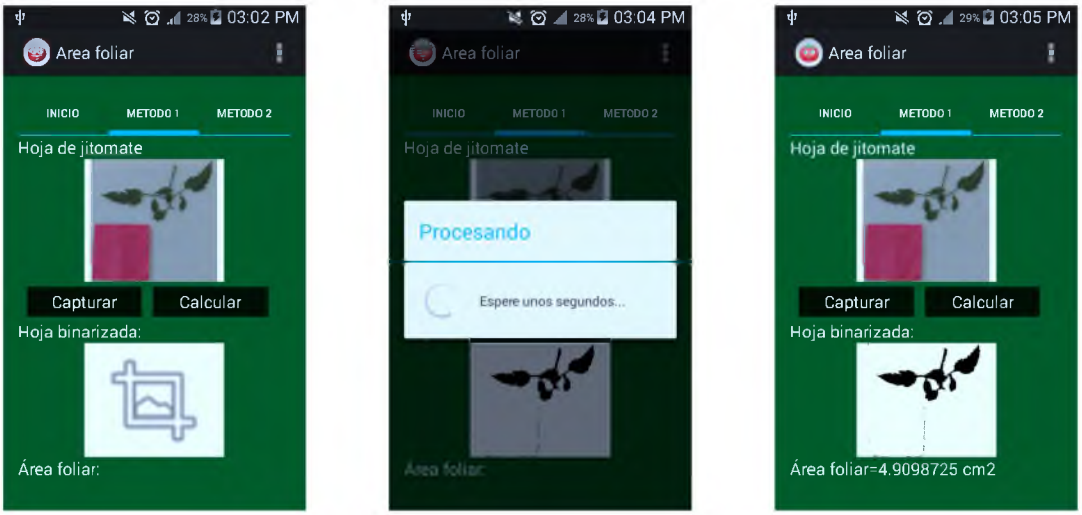
4.4.1. Pseudocódigo de la función del cálculo del área foliar

```
1: areaPixel(imagen)
2: area=0
3: for (x = 0: hasta largo de la imagen) do
4:     for (int y = 0:hasta ancho de la imagen) do
5:         if (imagen(x,y)!=-1) then
6:             area=area+1
7:         end if
8:     end for
9: end for
10: return area
```

4.5. Pruebas en la aplicación móvil

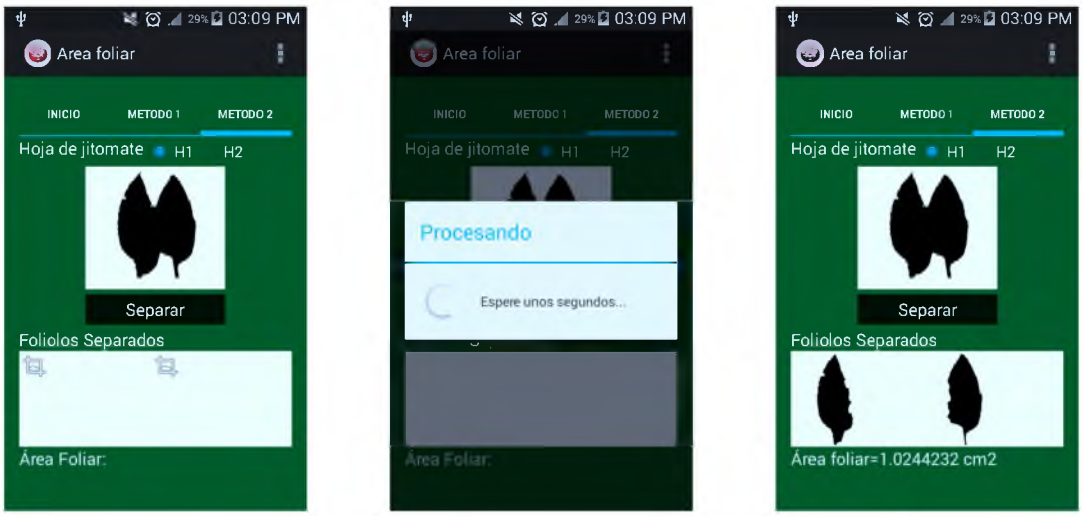
La aplicación es dividida en tres partes: Inicio, Método 1, Método 2. En la Figura 4.13 se muestra el inicio de la aplicación, en la Figura 4.14 se observa el método1 en el cual se realiza la obtención del área foliar de la hoja de jitomate, empezando por capturar la hoja para después ser binarizada (Figura 4.14-b) y por último rotarla (Figura 4.14-c). En las Figuras 4.15-a y 4.16-a se observa el método2 donde se tienen dos ejemplos de dos foliolos cuando se encuentran uno sobre otro. En las Figuras 4.15-b y 4.16-b se observa que se están ejecutando el proceso de separación y reconstrucción de los foliolos; al finalizar los foliolos reconstruidos se pueden observar en las Figuras 4.15-c y 4.16-c.

Figura 4.13: Inicio de la aplicación.



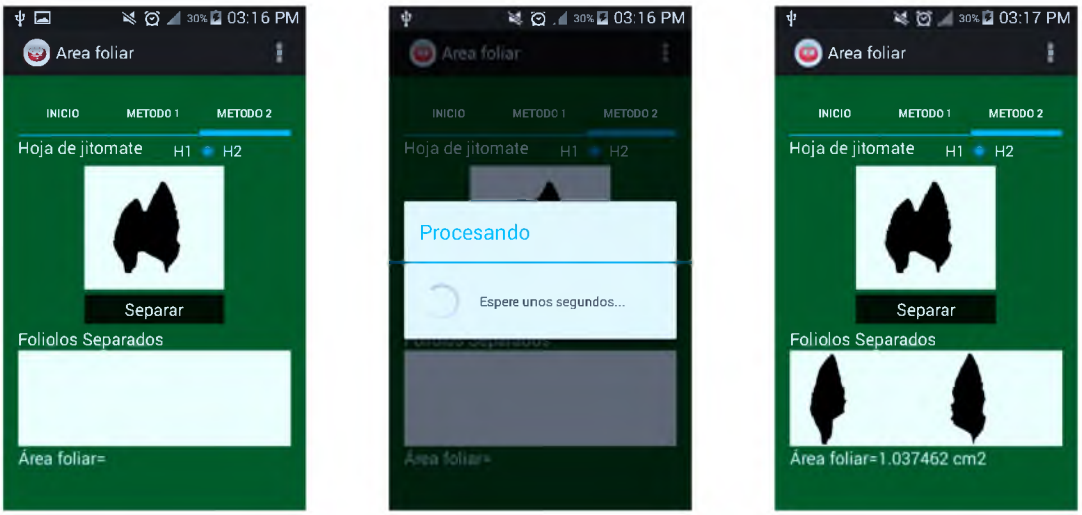
(a) Capturada (b) Binarizada (c) Rotada

Figura 4.14: Hoja de jitomate.



(a) Foliolo (b) Procesando (c) Separación y reconstrucción

Figura 4.15: Ejemplo 1, foliolos sobrepuestos.



(a) Foliolo (b) Procesando (c) Separación y reconstrucción

Figura 4.16: Ejemplo 2, foliolos sobrepuestos.

Capítulo 5

Conclusiones

En la presente tesis se cumplió con el objetivo de crear una herramienta no destructiva para calcular el área foliar en una hoja de jitomate, sin cortar o dañar una hoja. La herramienta se realizó en la plataforma de Android en el lenguaje java.

En el proceso de aplicación, primero se tiene que binarizar la imagen de acuerdo a un umbral, que se obtiene dependiendo de la calidad de la cámara en la que se ejecuta la aplicación. Existen diferentes formas para calcular el umbral, en este trabajo se utilizó el método de dos picos, como resultado se obtuvo una binarización adecuada para la obtención del área foliar.

En la restructuración y obtención de los códigos de cadena en los foliolos es necesario que los modelos y los foliolos que se reconstruirán se encuentren del mismo tamaño, por lo cual se ajustaron en una escala de 100*150 pixeles. En la regeneración de los foliolos la obtención de los códigos de cadena se realizó por dos métodos: código de cadena vértice y código Freeman en ocho direcciones. Al realizar la alineación de las cadenas resulto más adecuado utilizar los códigos de cadena en ocho direcciones, debido a que los códigos de cadena vértice solo cuenta con tres posibles valores dando como resultado un alineamiento no adecuado para reconstruir los foliolos.

Este trabajo demuestra que con la utilización de los códigos de cadena en ocho direcciones y el alineamiento de secuencias se pueden reconstruir foliolos partiendo de un modelo general. Cabe aclarar que sólo se reconstruyen foliolos que se encuentren en una sola forma vertical, ya que existen muchas posibilidades de que dos o más foliolos estén sobrepuestos, quedando esto para trabajos futuros.

El cálculo del área foliar se realizó mediante dos métodos (el teorema de Pick y número de pixeles en negro), ambos generan el mismo resultado. Sin embargo, debido a que el teorema de Pick está diseñado para una malla cuadriculada (y no para una imagen pixeleada) su tiempo de ejecución es mayor; por lo tanto se utilizó el método que cuenta el número de pixeles en negro. Utilizando un objeto de referencia se obtiene el área foliar aproximada en cm^2.

Bibliografía

[1] AGUILAR, L. J. y AZUELA, M. F. *JAVA 2 Manual de programación* México: McGraw-Hill. 2001.

[2] BOUZO, C. A., FAVARO, J. C., y ASTEGIONADO, E. D. *Estimación del área foliar en distintos cultivares de tomate (Lycopersicon esculentum mill.) utilizando medidas foliares lineales*. Investigación agraria. Producción y protección vegetales 16 (2001), 249–256.

[3] BRIBIESCA, E. A *new chain code*. The Journal of the Pattern Recognition Society (1998).

[4] CHÁVEZ, J. L. *Introducción al tratamiento digital de imágenes.* Fondo de Cultura Económica, (2007).

[5] E, J. J. E., y M, L. E. P. *Fundamentos para el procesamiento de imágenes*. Uabc, (2005).

[6] FREEMAN, H. *Computer processing of line drawing images, Comput. Surv. 6.* (1974).

[7] GERTRUDIS, P., DE LA NOVAL W, T., y G, M. *Estimación del área foliar en distintos cultivares de tomate (Lycopersicon esculentum Mill.) utilizando medidas foliares lineales.* Pastos y Forrajes 29 (2006).

[8] HARRISON, J. *An formal proof of picks theorem* Mathematical Software-ICMS. Vol:6327 pp:52-154(2010).

[9] BARRERA. J., BANON, G., y LOTUFO, R. *MMaach: A Mathematical Morphology Toolbox for the KHOROS System*. Journal of Electronic Imaging, (1998).

[10] LEROY, C., SAINT-ANDRÉ, L., y AUCLAIR, D. *Practical methods for non-destructive measurement of tree leaf area*, 2–5, (2007).

[11] LÓPEZ, J. C. M. *Alineamiento de secuencias genéticas en procesadores multicore*, 1720, (2010).

[12] MELO, S. B. *Transformaciones geométricas sobre imágenes digitales,(2005).*

[13] MOUNT, D. W. *Bioinformatics: sequence and genome analysis.* Cold Spring. Cold Spring Harbor, New York., (2004).

[14] PARKER, J. R. *Algorithms for image processing and computer vision*. Second Edition. Wiley Publishing, Inc., Indianapolis, Indiana, (2011).

[15] R., B. M., G., S. A., S., G. P., y PAREDES, D. G. *Acumulación diaria de materia seca y de potasio en la biomasa área total de tomate*, (2002).

[16] RUIZ-ESPINOZA F. H., MURILLO-AMADOR, B., GARCÍA-HERNÁNDEZ, J. L., TROYO-DIÉGUEZ, E., , PALACIOS-ESPINOZA, A., BELTRÁN-MORALES, A., FENECH-LARIOS, L., ZAMORA-SALGADO, S., MARRERO-LABRADOR, P., NIETO-GARIBAY, A., y DE LA PAZ, O. C. *Mediciones lineales en la hoja para la estimación no destructiva del área foliar en albahaca (Ocimun basilicum L)*, (2007).

[17] SÁNCHEZ-CRUZ. H, BRIBIESCA. E y RODRÍGUEZ-DAGNINO RAMÓN M. *Efficiency of chain codes to represent binary objects.* The Journal of the Pattern Recognition Society, (2006).

[18] TREIBER, M. *An Introduction to Object Recognition.* Springer London Dordrecht Heidelberg New York, (2010).