

Universidad de las Ciencias Informáticas

Facultad 1



Componente para la segmentación de imágenes de huellas
dactilares

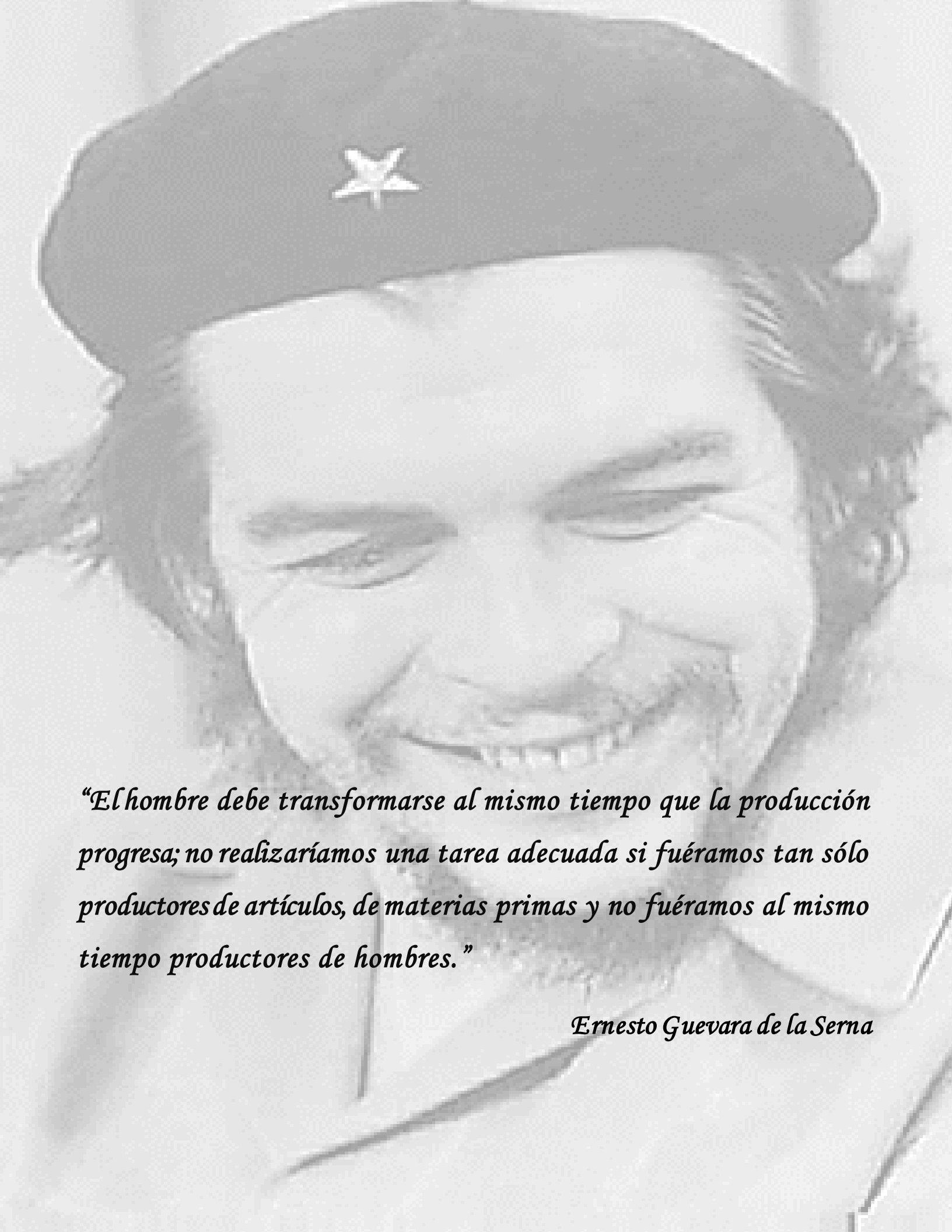
**Trabajo de Diploma para optar por el título de
“INGENIERO EN CIENCIAS INFORMÁTICAS”**

AUTOR: Carlos Enrique Matos Lezcano

Tutora: Ing. Irina Brito Reyes

Co-tutora: Ing. Evelyn Yanez Clark

La Habana, Cuba
Junio, 2014



“El hombre debe transformarse al mismo tiempo que la producción progresa; no realizaríamos una tarea adecuada si fuéramos tan sólo productores de artículos, de materias primas y no fuéramos al mismo tiempo productores de hombres.”

Ernesto Guevara de la Serna

Declaración de Autoría

Declaro ser el autor de la presente tesis, reconociendo a la Universidad de las Ciencias Informáticas los derechos patrimoniales de manera exclusiva.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Carlos Enrique Matos Lezcano

Irina Brito Reyes
Tutora

Evelyn Yanez Clark
Co-tutora

Síntesis de Tutores

Ing. Irina Brito Reyes.

Jefa del Grupo de Gestión de Proyectos (DGP) de la Universidad de las Ciencias Informáticas.

Correo electrónico: irinab@uci.cu

Ing. Evelyn Yanez Clark

Ingeniera Informática, UCI 2013

Correo electrónico: eyanez@uci.cu

Agradecimientos

A mis padres, por su inmenso amor y cariño, por educarme como un futuro hombre de bien con valores, conceptos y principios propios. A ellos por hacer de mí la persona que soy, por esto y mucho más.

A mis tutoras Irina y Evelyn, por apoyarme siempre en todo incondicionalmente, por brindarme gran parte de su tiempo a pesar de tanto trabajo.

A mi hermano por ser de gran influencia para yo seguir adelante, y permitirme tenerlo como ejemplo siempre, aunque no se lo demuestre.

A mi familia en general por apoyarme siempre en cuanto ha podido, a pesar de que me consideren la “oveja descarriada”.

A todos los que han compartido conmigo durante toda la carrera, a los que terminaron y a los que aún no lo han hecho.

A todos los amigos que han formado parte inolvidable de mi vida, a los presentes y a los que por algún motivo no se encuentran.

A mi novia, por apoyarme en momentos de estrés y desespero, y sobre todo por tener esa paciencia para conmigo.

A todos gracias.

Dedicatoria

A mis padres para darle otro motivo que los haga sentir orgullosos del hijo que han formado.

A mis tutoras porque de veras que sin ellas esto no hubiese sido posible.

A todos mis amigos, de veras les doy mil gracias por su amistad.

A mi novia, porque la amo.

Resumen

Las huellas dactilares son un identificador biométrico ampliamente utilizado, su uso se extiende desde aplicaciones policiales y forenses hasta aplicaciones civiles muy comunes, como el control de acceso. Ejemplo de estas aplicaciones son los sistemas de reconocimiento automático de huellas dactilares (AFIS, *“Automatic Fingerprint Identification System”*)

La segmentación de imágenes digitales es una de las etapas más importantes en la implementación de un sistema de reconocimiento automático de huellas dactilares. En este Trabajo de Diploma se plantea investigar sobre las diversas estrategias para la segmentación de imágenes digitales que utilizan una mezcla adecuada de operadores. A fin de adquirir superiores resultados que los reportados en el estado del arte cuando las imágenes a segmentar son de huellas dactilares latentes, es decir, son imágenes de huellas dactilares con una mala calidad. Para así posteriormente optar por una de las estrategias estudiadas e implementar la misma.

Palabras Claves: huella dactilar, verificación dactilar, segmentación.

Índice

Introducción	1
Capítulo 1 “Fundamentación Teórica”	5
1.1. Marco Conceptual.....	5
1.1.1. ¿Qué es una huella dactilar?.....	5
1.1.2. La segmentación.....	5
1.2. Biometría de la huella dactilar	6
1.3. La segmentación en la Biometría	7
1.4. Tipos de segmentación de imágenes.....	7
1.4.1. Segmentación manual.....	8
1.4.2. Segmentación automática	8
1.4.3. Segmentación semiautomática.....	8
1.5. Clasificación general de los algoritmos de segmentación de imágenes	8
1.6. Análisis de varios métodos para la segmentación de imágenes de huellas dactilares.....	9
1.6.1. Detección de puntos	11
1.6.2. Detección de líneas.....	11
1.6.3. Detección de bordes	12
1.7. Análisis de sistemas que trabajan la segmentación de imágenes.....	13
1.8. Métricas de evaluación para los algoritmos de segmentación.....	13
1.8.1. Exactitud (precisión)	14
1.8.2. Estabilidad.....	14
1.8.3. Eficiencia.....	15
1.9. Entorno de desarrollo para la solución del problema.....	15
1.9.1. Metodologías de desarrollo del software	15
- XP.....	16
1.9.1.1. Fundamentación de la elección.....	18
1.10. Lenguaje de programación	18
1.11. Herramientas.....	18
1.11.1. Entorno Integrado de Desarrollo.....	18
1.12. Tecnologías	19
1.12.1. Framework.....	19
1.12.1.1. Framework .NET (4.5)	19
1.13. Conclusiones parciales.	20

Capítulo 2 “Características, Análisis y Diseño del componente”	21
2.1. Algoritmos analizados que realizan la segmentación de imágenes de huellas dactilares	21
2.1.1. Tabla comparativa de los diferentes algoritmos.....	21
2.1.2. Descripción de los diferentes parámetros evaluados para cada algoritmo	23
• Algoritmo basado en el <i>Etiquetado de Regiones</i>	23
• Algoritmo basado en la detección de bordes mediante <i>El Gradiente</i>	24
• Algoritmo que combina el <i>Etiquetado de Regiones</i> y el uso del <i>Gradiente</i>	26
• Algoritmo de <i>Canny</i>	28
• Algoritmo de Otsu.....	31
2.1.3. Justificación de la elección.....	34
2.2. Implementación del componente	35
2.2.1. Estándar de programación.....	35
Conclusiones parciales	37
Capítulo 3 “Prueba y Validación del componente”	38
3.1. Pruebas.....	38
3.2. Validación de los resultados	38
3.2.1. Resultados Experimentales.....	39
3.3. Conclusiones parciales	41
Conclusiones	42
Recomendaciones	43
Referencias Bibliográficas	44
Glosario de Términos	45
Anexos	47

Introducción

En la actualidad la evolución de las tecnologías de la información y las comunicaciones (TIC) ha permitido automatizar y perfeccionar los sistemas de identificación personal, de forma que poseen múltiples aplicaciones y finalidades.

La Universidad de la Ciencias Informáticas tiene dentro de su objeto la comercialización de productos Software y desarrollos a la medida para varios clientes nacionales e internacionales. Para poder cumplir con su objetivo cuenta con 14 centros de desarrollo especializados en el desarrollo de software y áreas temáticas dentro de este entre los cuales se encuentra el CISED.

El Centro de Identificación y Seguridad Digital (CISED), es un centro de desarrollo de soluciones integrales, productos y servicios en el campo de la identificación y la seguridad digital. El Centro cuenta con 4 líneas investigativas: Identificación, Seguridad Digital, Tarjetas Inteligentes (*SmartCard*) y Biometría. El Centro ha desarrollado importantes proyectos donde ha incorporado componentes de verificación dactilar pero siempre de licencia propietaria y como cajas negras en su funcionamiento por lo cual se hace complejo cualquier cambio, mantenimiento y sostenibilidad en el tiempo, así como su integración con tecnologías libres para nuevos desarrollos.

Dentro de la línea de Biometría se ha venido trabajando en el desarrollo de un sistema de verificación dactilar propio del Centro debido a la necesidad para nuevos desarrollos a la medida que realiza el Centro en proyectos que brindan soluciones integradoras y seguras para varios clientes.

El uso práctico y eficaz de huellas dactilares para la identificación de personas las ha convertido en uno de los rasgos biométricos más usados en la actualidad, por lo que la variedad de usos y aplicaciones de estos sistemas también va en aumento. [1]

Los sistemas de verificación de huellas dactilares han adquirido gran importancia y auge en la actualidad puesto que son de gran ventaja en cuestiones conexas con la seguridad de diversos recursos como pueden ser recintos o información clasificada, garantizando el control de acceso y uso eficiente de los mismos. Generalmente estos sistemas son más económicos que otros sistemas de verificación, como lo son los de reconocimiento de iris o retina. Por otra parte, son más confiables que los sistemas de reconocimiento de voz o de firma y para ejecutarse necesitan fracciones de segundo. [2]

La verificación dactilar consiste básicamente en la captura de datos de la huella dactilar de una persona. Dicha captura genera una imagen digital de la huella dactilar que posteriormente es

perfeccionada mediante un pre-procesamiento de la imagen para luego extraer las características de la misma, las cuales son denominadas minucias. Esto le permite al sistema comparar los datos tomados, con la información existente en su base de datos. A este punto, hay que mencionar que los sistemas de verificación dactilar no almacenan la imagen completa de una huella dactilar, sino las minucias de la misma.

Uno de los sub-procesos que se realizan durante el pre-procesamiento de la imagen de una huella dactilar para la identificación, es el de la segmentación, el cual tiene como objetivo seleccionar la región de interés en la imagen (huella dactilar) y desechar lo indeseable como pueden ser manchas, ruidos, sombras o cualquier elemento que no conforme la huella dactilar y que podría causar problemas en las etapas posteriores del proceso de reconocimiento.

Actualmente existen disímiles algoritmos para resolver el problema de la segmentación de imágenes de huellas dactilares. En los últimos 45 años, las investigaciones y el desarrollo de algoritmos de segmentación de imágenes han seguido un crecimiento muy rápido, se ha desarrollado un elevado número de algoritmos de segmentación y este número crece continuamente cada año. [3]

En la actualidad el CISED no cuenta con ningún algoritmo para realizar la segmentación de imágenes de huellas dactilares, por lo que la presente investigación centrará su objetivo en la búsqueda y selección de un algoritmo que realice dicho proceso, posteriormente se procederá a su implementación para comprobar el correcto funcionamiento del mismo.

Por lo anteriormente expuesto se define como **problema a resolver** de este trabajo: ¿cómo lograr segmentar las imágenes de huellas dactilares como parte del pre-procesamiento de imágenes en el sistema de verificación dactilar a desarrollar en el CISED?

Para dar solución al problema enunciado se propone como **objetivo general** de esta investigación: Seleccionar un algoritmo de segmentación de imágenes de huellas dactilares que permita el pre-procesado de las imágenes de huellas dactilares en el sistema de verificación dactilar, que se desarrolla en la línea de biometría del CISED.

En consecuencia se define como **objeto de estudio** el pre-procesamiento de imágenes de huellas dactilares y el **campo de acción** está orientado a la segmentación de imágenes de huellas dactilares.

A partir del análisis del objetivo general se derivan los siguientes **objetivos específicos**:

- Analizar los algoritmos de segmentación existentes.
- Seleccionar e implementar uno de los algoritmos de segmentación analizados.

- Realizar pruebas de eficiencia y calidad al algoritmo implementado.

Para el desarrollo de la investigación los **métodos científicos** empleados son:

Analítico-Sintético: Se utiliza para el análisis de teorías y documentos. En el presente trabajo se lleva cabo el análisis de las teorías y documentos relacionados con la segmentación de imágenes de huellas dactilares, permitiendo la selección de los elementos más importantes que con este trabajo se relacionan.

De los **Métodos Empíricos**, se utilizaron los métodos de

Experimento:

Se utilizó el método Experimento en su forma Transformador, que se refiere a revelar la realidad y actuar sobre ella para transformarla. Se encuentra presente en las pruebas que comprobarán la efectividad del trabajo realizado, de acuerdo con los resultados, se decide si es necesario modificarlos para lograr la efectividad que se persigue.

Revisión documental:

Se pone de manifiesto al revisar la bibliografía existente y consultar la información en internet para llevar a cabo las tareas de investigación.

Para cumplir los objetivos propuestos se proponen las siguientes **tareas de la investigación:**

- Realizar el diseño teórico-metodológico de la investigación.
- Realizar un estudio del estado del arte sobre los algoritmos de segmentación de imágenes existentes en el mundo.
- Seleccionar tecnología, lenguaje, herramienta y metodología a utilizar en la implementación del algoritmo para la segmentación de imágenes de huellas dactilares.
- Realizar una comparación sobre los algoritmos encontrados y seleccionar uno para su posterior implementación.
- Confeccionar los casos de pruebas.
- Ejecutar las pruebas de calidad y eficiencia.

El presente documento consta de 3 capítulos:

Capítulo 1. “Fundamentación Teórica” En este capítulo se presentan los elementos teóricos que sirven de base a la investigación del problema planteado. Se puntualizan los conceptos fundamentales asociados al dominio del problema. Se definen lenguaje, metodología de desarrollo

de software, tecnología y herramienta a utilizar para la implementación del algoritmo de segmentación una vez seleccionado.

Capítulo 2. “Características, Análisis, Propuesta e Implementación del algoritmo” En este capítulo se realiza un estudio y análisis de varios algoritmos de segmentación de imágenes de huellas dactilares, para así darle solución al objetivo planteado. Se procede a la implementación del componente según el algoritmo propuesto.

Capítulo 3. “Prueba y Validación del componente” En este capítulo se realizan las pruebas según la metodología escogida para comprobar la calidad y eficiencia del componente desarrollado.

Capítulo 1 “Fundamentación Teórica”

En este capítulo se realiza un estudio del estado del arte de los distintos métodos existentes para la segmentación de imágenes de huellas dactilares. Así como también un breve estudio de distintos sistemas que utilizan la segmentación de imágenes. Finalmente se especifica el lenguaje de programación, metodología y herramienta a utilizar como parte de la propuesta de solución.

1.1. Marco Conceptual

A continuación se hace mención a los conceptos que brindan mejor comprensión de la investigación realizada.

1.1.1. ¿Qué es una huella dactilar?

Una huella dactilar es la impresión visible o moldeada que produce el contacto de las crestas papilares de un dedo de la mano (generalmente se usan el dedo pulgar o el dedo índice) sobre una superficie. Se clasifican por sus características en: **Visibles o Positivas**.-Son las que dejan los dedos al estar impregnados de algún colorante (sangre, polvo o cualquier otra sustancia con la que puedan quedar marcadas las crestas papilares y puedan ser observadas a simple vista. **Moldeadas**.-Son las que aparecen impresas en forma de molde, éstas se marcan en materia plástica (grasa, jabón, plastilina). **Naturales**.-Aparecen de forma natural en los pulpejos de ambas manos, desde los seis meses de vida intrauterina hasta la muerte e incluso en el proceso de putrefacción, y **Artificiales**.-Son aquellas que se encuentran plasmadas en forma intencional con alguna sustancia, esencialmente con tinta para su estudio. [1]

1.1.2. La segmentación

La segmentación en el campo de la visión artificial¹ es el proceso de dividir una imagen digital en varias partes (grupos de píxeles) u objetos. El objetivo de la segmentación es simplificar y/o cambiar la representación de una imagen en otra más significativa y fácil de analizar. La segmentación se usa tanto para localizar objetos como para encontrar los límites de estos dentro de una imagen. La segmentación de la imagen es el proceso de asignación de una etiqueta a cada píxel de la imagen de forma que los píxeles que compartan la misma etiqueta también tendrán ciertas características visuales similares. El resultado de la segmentación de una imagen es un

¹ Técnicas y modelos que permiten la adquisición, procesamiento, análisis y explicación de cualquier tipo de información obtenida a través de imágenes digitales.

conjunto de segmentos que cubren en conjunto toda la imagen, o un conjunto de las curvas de nivel extraídas de la imagen. Cada uno de los píxeles de una región es similar en alguna característica, como el color, la intensidad o la textura. Regiones adyacentes son significativamente diferentes con respecto a la(s) misma(s) característica(s). [4]

Los algoritmos de segmentación se basan en los siguientes principios:

1. Discontinuidades del nivel de gris. Consisten en segmentar la imagen a partir de los cambios grandes en los niveles de gris entre los píxeles. Las técnicas que utilizan las discontinuidades como base son la detección de líneas, puntos aislados, entre otros.
2. Similitud de niveles de gris. Es lo contrario al método anterior, las divisiones de la imagen se hacen agrupando los píxeles que tienen características similares. Algunas técnicas que usan este principio son el crecimiento de regiones, la detección de bordes y la segmentación según el umbral.

Ejemplos de las aplicaciones prácticas de la segmentación de imágenes pueden ser apreciadas en:

- Pruebas médicas
- Localización de tumores y otras patologías
- Medida de volúmenes de tejido
- Cirugía guiada por ordenador
- Reconocimiento de iris
- Reconocimiento facial
- Reconocimiento de huella dactilar

1.2. Biometría de la huella dactilar

El primer estudio científico publicado sobre la estructura de crestas, valles y poros de las huellas dactilares data de 1684, realizado por el morfologista inglés Nehemiah Grew. Desde entonces han sido mucho los investigadores que han trabajado en este campo. En general los estudios de principios de 1800 llegaron a dos importantes conclusiones que, hasta hoy, han servido de base para el reconocimiento biométrico, especialmente en entornos forenses: la no existencia de dos huellas de individuos diferentes con un patrón de crestas coincidentes, y la invariabilidad en el tiempo de dichos patrones durante toda la vida del individuo. A principios de 1900, se admitieron las siguientes características biológicas de las huellas dactilares como base de identificación de individuos [1]:

- ✓ La estructura de crestas y valles de un individuo, aunque puede variar, lo hace dentro de unos límites tan reducidos, que hacen posible una clasificación sistemática.
- ✓ Los detalles de las estructuras de crestas y valles, así como las minucias, son particulares de cada individuo e invariables en el tiempo.

1.3. La segmentación en la Biometría

El reconocimiento de los elementos de las imágenes es una tarea de alta complejidad, que usualmente muestra los mejores resultados al ser realizada por los humanos. Sin embargo, la delineación de los mismos requiere una precisión esencial, mejor obtenida con la utilización de medios computacionales. Algoritmos cuidadosamente diseñados pueden guiar a excelentes soluciones para el problema de la delineación. El nivel al que se realiza esta subdivisión depende de la aplicación en particular. En la mayor parte de los casos, una buena segmentación dará lugar a buenos resultados de la aplicación que se esté desarrollando, por lo que se debe poner gran empeño en este proceso. [1]

En el campo de la Biometría, la segmentación de imágenes de huellas dactilares forma parte de un conjunto de sub-procesos (Normalización, Segmentación, Definición de perfil de las crestas, Adelgazamiento, Extracción de minucias, Eliminación de minucias falsas) para la mejora de la imagen de huella dactilar que se ejecutan secuencialmente en un orden pre-establecido.

La fase de segmentación pretende separar la zona de la imagen que contiene la huella dactilar respecto al fondo, que carece de información relevante para la obtención de minucias. Para la misma se han desarrollado una gran cantidad de algoritmos, cada uno con sus respectivas peculiaridades y características con el fin de realizar el proceso de segmentación lo más eficiente posible.

1.4. Tipos de segmentación de imágenes

Para realizar la segmentación existen muchos algoritmos que aportan diversos resultados, en dependencia de las imágenes con las que trabajen y su funcionamiento particular. Estos métodos se pueden clasificar en dependencia de su nivel de automatización en: métodos manuales, automáticos y semiautomáticos. Cada uno aporta una serie de ventajas y desventajas que actúan a la hora de tomar decisiones sobre ellos.

1.4.1. Segmentación manual.

La manera más general y fácil de lograr la segmentación es trazando manualmente las características y elementos importantes de las estructuras presentes en las imágenes. En este caso, el usuario delinea con alguna herramienta las estructuras relevantes, lo que le da la ventaja de redibujar cualquier porción y corregir algún error introducido.

Este método es robusto (siempre aplicable) pero consume mucho tiempo y recursos, cuando se trabaja con cúmulos de imágenes muy grandes se vuelve impracticable y no es preciso debido a que el usuario generalmente se desvía del contorno deseado y si los objetos son difíciles de delinear, pueden ser difíciles de segmentar. [5]

1.4.2. Segmentación automática

La segmentación automática, como su nombre lo indica es el tipo de segmentación donde el ordenador realiza todo el proceso de manera automática sobre la imagen objetivo. Es improbable que los métodos de segmentación automática reemplacen alguna vez a los métodos manuales pero si es probable que se conviertan en elementos cruciales para el análisis de imágenes médicas. [6]

1.4.3. Segmentación semiautomática

En este tipo de modalidad el ordenador realiza el proceso, pero el usuario interviene en determinados momentos sobre el mismo, como por ejemplo para definir las regiones de interés y otros parámetros mediante dispositivos como el ratón o para corregir resultados. Luego, los algoritmos se aplicarán de forma que se elige el camino que mejor se ajusta al borde de la imagen. [3]

1.5. Clasificación general de los algoritmos de segmentación de imágenes

Teniendo en cuenta la documentación existente y tantos algoritmos de segmentación de imágenes que se han desarrollado y se siguen desarrollando, se hizo necesario establecer una clasificación que contuviese a todos esos algoritmos y facilitase la realización de estudios sistemáticos sobre los mismos.

Hacer una clasificación de los algoritmos en grupos, en principio, es un problema para definir los conjuntos y los sub-conjuntos de dichos algoritmos conociendo la diversidad de bases teóricas. Sin

embargo, se ha creído que los grupos de algoritmos, después de hacer una clasificación que respete las bases del proceso de segmentación y sus objetivos, deben satisfacer las siguientes cuatro condiciones:

1. Cada algoritmo debe estar en un grupo.
2. Todos los grupos unidos, deben incluir a todos los algoritmos.
3. Los algoritmos de un mismo grupo deben tener propiedades comunes.
4. Los algoritmos de grupos diferentes deben tener ciertas propiedades que los diferencie.

Para llevar a vías de hecho las cuatro condiciones anteriores, hay que encontrar “ciertos criterios” que faciliten la clasificación de los algoritmos de segmentación de imágenes. Los dos criterios de clasificación podrían ser: 1) una propiedad vinculada al pixel segmentado (discontinuidad, similitud), y 2) una propiedad vinculada a la estrategia del segmentado (paralelo, secuencial). [4]

1.6. Análisis de varios métodos para la segmentación de imágenes de huellas dactilares

El análisis del procesamiento de imágenes de huellas dactilares comprende todos los métodos y técnicas que se utilizan para extraer información de una imagen digital de una huella dactilar. El primer paso para ello lo constituye la segmentación, que se ocupa de descomponer la imagen en sus partes constituyentes, es decir, la región de interés y el fondo (así como ruidos que estén dentro de la región de interés), basándose en ciertas características locales que nos permiten distinguir una huella del fondo.

Las imágenes están constituidas por regiones o zonas que tienen características homogéneas (nivel de gris, textura, y momentos.). Generalmente estas regiones corresponden a objetos de la imagen. La segmentación de una imagen consiste en la división o partición de la imagen en varias zonas o regiones homogéneas y disjuntas a partir de su contorno, su conectividad, o en términos de un conjunto de características de los píxeles de la imagen que permitan discriminar unas regiones de otras. Los tonos de gris, la textura, los momentos, la magnitud del gradiente, la dirección de los bordes, las modas de los tonos de gris en ventanas 3x3, 7x7 y 15x15, son características a utilizar para la segmentación. [4]

Se conoce como segmentación completa, cuando las regiones disjuntas corresponden directamente a objetos de la imagen y segmentación parcial, cuando las regiones no se

corresponden directamente con objetos de la imagen. Para conseguir la segmentación completa se necesita un conocimiento específico del dominio de la escena.

La segmentación trata de distinguir si un píxel pertenece, o no, a un objeto de interés y, por lo tanto, produce una imagen binaria. Todavía no hay una teoría unificada de la segmentación de imágenes, solamente se conoce de un conjunto de algoritmos que realizan dicha operación para el procesamiento de una imagen.

Los algoritmos de segmentación de imágenes de huellas dactilares se basan en alguna de las tres propiedades siguientes:

- a) Discontinuidad en los tonos de gris de los píxeles de un entorno, que permite detectar puntos aislados, líneas y aristas (bordes).
- b) Similitud en los tonos de gris de los píxeles de un entorno, que permite construir regiones por división y fusión, por crecimiento o por un valor umbral.
- c) Conectividad de los píxeles, desempeña un papel importante en la segmentación. Vale aclarar que una región D se dice conexa o conectada si para cada par de píxeles de la región existe un camino formado por píxeles de D que los conecta. Un camino de píxeles es una secuencia de píxeles adyacentes (que pertenecen a su entorno inmediato).

Los métodos de segmentación se pueden agrupar en cuatro clases diferentes:

- a) Métodos basados en píxeles, que a su vez pueden ser:
 - Locales (basadas en las propiedades de los píxeles y su entorno)
 - Globales (basadas en la información global obtenida, por ejemplo, con el histograma de la imagen).
- b) Métodos basados en bordes.
- c) Métodos basados en regiones, que utilizan las nociones de homogeneidad y proximidad geométrica, como las técnicas de crecimiento, fusión o división.
- d) Métodos basados en modelos.

Antes de pasar a estudiar alguno de estos métodos vale explicar algunas técnicas para la detección de puntos, líneas, y bordes, como herramientas previas para un mayor entendimiento de los métodos de segmentación.

1.6.1. Detección de puntos

Un punto aislado de una imagen tiene un tono de gris que difiere significativamente de los tonos de gris de sus píxeles vecinos, es decir, de los ocho píxeles de su entorno 3×3. Una máscara para detectar un punto aislado es la siguiente:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Aplicando esta máscara al píxel (i,j) se obtiene:

$$g(i,j) = -f(i-1,j+1)-f(i-1,j)-f(i+1,j+1)- f(i-1,j)+ 8f(i,j)-f(i+1,j)-f(i-1,j-1)-f(i,j-1)-f(i+1,j-1)$$

Se dice que el píxel (i,j) es un punto aislado si $|g(i,j)|>T$ donde T es el valor umbral fijado de forma empírica. Dicho valor depende de la aplicación que se vaya a realizar.

Sin embargo, esta máscara puede detectar, como puntos aislados, píxeles que forman parte de un borde. Por ello, es más conveniente utilizar el filtro no lineal siguiente:

$$R(i, j) = \min_{\substack{(r,s) \in N_8(i,j) \\ (r,s) \neq (i,j)}} |f(r, s) - f(i, j)|$$

Se dice que el píxel (i,j) es un punto aislado si $|R(i,j) > T|$. [5]

1.6.2. Detección de líneas

Una *línea* es una secuencia de píxeles en la que dos píxeles consecutivos están conectados, es decir, son vecinos en un entorno 3×3 de alguno de ellos. Cada píxel se puede conectar con alguno de sus 8 píxeles vecinos, y por lo tanto, se tendría sólo 4 direcciones (tramos lineales) posibles: horizontal, vertical, de 45° y de -45°. En consecuencia, cualquier curva digital va a venir dada por una línea compuesta sólo por tramos lineales de cualquiera de estos cuatro tipos. Así, para la detección de líneas es posible utilizar una máscara 3×3 que se irá moviendo por los píxeles de la imagen. Los píxeles que forman parte de una línea tendrán respuestas extremas. Por ejemplo, para detectar una línea o segmento horizontal (línea recta), cada píxel del segmento tiene un tono

de gris que difiere del tono de gris de los píxeles vecinos superiores e inferiores. Por ello, se utiliza la siguiente máscara para detectar una línea recta horizontal:

$$M_1 = \begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix}$$

Para detectar líneas verticales, de 45° y de -45° , conviene utilizar, respectivamente, las máscaras siguientes:

$$M_2 = \begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix}, \quad M_3 = \begin{bmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{bmatrix}, \quad \text{y} \quad M_4 = \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$$

Como se puede observar la dirección preferida de cada máscara está ponderada con un coeficiente mayor que las demás direcciones posibles.

Aplicando las cuatro máscaras anteriores a todos los píxeles de la imagen. Sea $g_k(i,j)$ el resultado de aplicar la máscara M_k centrada en el píxel (i,j) . Fijado un valor umbral $T > 0$, se asegura que el píxel (i,j) constituye una parte horizontal de una línea si

$$|g_1(i,j)| > |g_k(i,j)|, \quad \forall k \neq 1, \quad \text{y} \quad |g_1(i,j)| > T.$$

1.6.3. Detección de bordes

Un *borde* o *arista* es la frontera entre dos regiones cuyos tonos de gris difieren significativamente o tienen propiedades diferentes, como ocurre en el caso de texturas. Si se quiere detectar los bordes hay que hacer énfasis en los cambios bruscos de los niveles de gris de píxeles vecinos y suprimir aquellas áreas con valores de gris constantes. Los operadores derivada serán útiles para realizar dichas tareas.

Las técnicas de segmentación basadas en la detección de bordes son adecuadas cuando las regiones son suficientemente homogéneas de manera que la transición entre regiones se pueda realizar en base a los tonos de gris. En caso contrario es más adecuado utilizar las técnicas de segmentación basadas en regiones. [7]

Un *borde local* (*eje o arista local*), es un píxel cuyo nivel de gris difiere significativamente del nivel de gris de algunos píxeles de su entorno. Es decir, hay diferencia de contraste local. Ello se debe esencialmente a dos situaciones:

- a) El píxel forma parte del borde entre dos regiones diferentes de la imagen (cada región tiene cierta homogeneidad en sus niveles de gris, con respecto a algún criterio de homogeneidad).
- b) El píxel forma parte de un arco muy fino sobre un fondo de diferente nivel de gris.

1.7. Análisis de sistemas que trabajan la segmentación de imágenes

Existen varios paquetes de software para realizar la segmentación de imágenes.

1. **ITK** - Kit de herramientas de segmentación de imágenes.
2. **ITK-SNAP** - Es una interfaz gráfica que combina segmentación manual y semiautomática, con conjuntos de nivel.
3. **MITK** - Tiene un módulo para la segmentación manual.
4. **OpenCV** - Es una librería de visión artificial originalmente desarrollada por Intel.
5. **Fiji** - Es un paquete de procesamiento de imágenes que incluye varios plug-ins de segmentación.

1.8. Métricas de evaluación para los algoritmos de segmentación

Según lo planteado en la introducción del presente trabajo de diploma, el desarrollo de algoritmos de segmentación es un área en el que se lleva empleado mucho esfuerzo; no obstante, es patente que hay menos interés puesto en la evaluación precisa de estos algoritmos que en su desarrollo. En el mundo de la investigación el desarrollo de nuevas técnicas de segmentación es un ámbito de gran interés, mientras que no siempre se dedica el esfuerzo suficiente en la evaluación precisa, sistemática y exhaustiva de dichas técnicas usando métricas de medición del desempeño estándar o uniformes.

A pesar de que muchos diseñadores de algoritmos de segmentación suelen proporcionar algo de información sobre la evaluación de sus algoritmos, aún están por desarrollarse marcos y enfoques globales y consistentes con los que poder evaluar el desempeño de los distintos algoritmos. Aunque han sido propuestas teorías unificadas, el problema no se ha resuelto aún de manera satisfactoria. Como consecuencia, encontrar el algoritmo de segmentación adecuado para una

tarea en particular, así como elegir los parámetros internos al algoritmo de segmentación que son óptimos para la tarea a resolver es aún un problema. Es por esta razón de vital importancia identificar las métricas apropiadas para evaluar y comparar algoritmos de segmentación. A continuación se describen algunas de las diversas métricas que han venido siendo usadas por los desarrolladores a la hora de optimizar y evaluar sus algoritmos.

En todo tipo de tareas en las cuales esté presente la segmentación de imágenes, hay un consenso sobre la existencia de tres tipos de métricas que deben evaluarse, y estas son la exactitud, la estabilidad y la eficiencia. [3]

1.8.1. Exactitud (precisión)

La exactitud de un algoritmo de segmentación se refiere al grado en el que los resultados de la segmentación coinciden con la segmentación verdadera. Hay situaciones en las que la segmentación verdadera es conocida, y otras en las que no. En dichas situaciones la segmentación verdadera se sustituye por una segmentación manual realizada por un experto, o usando un algoritmo de segmentación del cual se conoce que produce resultados precisos. Idealmente la medida de la precisión debe reflejar el grado de discordancia entre la segmentación verdadera y aquella que se desea evaluar, y no debería depender de las dimensiones de la imagen.

1.8.2. Estabilidad

La estabilidad de un algoritmo de segmentación da una idea de la repetitividad de la técnica al usarse con un tipo particular de imagen. En la evaluación de la estabilidad de un algoritmo se ha de determinar la métrica apropiada para un conjunto de imágenes usando tanto el algoritmo como estimaciones repetidas de la segmentación verdadera, es decir, diversas segmentaciones manuales de la misma imagen. De este modo, repetir las segmentaciones proporciona una estimación de la varianza de la misma.

Usar estimaciones de la varianza tanto de la segmentación del algoritmo, como de la segmentación verdadera, permite comparar la variabilidad del algoritmo con la variabilidad en la determinación de la segmentación verdadera. De este modo, pueden compararse las variabilidades de diversos algoritmos con otro algoritmo o con el mismo algoritmo pero con un conjunto distinto de parámetros. Estas comparaciones deben incluir pruebas estadísticas como el análisis de varianza (ANOVA). Se debe ser especialmente cuidadoso para asegurar que se

emplea un número suficiente de pruebas de segmentación que permitan comparaciones significativas.

1.8.3. Eficiencia

La eficiencia de la segmentación da una idea del uso práctico del algoritmo. Es frecuente que se mida la eficiencia a través del tiempo de segmentación. Sin embargo, la eficiencia debería incluir la cuantificación de todos los aspectos de la interacción con el usuario así como valorar que la técnica de segmentación sea apropiada o no para todas las imágenes. En consecuencia, además del tiempo de ejecución del algoritmo de segmentación, deben considerarse los tiempos de inicialización, edición e inspección para ser documentados también.

1.9. Entorno de desarrollo para la solución del problema

Durante el desarrollo de software, muchas tareas y actividades se tornan engorrosas, trayendo consigo que el proceso de desarrollo de software se convierta en riesgoso, difícil de controlar, de no dársele un tratamiento correcto, lo que se obtiene son clientes insatisfechos con el resultado obtenido. La forma para solucionar o tratar de llevar a cabo un eficiente desarrollo es aplicando una metodología de desarrollo de software, que no es más que un conjunto de pasos y procedimientos que deben seguirse para desarrollar un software; indican quién debe hacer cada actividad, cuándo hacerla y qué debe hacer. [8]

1.9.1. Metodologías de desarrollo del software

Las metodologías de desarrollo se clasifican en tradicionales y ágiles (Ver [Anexo 1 Comparación entre metodologías ágiles y tradicionales](#)). Entre las principales metodologías tradicionales están RUP y MSF, estas se recomiendan en proyectos complejos y de larga duración, obteniendo resultados satisfactorios cuando el equipo de trabajo tiene experiencia en su aplicación. La corrección de errores es muy costosa por ser metodologías poco flexibles a cambios, debido a que el cliente y el equipo de desarrollo no mantienen una relación directa, no siendo el caso de la presente investigación. [8]

Con la utilización de estas metodologías se genera una documentación amplia generando artefactos necesarios e innecesarios en algunos casos. Estas características hacen engorroso el desarrollo de proyectos pequeños, ralentizando su implementación y generando documentación que no será utilizada. Las metodologías robustas se guían por una fuerte planificación, centrando

toda la atención en la documentación exhaustiva de los procesos llevados a cabo en el desarrollo, así como en las herramientas y notaciones que se utilizarán y en cumplir el plan del proyecto. [9]

Para la implementación del algoritmo de segmentación de imágenes de huellas dactilares quedó descartada la selección de una metodología tradicional, dado que sus características no se adaptan a los requisitos de la investigación. Entre las metodologías ágiles cabe destacar SCRUM, Crystal y XP (*Extreme Programming*). Estas hacen menos énfasis en la documentación, puesto que el funcionamiento y el desarrollo del software son más importantes que la misma. La regla a seguir es no realizar documentos a menos que sean necesarios de forma inmediata, el cliente es muy importante el cual pasa a formar parte del equipo de desarrollo y también son muy flexibles ante los cambios y no siguen estrictamente un plan. En cuanto a metas a seguir las metodologías ágiles presentan los siguientes principios [9]:

- La prioridad es satisfacer al cliente mediante tempranas y continuas entregas de *software* que le aporte un valor.
- Dar la bienvenida a cambios. Se capturan los cambios para que el cliente tenga una ventaja competitiva.
- Entregar frecuentemente un *software* que funcione desde un par de semanas a un par de meses, con el menor intervalo de tiempo posible entre entregas.
- Los clientes del negocio y los desarrolladores deben trabajar juntos a lo largo del proyecto.
- La atención continua a la calidad técnica y al buen diseño mejora la agilidad.
- La simplicidad es esencial.

- XP

La programación extrema es una metodología de desarrollo ligero (o ágil) basada en una serie de valores y de prácticas de buenas maneras que persigue el objetivo de aumentar la productividad a la hora de desarrollar software.

Este modelo de programación se basa en una serie de metodologías de desarrollo de software en la que se da prioridad a los trabajos que dan un resultado directo y que reducen la burocracia que existe alrededor de la programación. [10]

Una de las características principales de este método de programación, es que sus ingredientes son conocidos desde el principio de la informática. Los autores de XP han seleccionado aquellos

que han considerado mejores y han profundizado en sus relaciones y en cómo se refuerzan los unos con los otros.

A continuación se muestran ciertas particularidades de XP que se adaptan a las necesidades de un software, así como a las circunstancias de trabajo:

- Desarrollo iterativo e incremental: pequeñas mejoras, unas tras otras.
- Pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión. Se aconseja escribir el código de la prueba antes de la codificación.
- Frecuente interacción del equipo de desarrollo con el cliente. Se recomienda que un representante del cliente trabaje junto al equipo de desarrollo.
- Corrección de todos los errores antes de añadir una nueva funcionalidad.
- Refactorización del código, es decir, reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento. Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo.
- Simplicidad en el código: la programación extrema apuesta que es más sencillo hacer un código simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar funciones complicadas y quizás nunca utilizarlas.

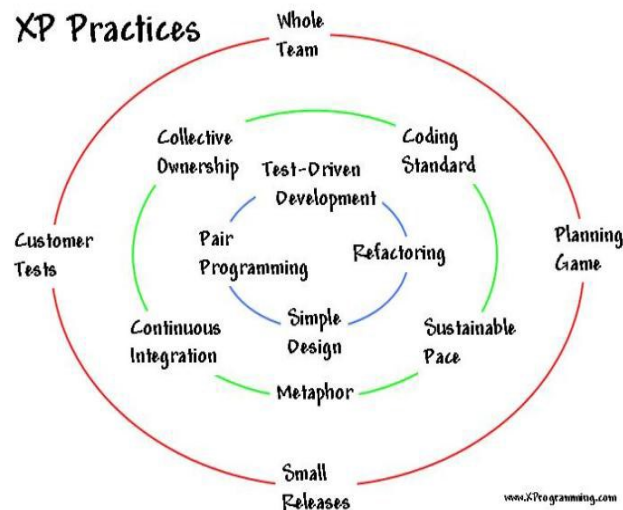


Fig. 1 Modelo de Extreme Programming (XP)

La metodología XP, como metodología ágil, enfatiza en el carácter interactivo e incremental del desarrollo, donde una iteración es un período de una a cuatro semanas, en el cual el cliente selecciona las funcionalidades que desea que se implementen en dicha iteración

1.9.1.1. Fundamentación de la elección

Luego de un análisis de las características, ventajas y desventajas de la metodología ágil XP se acordó el uso de la misma debido a que el ambiente de desarrollo basado en un único local y un ordenador por parte del equipo de trabajo se ajustan perfectamente a las características de XP.

1.10. Lenguaje de programación

Un lenguaje de programación es un idioma artificial diseñado para expresar procesos que pueden ser ejecutados por máquinas computadoras. Pueden usarse para expresar algoritmos con precisión, o como modo de comunicación humana. Al proceso mediante el cual se escribe, se prueba, se depura, se compila y se mantiene el código fuente de un programa informático se denota programación.

C#: C Sharp es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET, que después fue aprobado como un estándar por la ECMA (ECMA-334) e ISO (ISO/IEC 23270). Presenta como principales características: sencillez, orientación a componentes y eficiente.

1.11. Herramientas

En la elaboración de un software se pueden utilizar diferentes herramientas (herramienta de programación y herramienta de modelado) y es de mucha importancia la sabia elección de cada una de ellas, para garantizar el mejor funcionamiento posible del mismo.

1.11.1. Entorno Integrado de Desarrollo

Un entorno de desarrollo integrado, llamado también IDE (según sus siglas en inglés, *Integrated Development Environment*), es un programa informático compuesto por un conjunto de herramientas de programación. Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, que consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). [11]

Microsoft Visual Studio Ultimate 2013: es un entorno de desarrollo integrado para sistemas operativos Windows. Soporta múltiples lenguajes de programación tales como C++, C#, Visual Basic .NET, F#, Java, Python, Ruby, PHP; al igual que entornos de desarrollo web como ASP.NET, MVC, Django, entre otros. **Visual Studio Ultimate 2013** simplifica la compilación, la depuración y el despliegue de las aplicaciones en una variedad de plataformas incluyendo SharePoint. También viene con el soporte integrado para el desarrollo con pruebas y con las herramientas de depuración que ayudan a garantizar unas soluciones de alta calidad.

1.12. Tecnologías

1.12.1. Framework.

Un *framework* es una estructura de soporte definida a partir de la cual un proyecto de software puede ser organizado y/o desarrollado. Típicamente, un *framework* incluye soporte de programas, bibliotecas y un lenguaje interpretado para ayudar a desarrollar y unir los diferentes componentes de un proyecto. [12]

El uso de *framework* simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además proporciona estructura al código fuente, forzando al desarrollador a crear código más legible y más fácil de mantener. También facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas. [13]

1.12.1.1. Framework .NET (4.5)

El Framework .Net es una infraestructura sobre la que se reúne todo un conjunto de lenguajes y servicios que simplifican enormemente el desarrollo de aplicaciones, mediante la cual se ofrece un entorno de ejecución altamente distribuido, que permite crear aplicaciones robustas y escalables .

Ventajas:

- a. El desarrollo rápido de aplicaciones. Los componentes incluidos constituyen una capa que libera al programador de la escritura de código de bajo nivel.
- b. El uso y la programación de componentes que siguen una política de diseño uniforme. Un framework orientado a objetos logra que los componentes sean clases que pertenezcan a una gran jerarquía de clases, lo que resulta en bibliotecas más fáciles de aprender a usar.

1.13. Conclusiones parciales.

Después del estudio y análisis realizado del objeto de investigación, apoyado en los métodos de la investigación definidos, se pudo construir el marco teórico-conceptual que soporta la investigación. Se adquirieron los conocimientos necesarios sobre la herramienta, metodología y lenguaje a utilizar para la implementación del algoritmo de segmentación de imágenes de huellas dactilares una vez seleccionado.

Como herramienta, lenguaje y tecnología para el desarrollo de la solución propuesta se propone que el lenguaje de programación será C# por ser orientado a objetos y desarrollado y estandarizado por Microsoft, además de que el sistema al cual se debe integrar el componente está implementado utilizando dicho lenguaje, lo que facilitará el trabajo a la hora de llamar a funciones y usar tipos de datos nativos del Framework .NET. Dicho lenguaje se pondrá en práctica haciendo uso de Microsoft Visual Studio 2013 como IDE

Capítulo 2 “Características, Análisis y Diseño del componente”

El objetivo que se persigue con la elaboración de este capítulo es describir las ventajas y desventajas de cada uno de los algoritmos para la segmentación de imágenes de huellas dactilares, con el objetivo de seleccionar el que más se ajusta a las necesidades del CISED, así como justificar la decisión tomada. Posteriormente se procederá a la implementación del algoritmo seleccionado.

2.1. Algoritmos analizados que realizan la segmentación de imágenes de huellas dactilares

Una variedad de algoritmos para la segmentación de imágenes de huellas dactilares se han propuesto en los últimos decenios. Diferentes técnicas de segmentación están presentes en el estado del arte, todos los métodos no son igual de buenos para un tipo particular de imagen. Por lo tanto, a pesar de varias décadas de investigación, no existe un método universalmente aceptado para la segmentación de imágenes y por lo tanto sigue siendo un problema difícil en el procesamiento de imágenes de huellas dactilares. Sobre la base de las diferentes tecnologías y enfoques de la segmentación de imágenes se encuentran un conjunto de algoritmos, evaluados mediante el uso de métricas para su respectiva comparación. [5] [3]

2.1.1. Tabla comparativa de los diferentes algoritmos

	Algoritmos				
Parámetros	A	B	C	D	E
Func.	Asigna a cada píxel un id.	Derivadas parciales para cada píxel	Func (A) + Func (B)	Func (B) + cálculo de la media	Media, Varianza, Histograma
C.M	Fácil	Media	Elevada	Media	Fácil
C.C	Fácil	Media	Media	Media	Fácil
N. g	No	Sí	Sí	No	Sí
Bnz.	No; binariza antes de segmentar	Sí	Sí	No, binariza antes de segmentar	Sí
CPU	Básico	Básico	Básico	Básico	Básico

Valor Umbral	No	No	No	No	Sí
Operaciones	8	6	12	8	8

Fig. 2 Algoritmos para la segmentación de imágenes de huellas dactilares

Leyenda

- A- Algoritmo basado en el *Etiquetado de Regiones*.
 - B- Algoritmo basado en la detección de bordes mediante *El Gradiente*.
 - C- Algoritmo que combina el *Etiquetado de Regiones* y el uso del *Gradiente*.
 - D- Algoritmo de *Canny*.
 - E- Algoritmo de *Otsu*.
- **Funcionamiento (Func.):**
Breve descripción de las funciones que realiza cada algoritmo.
 - **Complejidad matemática (C.M):**
Estima -en dependencia del funcionamiento que posee cada algoritmo- un valor para su correcto entendimiento matemático (Fácil, Media, Elevada).
 - **Complejidad computacional (C.C):**
Estima -en dependencia de la complejidad matemática que tiene cada algoritmo- un costo para su adecuada implementación (Fácil, Media, Elevada).
 - **Nivel de gris (N. g):**
Indica para cada algoritmo si el mismo trabaja con la imagen en escala de grises o no (Sí o No).
 - **Binarización (Bnz.):**
Indica para cada algoritmo si el mismo realiza una Binarización o no de la imagen una vez segmentada (Sí o No).
 - **Hardware (CPU):**
Especifica el tipo de hardware de cómputo necesario para la implementación de cada algoritmo.
 - **Valor umbral:**
Determina si el algoritmo realiza un método para calcular un valor umbral óptimo (Sí) o si simplemente se basa en resultados experimentales para la obtención de dicho valor umbral (No).
 - **Cantidad de operaciones (Operaciones):**

Detalla la cantidad de operaciones (así como una breve descripción de las mismas) que realiza cada algoritmo para realizar el proceso de segmentación.

2.1.2. Descripción de los diferentes parámetros evaluados para cada algoritmo

Algoritmo basado en el *Etiquetado de Regiones*

- **Func.**

El algoritmo de etiquetado permite identificar zonas formadas por píxeles interconectados en una imagen binaria. Estas zonas pueden representar regiones de interés, son identificadas mediante valores numéricos únicos (etiquetas), lo que permite separar la región de interés en la imagen. El proceso de etiquetado consiste en realizar un análisis de conectividad, para determinar si dos o más píxeles están unidos y por lo tanto pertenecen a un mismo objeto. [14]

- **C.M**

Fácil entendimiento, pues no requiere de conocimientos matemáticos muy elevados para su entendimiento

- **C.C**

Fácil implementación, debido a que no demanda de una programación avanzada.

- **N. g**

No trabaja la imagen en escala de grises.

- **Bnz.**

Este algoritmo toma como entrada una imagen binaria en donde el fondo de la imagen es negro y las regiones de interés son blancas.

- **CPU**

Este algoritmo no requiere de gran cantidad de recursos de hardware para su correcto funcionamiento.

- **Operaciones**

- 1- Se prueba si el píxel actual no es parte del fondo. En caso falso ir al paso 7.
- 2- Se verifica si el píxel es el primero de una nueva región.
- 3- Se evalúa si alguno de los píxeles vecinos ha sido etiquetado.
- 4- Si existe más de un píxel vecino etiquetado, se comparan los valores de todos ellos. Si la etiqueta es la misma, se asigna esta etiqueta al píxel que se está evaluando.
- 5- Si las etiquetas son diferentes, se almacena el dato en una tabla de equivalencias, es decir que ambos valores pertenecen a una misma región para realizar la corrección después de etiquetar toda la imagen.
- 6- Por último, si el píxel no posee vecinos etiquetados, se etiqueta con un nuevo valor.
- 7- Se procede a analizar otro píxel.
- 8- Después de etiquetar toda la imagen, se deben corregir las etiquetas utilizando la tabla de equivalencias creada. Esta corrección es necesaria porque en general un objeto puede estar etiquetado con más de un valor. La tabla de equivalencias posee información sobre las etiquetas que pertenecen a un mismo objeto. Por lo tanto esta tabla se utiliza para asignar a todos los píxeles que pertenecen a una misma región, el valor de una de las etiquetas. El valor asignado es el mayor.

 **Algoritmo basado en la detección de bordes mediante *El Gradiente***

- **Func.**

Una herramienta comúnmente usada para la segmentación mediante la detección de bordes en una imagen f en un punto (x, y) es el gradiente, denotado por $grad(f)$, y definido como un vector bidimensional: [14]

$$grad(f) \equiv \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \quad (1)$$

Siendo un vector perpendicular al borde, donde el vector $grad(f)$ apunta en la dirección de variación máxima de f en el punto (x, y) por unidad de distancia.

La magnitud (longitud) del vector $grad(f)$, denotada como $M(x, y)$, se define como:

$$M(x, y) = mag(\nabla f) = \sqrt{g_x^2 + g_y^2} \quad (2)$$

La cual representa la variación en la dirección del vector ∇f . La dirección del vector ∇f está dada por el ángulo medido con respecto al eje x , y se calcula como:

$$\alpha(x, y) = \tan^{-1} \begin{bmatrix} g_y \\ g_x \end{bmatrix}$$

Para obtener el gradiente de una imagen es necesario calcular las derivadas parciales

$$\frac{\partial f}{\partial x} \text{ y } \frac{\partial f}{\partial y}$$

en cada píxel de la imagen, usando las siguientes ecuaciones:

$$g_x = \frac{\partial f(x, y)}{\partial x} \cong f(x+1, y) - f(x, y) \quad (4)$$

$$g_y = \frac{\partial f(x, y)}{\partial y} \cong f(x, y+1) - f(x, y) \quad (5)$$

- **C.M**

Este algoritmo tiene una complejidad media en términos matemáticos, puesto que es necesario conocimiento todo lo relacionado en cuanto a derivadas y lo relacionado a espacios vectoriales.

- **C.C**

Dicho algoritmo requiere un nivel medio en cuanto a conocimientos de programación, ya que el tema de las derivadas parciales en la programación no es algo trivial.

- **N.g**

Requiere como entrada la imagen en escala de grises

- **Bnz.**

Retorna una vez segmentada, la nueva imagen de manera binarizada.

- **CPU**

Este algoritmo no requiere de gran cantidad de recursos de hardware para su correcto funcionamiento.

 **Algoritmo que combina el *Etiquetado de Regiones* y el uso del *Gradiente***

- **Func**

En un primer paso se aproximan las derivadas parciales en las coordenadas de cada píxel de la imagen, como se muestran en las ecuaciones (4) y (5). Con estas aproximaciones de las derivadas parciales se calcula la magnitud del gradiente.

La magnitud del gradiente será grande en las áreas donde existan cambios abruptos de intensidad y pequeña en aquellas que presenten cambios suaves, estos resultados posteriormente son normalizados por lo que los valores de la magnitud del gradiente en áreas donde existan cambios de intensidad abruptos se elevarán acercándose aún más al nivel de intensidad máximo, lo que facilita la binarización.

El propósito de esta primera binarización es hacer que los bordes detectados en la imagen se vuelvan más notables y faciliten la detección del área de interés.

Enseguida, se utiliza un filtro de promediado con un tamaño de máscara de 19 x 19 píxeles para obtener áreas con una escala de gris uniforme.

La imagen resultante de la última binarización servirá como entrada del algoritmo de etiquetado. Por último, en la imagen filtrada y binarizada se guardará la región que tiene el mayor número de píxeles conectados (etiquetado de regiones) y se desechará el resto de las regiones con menos píxeles. Esta región corresponderá a la región donde se encuentra la información más significativa de una huella dactilar. [15]

- **C.M**

Este algoritmo tiene una complejidad media en términos matemáticos, puesto que es de necesario conocimiento todo lo relacionado en cuanto a derivadas y lo relacionado a espacios vectoriales

- **C.C**

Dicho algoritmo requiere un nivel medio en cuanto a conocimientos de programación, ya que el tema de las derivadas parciales en la programación no es algo trivial.

- **N. g**

Dicho algoritmo lleva la imagen original a escala de grises para en luego realizar una segmentación más eficiente.

- **Bnz.**

Este algoritmo realiza 2 binarizaciones, una inicial luego de haber normalizado la imagen, y una final después de aplicar un filtro de promediado para devolver una imagen segmentada y binarizada.

- **CPU**

Este algoritmo no requiere de gran cantidad de recursos de hardware para su correcto funcionamiento.

- **Operaciones**

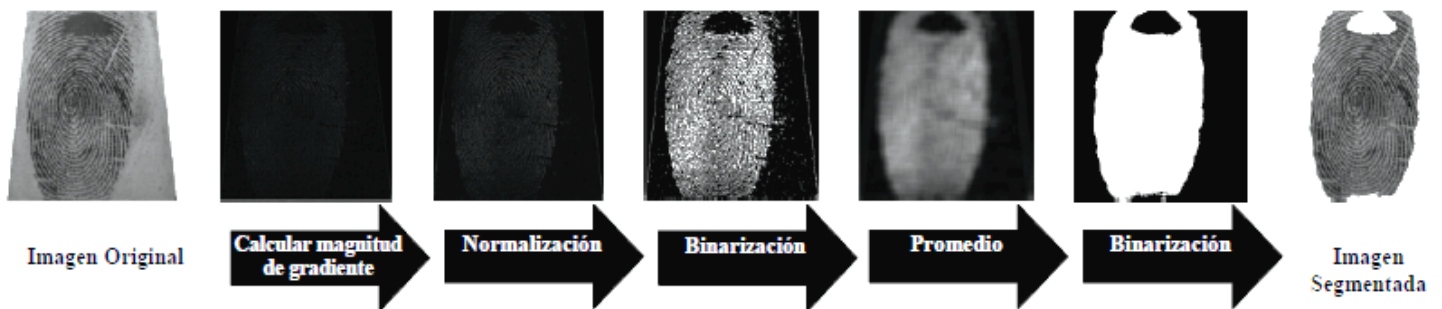


Fig. 3 Proceso de segmentación mediante el Etiquetado de regiones y uso del Gradiente

A continuación se listan a detalle las principales tareas del algoritmo de segmentación analizado.

- 1) Dada una imagen de huella dactilar de tamaño $N \times M$, calcular el gradiente de la imagen filtrándola mediante un filtro Laplaciano y posteriormente la magnitud del gradiente usando la ecuación (2).

- 2) Normalizar los valores de la imagen, para asegurarse de que se encuentren en un rango de 0 a 255. Para esto, es necesario:
 - 2.1. Encontrar el valor mínimo de intensidad en la imagen.
 - 2.2. Restar el valor mínimo encontrado a cada uno de los valores de la imagen.
 - 2.3. Encontrar el valor máximo de intensidad en los nuevos valores de la imagen y dividirlo entre 255.
 - 2.4. Por último, multiplicar cada uno de los valores de la imagen por el resultado obtenido de la división en el paso anterior.
- 3) Binarizar la imagen normalizada, utilizando como valor umbral el cálculo de la media de la imagen.
- 4) Aplicar un filtro de promediado a la imagen binarizada, utilizando una máscara de convolución con coeficientes de valor $1/fc$, donde f y c representan el número de filas y columnas de la máscara respectivamente.
- 5) Binarizar nuevamente la imagen filtrada, obteniendo nuevamente la media de la imagen para que ésta sea usada como umbral.
- 6) Obtener la matriz de etiquetas de la imagen binaria utilizando algún algoritmo de etiquetado.
- 7) Encontrar la etiqueta con el mayor número de ocurrencias en la matriz de etiquetas, esto descartando las pertenencias al fondo.
- 8) Detectar la región más grande y enviar al fondo todas las regiones restantes detectadas, rescatando así sólo la información que pertenece a la huella dactilar.

Algoritmo de *Canny*

- *Func*

Está basado en un proceso de optimización de ciertos objetivos, tales como [4]:

1. Maximizar la relación señal-ruido: mediante un filtrado de la imagen, la cual es suavizada usando un filtro Gaussiano.

2. Minimizar la diferencia borde real y borde detectado: para disminuir la diferencia entre uno y otro, y evitar falsas detecciones, se eliminan los puntos que no sean de máximos locales de la imagen filtrada.
3. Identificar bordes usando un conjunto de píxeles con cierta conectividad. Se construyen dos imágenes binarias a partir de ciertas condiciones y se usan complementariamente para encontrar un borde final.

En la determinación del borde final pueden quedar tramos de bordes abiertos. Para solucionar este problema se usó la media de los píxeles bordes como valor de corte para definir los píxeles señal y fondo.

- **C.M**

Este algoritmo tiene una complejidad media en términos matemáticos, puesto que es de necesario conocimiento todo lo relacionado en cuanto a derivadas y lo relacionado a espacios vectoriales.

- **C.C**

Dicho algoritmo requiere un nivel medio en cuanto a conocimientos de programación, ya que el tema de las derivadas parciales en la programación no es algo trivial.

- **N. g**

No trabaja la imagen en escala de grises.

- **Bnz.**

Este algoritmo binariza con el objetivo de determinar los bordes de la imagen en dependencia de un valor umbral determinado por pruebas experimentales realizadas.

- **CPU**

Este algoritmo no requiere de gran cantidad de recursos de hardware para su correcto funcionamiento.

- **Operaciones**

Sea $f(x, y)$ la intensidad de un píxel que se encuentra en la posición (x, y) de la imagen.

1. Aplicar un filtro Gaussiano a la imagen para suavizarla:

$$g(x, y) = f(x, y) \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-x')^2 + (y-y')^2}{2\sigma^2}} \rightarrow \text{Suavizado de la imagen.}$$

Calcular el gradiente de g para maximizar la diferencia entre el ruido y la señal:

$$\nabla g(x, y) = \nabla [f(x, y)h_\sigma(x, y)] = f(x, y)\nabla h_\sigma(x, y) = f(x, y) \begin{pmatrix} \frac{\partial h_\sigma(x, y)}{\partial x} \\ \frac{\partial h_\sigma(x, y)}{\partial y} \end{pmatrix}$$

2. Se calcula el módulo y el argumento de la imagen suavizada y filtrada.

$$G^{\text{mod}}(x, y) = \sqrt{g_x^2(x, y) + g_y^2(x, y)} \rightarrow \text{Módulo}$$

$$G^{\text{arg}}(x, y) = \tan^{-1} \left[\frac{g_y(x, y)}{g_x(x, y)} \right] \rightarrow \text{Argumento}$$

3. Y entonces en la dirección del gradiente, se eliminan aquellos píxeles que no sean de máximos locales del módulo. Ya que el operador gradiente, aplicado a un píxel (x, y) , devuelve un vector que indica la dirección de máxima variabilidad de la intensidad luminosa y su nivel de variación.
4. Se construyen dos imágenes binarias auxiliares: I_1 e I_2 a partir de dos umbrales $T_1 < T_2$, tales que, con T_1 habrían muchos más píxeles bordes que con T_2 .

$$I_{i, \text{bin}} = \begin{cases} 1, & \text{si } G^{\text{mod}}(x, y) > T_i \\ 0, & \text{si } G^{\text{mod}}(x, y) < T_i \end{cases}$$

Entonces a partir de los píxeles bordes de la imagen I_1 , se van construyendo bordes, mediante la adición de píxeles bordes de la imagen I_2 .

5. Calcular la media (M_b) de los píxeles que definitivamente fueron bordes.

6. Si $f(x, y) > M_b$, entonces el píxel (x, y) pertenece a la señal, si no pertenece al fondo.

Algoritmo de Otsu

- Func

La segmentación según un valor umbral es una técnica de segmentación ampliamente utilizada en aplicaciones biométricas. Se emplea cuando hay una clara diferencia entre los objetos a extraer respecto del fondo de la imagen. Los principios que rigen esta técnica son la similitud entre los píxeles pertenecientes a un objeto y sus diferencias respecto al resto. [16] [17]

La mayoría de las técnicas de segmentación según un valor umbral se basan en estadísticas sobre el histograma unidimensional de una imagen. También se utiliza la matriz de co-ocurrencia de una imagen. Para localizar los umbrales se pueden usar procedimientos paramétricos y no paramétricos. En los paramétricos, la distribución de los niveles de gris de una clase de objeto lleva a encontrar los umbrales. En los procedimientos no paramétricos, los umbrales se obtienen de forma óptima de acuerdo a algún criterio.

En particular, el método de Otsu, elige el umbral óptimo maximizando la varianza entre clases (*between-class variance*) mediante una búsqueda exhaustiva.

La ventaja del método de Otsu radica en que es automático, es decir, no necesita supervisión humana ni información previa de la imagen antes de su procesamiento.

- C.M

Formulación Matemática:

Una imagen es una función bidimensional de la intensidad del nivel de gris, y contiene N píxeles cuyos niveles de gris se encuentran entre 1 y L . El número de píxeles con nivel de gris i se denota como f_i , y la probabilidad de ocurrencia del nivel de gris i en la imagen está dada por

$$p_i = \frac{f_i}{N}$$

En el caso de la segmentación según un valor umbral en dos niveles de una imagen (a veces llamada binarización), los píxeles son divididos en dos clases C_1 , con niveles de gris $[1, \dots, t]$ y C_2 , con niveles de gris $[t+1, \dots, L]$. Entonces, las distribuciones de probabilidades de los niveles de gris para las dos clases son:

$$C_1 : \frac{p_1}{\omega_1(t)}, \dots, \frac{p_t}{\omega_1(t)} \quad (2)$$

$$C_2 : \frac{p_{t+1}}{\omega_2(t)}, \frac{p_{t+2}}{\omega_2(t)}, \dots, \frac{p_L}{\omega_2(t)} \quad (3)$$

Donde:

$$\omega_1(t) = \sum_{i=1}^t p_i \quad \omega_2(t) = \sum_{i=t+1}^L p_i$$

También, la media para la clase C1 y la clase C2 es

$$\mu_1 = \sum_{i=1}^t \frac{i \cdot p_i}{\omega_1(t)} \quad \mu_2 = \sum_{i=t+1}^L \frac{i \cdot p_i}{\omega_2(t)}$$

Sea μ_T la intensidad media de toda la imagen. Es muy evidente entonces que

$$\omega_1 \cdot \mu_1 + \omega_2 \cdot \mu_2 = \mu_T \quad \omega_1 + \omega_2 = 1$$

Usando análisis discriminante, Otsu definió la variancia entre clases de una imagen umbralizada como:

$$\sigma_B^2 = \omega_1 \cdot (\mu_1 - \mu_T)^2 + \omega_2 \cdot (\mu_2 - \mu_T)^2$$

Para una segmentación según un valor umbral de dos niveles, Otsu verificó que el umbral óptimo t^* se elige de manera que σ_B^2 sea máxima; esto es:

$$t^* = \underset{t}{\text{Max}} \{ \sigma_B^2(t) \} \quad 1 \leq t \leq L$$

- **C.C**

Este algoritmo no requiere de una programación avanzada, ya que su implementación es muy sencilla, aunque robusta. Brindando esto un aspecto fundamental a tener en cuenta a la hora de decidir qué algoritmo diseñar para nuestro componente.

- **N. g**

Dicho algoritmo lleva la imagen original a escala de grises para realizar una segmentación más eficiente.

- **Bnz.**

Este algoritmo binariza en correspondencia al valor umbral determinado por el método de Otsu.

- **CPU**

Este algoritmo no requiere de gran cantidad de recursos de hardware para su correcto funcionamiento.

- **Operaciones**

Estos son los pasos del algoritmo:

Para cada umbral T, se tiene que:

1. Separar los píxeles en dos grupos de acuerdo con el valor umbral calculado.
2. Hallar la media de cada grupo.
3. Cuadrar las diferencias entre las medias.
4. Multiplicar la cantidad de píxeles de un clúster por la cantidad de píxeles en otro.

Si se tratase de comprender este algoritmo de una forma matemática más simple, sería algo así:

1. Calcular el histograma y las probabilidades de cada nivel de intensidad.
2. Configurar $q_i(0)$ inicial y $\mu_i(0)$.
3. Pasar por todos los mínimos posibles de intensidad máxima.
4. Pasar a través de los posibles valores de umbral con mayor intensidad.
5. Actualizar los valores de q_i y μ_i .

6. Calcular $\sigma_b^2(t)$.

7. Los umbrales deseados corresponden al máximo.

Desde que Otsu opera en los histogramas, es muy conveniente analizar el histograma de la imagen y de la decisión del valor de umbral. El código es muy simple de usar. El método de Otsu funciona sobre imágenes en escala de grises. Entonces primeramente, se tiene que transformar la imagen a escala de grises. Después se obtiene el valor umbral de Otsu resolviéndole por t máxima. Finalmente se segmenta la imagen usando este valor umbral t , es decir, lo que esté por encima de ese valor umbral formará parte de la región de interés, de lo contrario se tomará como parte del fondo.

2.1.3. Justificación de la elección

Tomando como referencia el estudio comparativo realizado sobre los algoritmos y métodos, se concluye utilizar el diseñado por Otsu. El argumento principal de la selección tiene su fundamento en el valor umbral, puesto que los otros algoritmos seleccionan este valor umbral, basándose en resultados experimentales, es decir, se van haciendo pruebas de segmentación de imágenes con valores de umbral diferentes, para así en dependencia de los resultados de segmentación obtenidos, seleccionar un umbral ideal.

Si bien hay diferentes métodos y estrategias para determinar un valor umbral, la mayoría de ellos no dan buenos resultados cuando se trabaja con imágenes del mundo real debido a la presencia de ruido, histogramas planos o una iluminación inadecuada. Por el contrario, el método de Otsu fue uno de los mejores métodos de selección de umbral para imágenes del mundo real.

El algoritmo de Otsu, resuelve el problema de la segmentación de imágenes de huellas dactilares de manera eficiente, debido a que el mismo se encarga de obtener el valor umbral mediante una formulación matemática y así mismo en dependencia de este segmentar la imagen mediante el método de segmentación según un valor umbral, siguiendo los pasos descritos anteriormente. Además realiza la segmentación sin consumir muchos recursos de hardware y su implementación es de fácil entendimiento, permitiendo así realizarle las modificaciones deseadas a este algoritmo.

2.2. Implementación del componente

Una vez seleccionado el algoritmo diseñado por Otsu se procede a su implementación obteniéndose como resultado uno de los componentes que formarán parte del pre-procesamiento de imágenes en el sistema de verificación dactilar que se desarrolla en el CISED.

Para la implementación se utilizó C# como lenguaje de programación tomando como IDE a utilizar Microsoft Visual Studio 2013. Requiriendo la colaboración de las librerías *Drawing.Imaging.dll* y *MathNet.Iridum.dll*, las cuales permiten el uso de funcionalidades internas de la plataforma .NET para el tratamiento de imágenes, así como funcionalidades matemáticas requeridas por el algoritmo diseñado por Otsu.

Las funcionalidades definidas para la implementación del algoritmo son:

- Cargar Imagen: permite cargar imágenes de huellas dactilares en cualquier formato de imagen.
- Segmentar: realiza los pasos a seguir para la segmentación de la imagen según el algoritmo de Otsu.
- Guardar Imagen: permite almacenar la nueva imagen una vez segmentada.

Además se le agregó la funcionalidad para medir el tiempo de procesamiento del algoritmo facilitando esto determinar la eficiencia del mismo.

La interfaz del componente desarrollado fue diseñada con el fin de realizar las pruebas pertinentes para verificar el correcto funcionamiento del algoritmo de Otsu, dicha interfaz presenta un diseño simple y amigable. (Ver [Interfaz de prueba](#))

2.2.1. Estándar de programación

Para la implementación del algoritmo fue necesario establecer nomenclaturas o estándares que fuesen entendibles por parte del desarrollador.

Un estándar de programación permite definir la escritura y organización del código fuente de un programa, facilitándole al programador la modificación de su propio código fuente; además de definir la forma en que deben ser declaradas las variables, las clases y los comentarios. A continuación se define el estándar de codificación utilizado para implementar el algoritmo de Otsu.

- Declaraciones

Como primer elemento se tiene la declaración de variables, y el número de declaraciones que se realizan sobre una misma línea.

- Por ello cada variable será declarada en líneas distintas. Ejemplo: *private byte p;*
- Los arreglos se deben declarar: *private int [] histograma;*

La inicialización de las variables cuando se declaran, sólo se efectuará si su valor inicial depende de algún cálculo. Ejemplo:

- El valor inicial no depende de un cálculo: *private byte p;*
- El valor inicial depende de un cálculo: *private byte p= bmData.Scan0.ToPointer ();*

- Sentencias

Sentencias simples: cada línea debe contener una sola sentencia.

Sentencias compuestas: son las que están entre las llaves “{” y “}” de forma anidada. Estas deben estar a cuatro espacios de la instrucción que la contiene.

Todas las sentencias del tipo *if, for, while, do... while* deben tener llaves, aunque sólo contengan una línea de código, de esta forma se evita la introducción accidental de errores, si se añaden posteriormente otras instrucciones.

Las normas genéricas para la asignación de nombres son:

1. Se usan descriptores en inglés que precisan el cometido de la variable o método.
2. Se usan terminologías aplicables al dominio.
3. Se debe evitar en lo posible los nombres largos (menos de 15 letras sería lo ideal), a menos que esto traiga consigo que no se cumpla la norma 1.

4. Evitar nombres que difieran en una letra o en el uso de mayúsculas.
5. Un nombre no debería contar con más de tres palabras.
6. No usar siglas en los nombres a menos que queden muy largos o sean siglas conocidas por todos.
7. Variables, atributos y métodos:
 - Variables: deben comenzar con minúscula.
 - Atributos: cada letra inicial de las palabras que conformen el nombre del atributo será en mayúsculas.
 - Métodos: la primera letra en mayúscula, si no es un método contenido dentro de otro, en este caso debe ser con minúscula y el resto de las palabras empiezan con mayúsculas.

Nota: no se utilizara en ningún caso el caracter “_” delante o entre dos palabras que conformen un nombre de método, variable o atributo.

Conclusiones parciales

Se analizaron los distintos algoritmos para la segmentación de imágenes de huellas dactilares. Se seleccionó el diseñado por Otsu teniendo en cuenta sus características y ventajas propias para darle cumplimiento al objetivo propuesto. Se implementó dicho algoritmo, obteniéndose el componente para la segmentación de imágenes de huellas dactilares.

Capítulo 3 “Prueba y Validación del componente”

Este capítulo abarca los temas relacionados con las pruebas y la validación de los resultados obtenidos con el desarrollo del componente. Se describen los casos de prueba según la metodología XP para validar los resultados.

3.1. Pruebas

Las pruebas de software constituyen una actividad en la cual un sistema o componente es ejecutado bajo condiciones específicas, donde se observan o almacenan los resultados y se ejecuta una evaluación de algún aspecto del sistema o componente. [9]

La metodología XP divide las pruebas en dos grupos:

- Pruebas unitarias: son pruebas desarrolladas por los programadores durante todo el ciclo de implementación y su función es verificar el código de forma automática.
- Pruebas de aceptación: están destinadas a evaluar si al final de una iteración se obtuvo la funcionalidad requerida, además de comprobar que dicha funcionalidad sea la esperada por el cliente.

En la evaluación del funcionamiento del componente se aplicaron las pruebas antes descritas, las unitarias para asegurar el correcto funcionamiento del código; y las pruebas de aceptación, evidenciadas mediante la elaboración de casos de prueba, realizando los mismos a través de resultados experimentales.

3.2. Validación de los resultados

En este epígrafe se muestran los resultados que se obtuvieron con el desarrollo del componente. Todas las pruebas se realizaron sobre una computadora personal con un procesador Pentium 4 a una frecuencia de 1.6GHz, 1 GB de memoria RAM DDR3 y tarjeta gráfica AMD Radeon HD 6320 con 256 MB de RAM para video. Una vez realizadas las distintas pruebas se concluyó que el algoritmo diseñado por Otsu aporta resultados satisfactorios

Esta afirmación tiene basamento en que el valor umbral obtenido mediante el algoritmo implementado no es el mismo para cada imagen; el tiempo promedio de ejecución fue de 0.44 segundos aproximadamente, evidenciando esto que el tiempo de respuesta siempre estará por

debajo de 1 segundo (tiempo máximo de ejecución), confirmando lo anteriormente expuesto el correcto funcionamiento de dicho algoritmo.

3.2.1. Resultados Experimentales

Para validar los resultados se emplearon imágenes de la base de datos del Fingerprint Verification Competition – FVC2002. En base al criterio de selección descrito se escogieron 40 imágenes en formato *tif*, de tamaño 300 x 300, 256 x 364, 448 x 478 y 240 x 320.



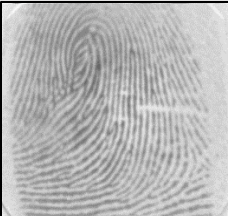








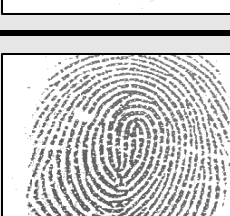
<i>Imagen original</i>	<i>Imagen segmentada</i>	<i>Valor umbral</i>	<i>Tiempo (segundos)</i>
		182	0.40
		183	0.30
		98	0.60
		109	0.61
		181	0.39
		164	0.38

Tabla 1 Resultados experimentales

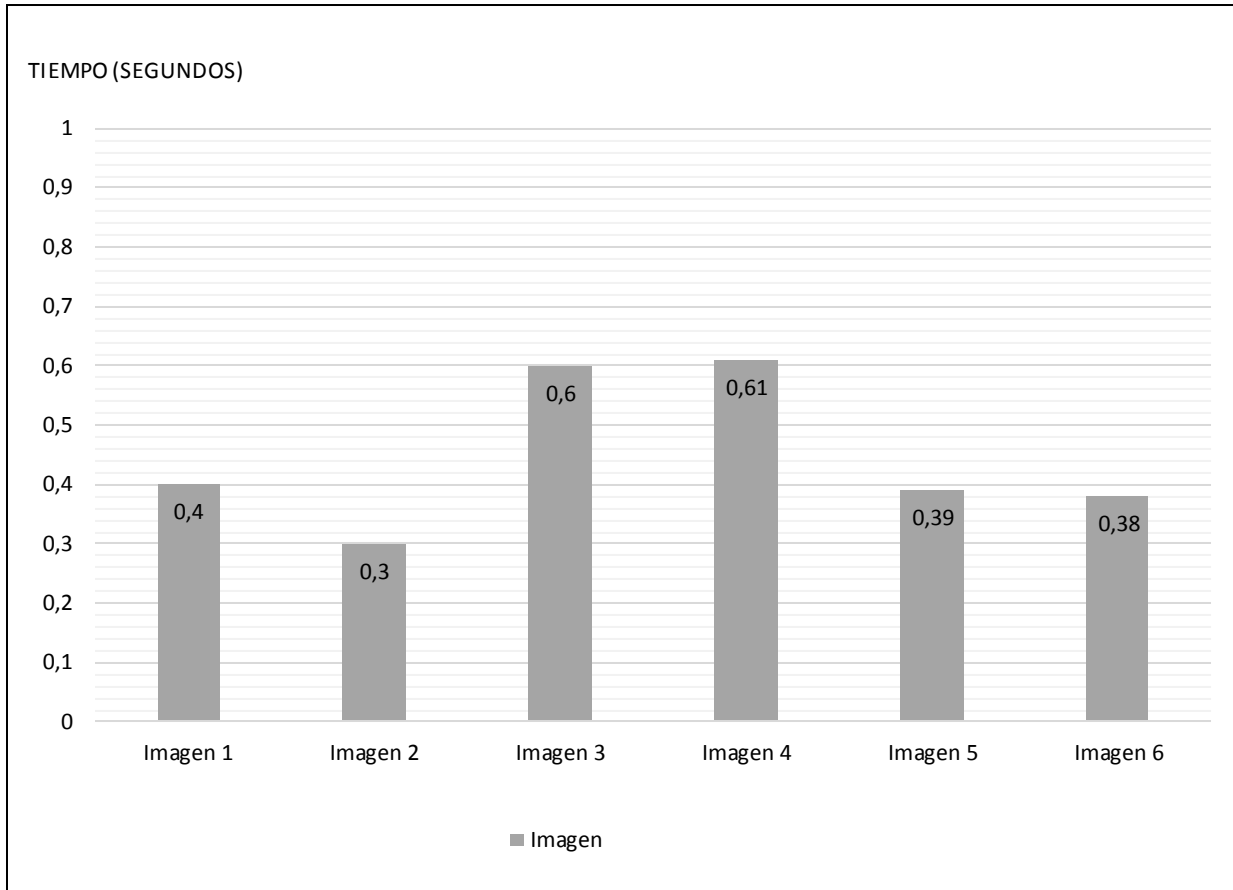


Tabla 2 Tiempo de ejecución del algoritmo

3.3. Conclusiones parciales

Al finalizar este capítulo, el desarrollo guiado por pruebas de aceptación arrojó resultados exitosos demostrando la eficiencia y calidad del componente obtenido, además de la satisfacción por parte del cliente.

Conclusiones

Producto a la investigación realizada y como conclusiones generales de la misma, se muestran alcanzados los objetivos propuestos satisfactoriamente:

- ✓ Se hizo un análisis profundo de los distintos algoritmos de segmentación de imágenes existentes, para luego determinar un criterio común que permitió la selección de los más significativos en el área de los sistemas biométricos.
- ✓ Se propuso seleccionar e implementar el algoritmo de segmentación diseñado por Otsu, teniendo en cuenta sus ventajas frente a otros algoritmos para la segmentación de imágenes de huellas dactilares estudiados.
- ✓ Teniendo en cuenta los valores definidos como aceptables por parte del cliente se realizaron pruebas para dictaminar el funcionamiento del algoritmo.

Recomendaciones

A partir de la investigación realizada se recomienda:

- ✓ La optimización del algoritmo propuesto sin sacrificar la eficiencia del mismo de forma considerable.
- ✓ El estudio de algoritmos diseñados recientemente, con el objetivo de implementar alguno estudiado que combine el propuesto en esta investigación con otro de los ya existentes.

Referencias Bibliográficas

1. C. Gonzalez, Rafael y E. Woods, Richard. *Digital Image Processing*. New Jersey : Prentice Hall, 2002.
2. IRM Press. *Advances in Image Segmentation*. Hershey, USA : s.n., 2006. ISBN 1-59140-755-9.
3. Tsinghua University. *Image Segmentation Evaluation in the Last 40 Years*. China, Beijing : s.n., 2009.
4. Rajeshwar, Dass y Priyanka, Swapna Devi. *Image Segmentation Techniques*. 2012.
5. Evaluation and comparison of different segmentation algorithm. *Pattern Recognition*. 1997.
6. IEEE. *A review of recent evaluation methods for image segmentation. Proceedings of the 6th International Symposium on Signal Processing and Its Applications*. Malaysia : s.n., 2001.
7. A survey on evaluation methods for image segmentation. *Pattern Recognition*. 1996.
8. Pérez González, Rodrigo, Carrillo Pérez, Isaías y Rodríguez Martín, Aureliano David. *Metodología de Desarrollo del Software*. 2008.
9. Patricio Letelier, M^a Carmen Penadés y Canó, José H. *Metodologías ágiles para el desarrollo de software: Extreme Programming (XP)*. Universidad Politécnica de Valencia : s.n.
10. Procesos de Software-Methodology Extreme Programming (XP). [En línea] [Citado el: 20 de marzo de 2013.] <http://procesosdesoftware.com/METODOLOGIA+XP>.
11. Carlos Reynoso, Nicolás Kicillof. *Estilos y Patrones en la Estrategia de Arquitectura de Microsoft*. Argentina: Universidad de Buenos Aires : s.n., 2004.
12. Jacobson, Ivar, Booch, Grady y Rumbaugh, James. *PROCESO UNIFICADO DE DESARROLLO DE SOFTWARE*. Madrid : Addison Wesley Longman Inc, 2000. ISBN 84-7829-036-2.
13. Comunidad de software libre Paraguay - Framework de Desarrollo. [En línea] [Citado el: 24 de marzo de 2014.] <http://www.pti.org.py/csl/index.php>.
14. ALba, José Luis, Cid, Jesús y Mora, Inmaculada. *Análisis de algoritmos para la segmentación de imágenes*. Madrid, España : s.n., 2006, marzo.
15. *Una estrategia de segmentación de imágenes digitales de huellas dactilares latentes*. E., Ruíz E. María, Miguel, Morales S. y Yahir, Hernández M. 2011.
16. Zhang-Hui. *Meta-Evaluation of Image Segmentation Using Machine Learning*. Washington, USA : SPIE Electronic Imaging-Storage and Retrieval Methods and Applications for Multimedia, 2005.
17. Telgadá, Rupali, Siddiqui, Almas M. N. y Deshmukh, Prapti D. *Fingerprint Image Segmentation using Global Thresholding*. 2014.

Glosario de Términos

A

Algoritmo: Es una lista que, dado un estado inicial y una entrada, propone pasos sucesivos para arribar a un estado final obteniendo una solución.

AFIS (Automated Fingerprint Identification System): Sistema de identificación automatizado de huellas dactilares que compara un registro de huellas dactilares con los registros de una base de datos para determinar la identidad de un individuo.

B

Borde: En una imagen, es el lugar en el que coinciden dos formas. La línea que separa dos formas, o una forma y un espacio.

C

Contorno: Conjunto de líneas que limitan una figura o composición.

CISED (Centro de Identificación y Seguridad Digital): Centro de desarrollo de soluciones integrales, productos y servicios en el campo de la identificación y la seguridad digital. Se encuentra estrechamente vinculado a la empresa comercializadora Albet SA y a la Universidad de las Ciencias Informáticas, lo que le ofrece una amplia posibilidad a su capital humano para la investigación, desarrollo e innovación. Cuenta además, con un equipo de especialistas de alto nivel, con gran experiencia y reconocimiento a nivel nacional e internacional.

G

Gradiente: Denota una dirección en el espacio según la cual se aprecia una variación de una determinada propiedad o magnitud física.

H

Hardware: Componentes físicos de una computadora o de una red (a diferencia de los programas o elementos lógicos que los hacen funcionar).

I

Imagen: Figura, representación, semejanza y apariencia de algo. En computación es formada por la unión de M x N píxeles (imagen bidimensional) o vóxeles (imagen tridimensional).

P

Píxel: Abreviatura de “*picture element*”. Es la mínima unidad de información dentro de una imagen bidimensional.

R

RAM: Es la memoria de acceso aleatorio (en inglés: *random access memory*), desde donde el procesador recibe las instrucciones y guarda los resultados. Es el área de trabajo para la mayor parte del software de un computador.

S

Segmentación: Se utiliza en el Procesamiento de Imágenes para el reconocimiento de objetos o estructuras de interés en la imagen.

Anexos

Anexo 1. Diferencias entre las metodologías ágiles y tradicionales.

Metodologías Ágiles	Metodologías Tradicionales
Basadas en heurísticas provenientes de prácticas de producción de código	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo
Especialmente preparados para cambios durante el proyecto	Cierta resistencia a los cambios
Proceso menos controlado, con pocos principios	Proceso mucho más controlado, con numerosas políticas/normas
No existe contrato tradicional o al menos es bastante flexible	Existe un contrato prefijado
El cliente es parte del equipo de desarrollo	El cliente interactúa con el equipo de desarrollo mediante reuniones
Grupos pequeños (<10 integrantes) y trabajando en el mismo sitio	Grupos grandes y posiblemente distribuidos
Pocos roles	Más roles
Menos énfasis en la arquitectura del <i>software</i>	La arquitectura del <i>software</i> es esencial y se expresa mediante modelos

Anexo 2 Interfaz de prueba



Ilustración 1 Vista Inicial

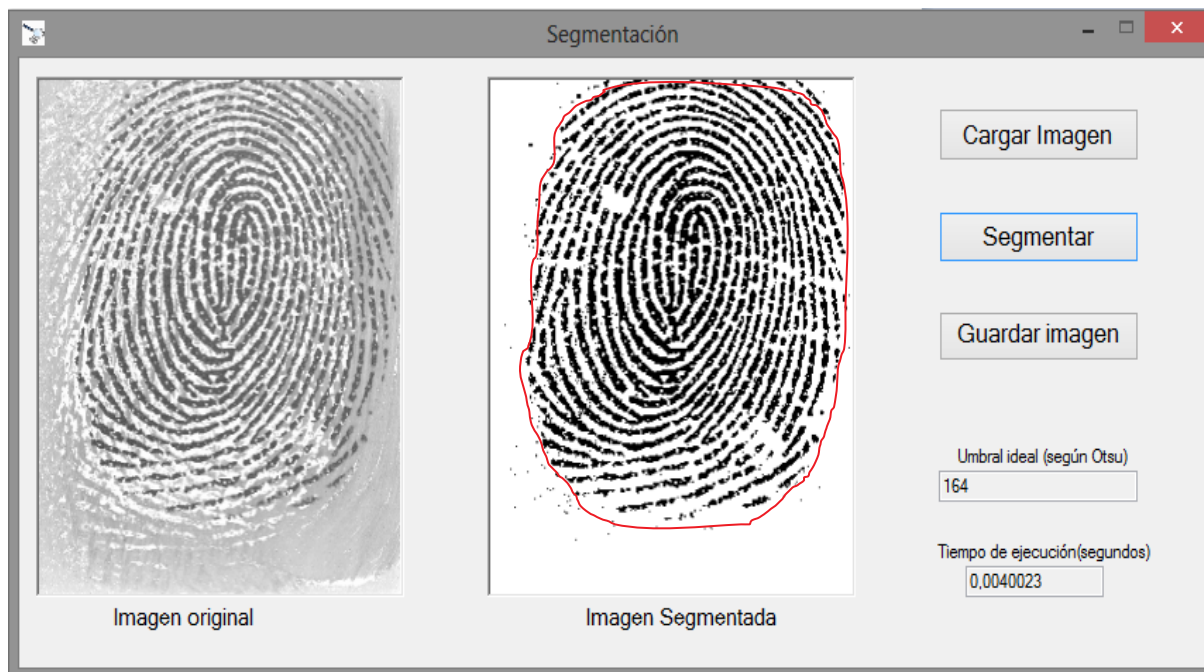


Ilustración 2 Vista que se obtiene al realizar la segmentación