31 DE MAYO DE 2016

PROYECTO BIOMETRIA

HUELLA DACTILAR

JUAN PEDRO TORRES MUÑOZ

1 TABLA DE CONTENIDO

2 INTRODUCIÓN

El proyecto propuesto para el desarrollo durante la mitad del semestre dedicado a biometría consiste en una aplicación que pueda realizar el tratamiento de una huella dactilar desde el momento de la captación hasta comprobar si dicha huella se encuentra presente en el sistema o si no existe. Dándonos también la oportunidad de añadir huellas al sistema para poder ampliar el número de huellas autorizadas. Todo esto se ha desarrollado en java y sin el uso de ninguna API externa como puede ser OpenCV.

El proyecto ha sido desarrollado en solitario, por lo que no se ha podido dejar totalmente terminado.

3 OBJETIVOS

Los objetivos detallados del proyecto, coinciden con el proceso de trabajo con huellas dactilares estudiado en clase, el cual es el siguiente:

- Obtención de la imagen de nuestro dispositivo.
- Convertir dicha imagen en escala de grises.
- Ecualización de la imagen.
- Umbralizar la imagen.
- Filtros binarios eliminación de ruido.
- Adelgazar la imagen.
- Obtención de las minucias.
- Almacenamiento de las mismas.
- Identificación de la huella en base a las minucias.

Cada aspecto de los anteriormente descritos será explicado en detalle ahora.

3.1 OBTENCIÓN DE LA IMAGEN

Ya que no disponíamos del hardware necesario para la captación de las huellas dactilares, nos hemos valido de una base de datos de imágenes de huellas, por lo que esta fase ha sido bastante pasada por alto ya que nuestra aplicación se ha dedicado a cargar desde las imágenes descargadas de dichas bases de datos.

3.2 Conversión de la imagen en escala de grises

Una vez cargamos la imagen en la aplicación debía pasarse a escala de grises para poder trabajar con ella, lo cual es necesario debido al aumento en la carga computacional que tiene al trabajar con imágenes en color. Este punto no supone una gran dificultad, debido a que la mayoría de imágenes descargadas de dichas bases de datos vienen ya en blanco y negro, aunque algunas en calidades bastante deplorables.

3.3 Ecualización

Consiste en realizar una distribución uniforme de los tonos de grises que están presentes en la imagen de la huella. El resultado de aplicar esta operación sobre una imagen es que dicha imagen mejora su contraste.

3.4 Umbralizar

Debido a que los algoritmos posteriores no trabajan con niveles de grises este paso es obligatorio. Este proceso se encarga de transformar una imagen en escala de gris en una imagen en blanco y negro estricto, para ello tomará un valor umbral y en relación a él transformará en blanco o negro.

3.5 Filtros Binarios

Las imágenes presentes en la librería de imágenes que se ha utilizado tienen una calidad bastante mejorable, por esto mismo gracias a estos filtros de vecindad, con la imagen una vez binarizada podemos aplicar dos filtros binarios para eliminar ruido.

El primero un filtro de relleno encargado de rellenar los pixeles vacíos en mitad de una cresta de la huella. El segundo filtro cumple la función de eliminar pixeles sueltos sin información en la imagen.

Como ya se ha dicho si las imágenes tienen buena calidad no es necesario utilizarlos.

3.6 Adelgazamiento o Thinning

Esta etapa del proceso consiste en convertir la huella dactilar, ya binarizada, en una huella con líneas de un pixel de ancho. Para ello utilizamos un algoritmo de erosión que se encargará de ello.

3.7 Obtención de minucias

La obtención de minucias consistía en detectar dichas minucias en la imagen ya adelgaza, para ello al principio se utilizó el algoritmo visto en clase, pero como generaba falsos positivos en medio de una huella, se buscó una alternativa hasta encontrar un artículo en el cual explicaban como solucionar dicho problema utilizando una técnica llamada “Crossing Number”.

3.8 Almacenamiento de minucias

Debido a la falta de tiempo para poder realizar esto de una forma adecuada, actualmente guarda la huella con la que vamos a comparar en un fichero, pero el objetivo deseable y el que se había propuesto al principio era el de almacenarlas en una base de datos sin importar si buena SQL o NoSQL.

3.9 Identificación o Matching

Esta sin duda ha sido la tarea que ha significado el mayor reto, pues consiste en identificar la huella y comprobar si es la que dice ser basándose en las minucias y en los datos almacenados. La dificultad radica en el método para realizarlo, ya que toda la información sobre este tema está en artículos académicos y con notación matemática. Al final se ha conseguido desarrollar un método de verificación 1:1 que no funciona muy mal en las pruebas realizadas. Volveremos sobre ello posteriormente.

4 Descripción del proyecto

El proyecto, en parte descrito anteriormente consistía en conseguir desarrollar la aplicación explicada en la sección de objetivos, para ello se ha ido desarrollando de manera incremental desde una Interfaz de Usuario esqueleto con unos botones muy simples hasta la versión final, la cual aunque tiene una interfaz sencilla, todo está identificado.

4.1 Tareas a realizar

Primeramente se creó una aplicación que cargaba imágenes y las transformaba en escala de grises, para posteriormente mostrarlas en la interfaz.

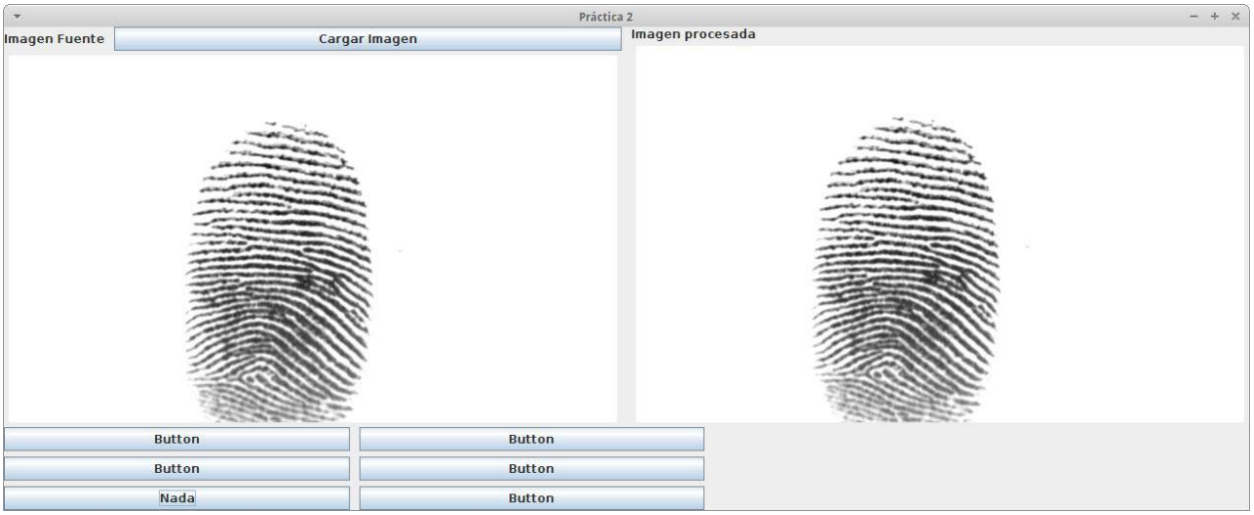
Una vez esa fase estuvo realizada y con el debido control de errores, se procedió a realizar las funciones encargadas de umbralizar y de binarizar la imagen, pudiendo mostrarlas ya que esa funcionalidad ya estaba funcionando.

Una vez realizado con éxito esa parte se realizaron los filtros binarios, los cuales estructuralmente son iguales, pero con un ligero cambio en la operación que realiza sobre el pixel.

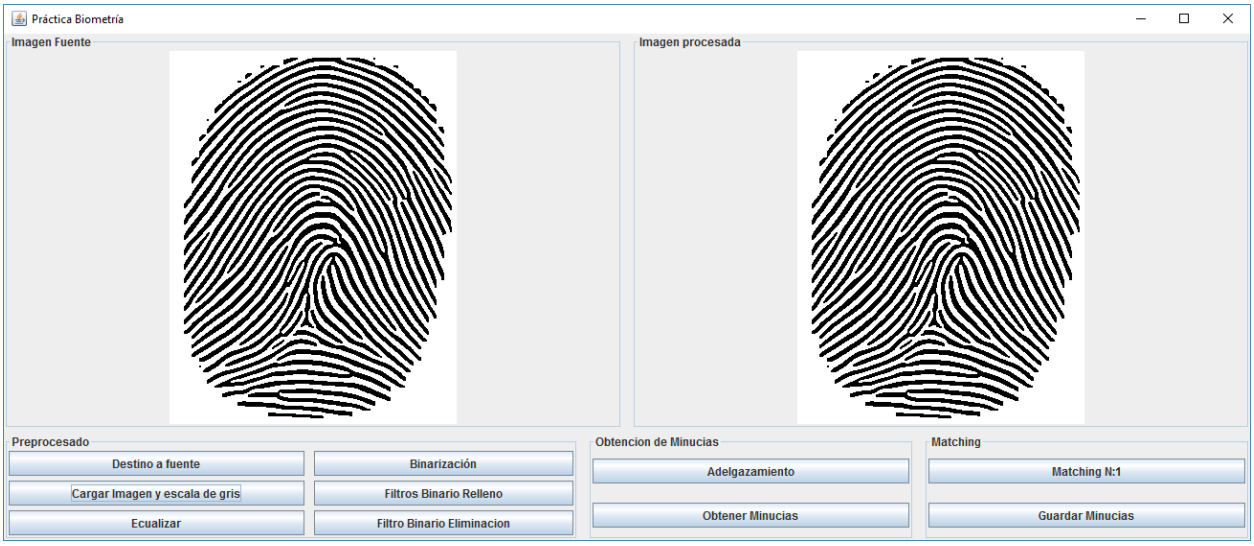
A partir de aquí el proceso se fue complicando poco a poco ya que el algoritmo de adelgazamiento se tuvo que buscar en internet ya que no era trivial realizarlo. La obtención de minucias hubo que buscarlos entre artículos ya que el trivial generaba falsos positivos cuando las crestas de la huella hacían “escaleras”. Una vez obtenidos el matching fue extremadamente difícil de realizar, ya que la documentación encontrada sobre ello tenía un nivel técnico muy elevado, por lo que se optó por un algoritmo basado en triángulos, que se describirá posteriormente.

5 INTERFAZ DE USUARIO

La interfaz de usuario ha ido cambiado desde la primera versión presente aquí:



Como podemos ver era una interfaz realmente básica con los botones para cargar la imagen y para tratarla con cada uno. A continuación veremos la versión final de la interfaz tras el proceso de desarrollo.

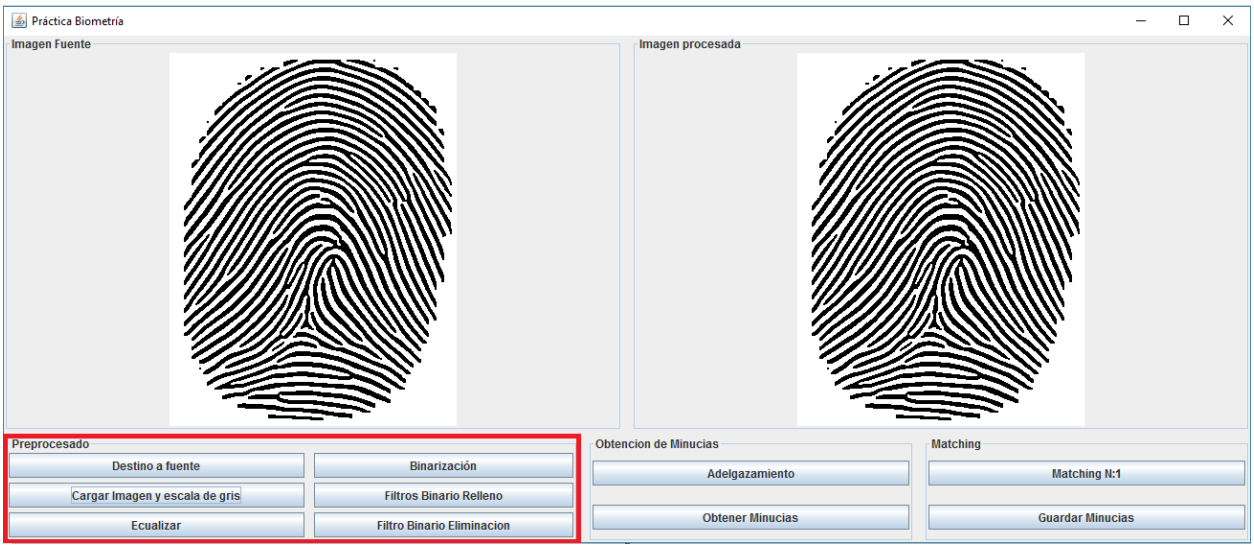


Podemos ver que la interfaz es mucho más explicativa, pero que mantiene el mismo estilo simple.

5.1 GUÍA DE USO

La interfaz es realmente simple, basándonos en la interfaz mostrada anteriormente, al cargar una imagen el programa ya la transformará en escala de grises automáticamente para reducir así la probabilidad de error en los métodos siguientes.

Si queremos cargar una imagen simplemente tenemos que pulsar el botón “Cargar imagen y escala de grises”.



Cada botón tiene una utilidad identificada en su nombre, los describiremos a continuación.

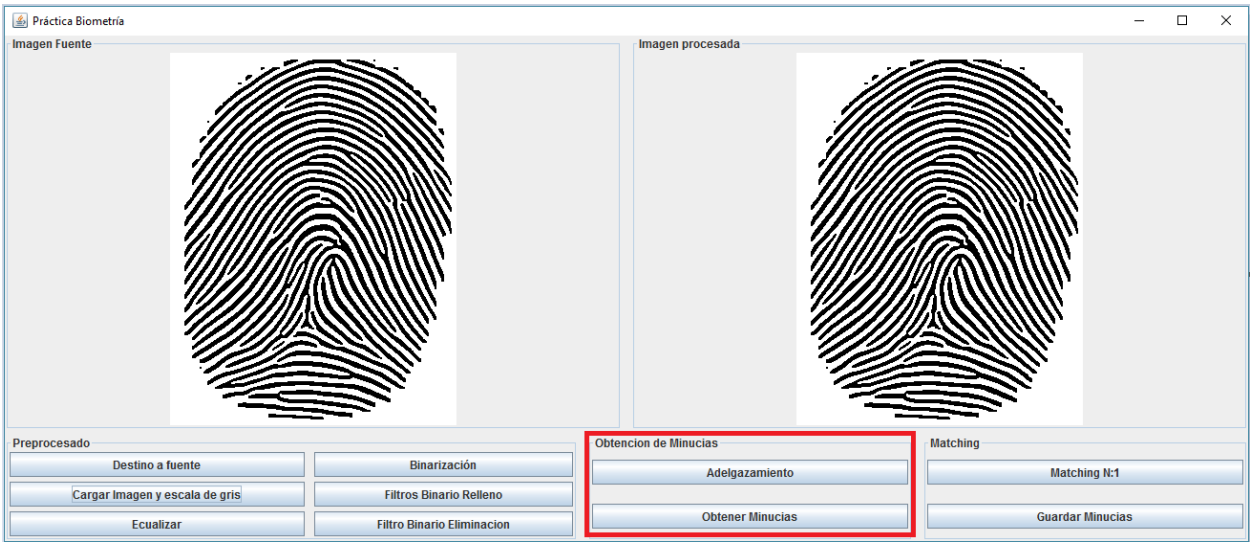
El botón “Ecualizar” una vez pulsado ecualizará la imagen de la izquierda y la mostrará en el panel de la derecha.

El botón “Binarización” una vez pulsado se abrirá una ventana de dialogo que te preguntará el valor umbral sobre el que se va a trabajar.

Una vez descritos estos botones nos quedan tres por describir del panel de “Preprocesado” dos de ellos funcionan igual, que son los llamados “Filtros Binarios” estos filtros trabajan la primera vez sobre la imagen de la izquierda y las sucesivas pulsaciones de cualquiera de los dos filtros sobre la de la derecha. Esto es así para poder observar el cambio entre las diferentes iteraciones de dichos filtros.

El ultimo botón de este panel es el más importante, el botón “Destino a Fuente” es el encargado de cargar la imagen procesada en la imagen fuente, ya que debido al funcionamiento de la aplicación trabaja sobre la imagen fuente (salvo la excepción anterior), por lo que después de realizar una operación si queremos usar los resultados para realizar otra operación tenemos que usar dicho botón.

Ahora se explicarán los botones del panel “Obtención de minucias”.

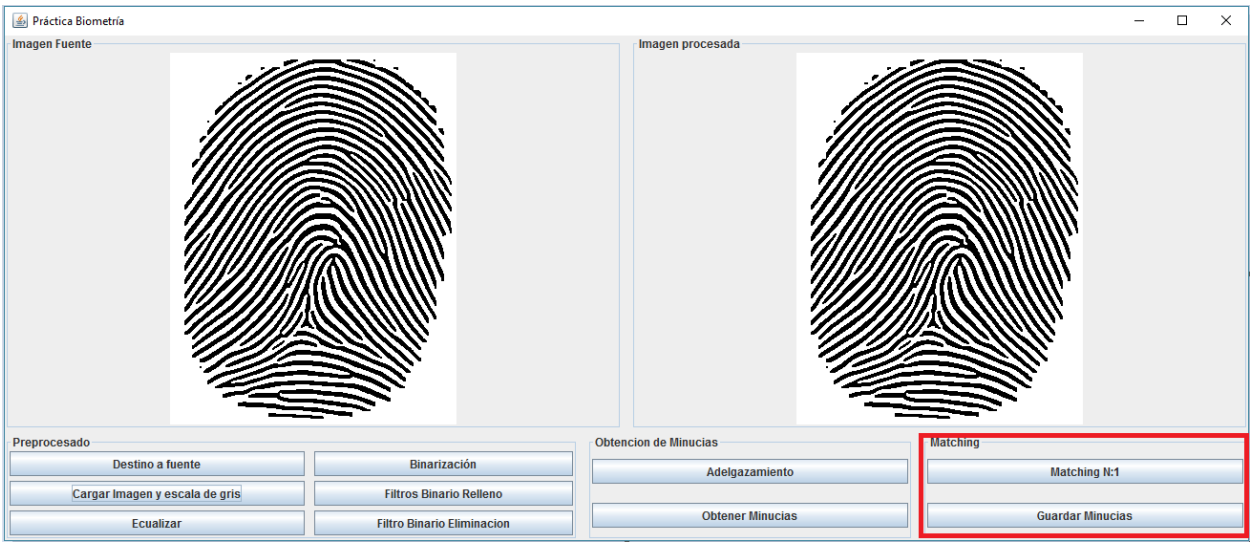


Podemos ver que solo tenemos dos botones, el primero “Adelgazamiento” y el segundo “Obtener minucias”

El primer botón se encarga de, teniendo en la ventana fuente una imagen binarizada, adelgazarla hasta dejarla de un pixel de ancho las crestas de la huella. Una vez realizado esto podemos volver a colocar el resultado en la fuente para proceder a usar el otro botón.

El botón de “Obtener minucias” cuyo funcionamiento es muy sencillo, una vez tengamos una imagen adelgazada en la parte de imagen fuente, al pulsarlo nos pedirá el valor de las ventanas laterales para así eliminar las falsas minucias que salen por el corte de la huella, el margen superior y el margen inferior, dichos márgenes serán como minino de un valor de 10 pixeles. Una vez terminado en la imagen destino se podrán ver las minucias encontradas en la imagen, de color rojo las de tipo bifurcación y de color azul las de tipo final.

Ahora se explicará el último panel de botones, “Matching”.



El primer botón que nos encontramos es “Matching N:1”, como su nombre indica realiza un matching de una imagen, en este caso destino ya que así podemos modificarla sin tener que repetir todo el proceso para hacer pruebas con diferentes ventanas, y los datos de una huella guardados en el archivo “Huella.ser”, ya que por falta de tiempo y de medios no ha sido posible desplegar una base de datos que almacene las huellas y poder realizar la comprobación 1:N, pero hubiese sido una tarea sencilla con los medios necesarios y un poco más de tiempo.

Por otro lado tenemos el botón “Guardar Minucias”, que es el encargado de una vez tengamos las minucias en la pantalla de la derecha, guardarlas en un fichero llamado “Huellas.ser” que será el utilizado para la verificación de las huellas. En un principio este botón estaba destinado a registrar en la base de datos las características de la huella, pero debido a que no hay se ha optado por esta solución.

6 MANUAL DEL PROGRAMADOR

En esta sección se explicarán detalladamente la estructura del proyecto desde el punto de vista del programar, los métodos más significativos y los algoritmos más representativos.

6.1 MÓDULOS

Los métodos más significativos para este proyecto han sido varios, los cuales sirven de soporte para los algoritmos que se explicarán más tarde.

Entre ellos tenemos el que más problemas ha dado, que ha sido el que se encarga del matching de las huellas con la huella almacena en el sistema, del cual hablaremos en la sección de algoritmos.

Otro método importante en el proyecto ha sido el encargado de convertir una matriz de bytes a un objeto de la clase BufferedImage para poder mostrarla así por pantalla sin necesidad de realizar ninguna conversión extraña.

El método encargado de generar y pintar las minucias es otro método bastante significativo del proyecto ya que a la vez que identificarlas, generarlas y almacenarlas en una estructura de datos contenedora, pinta las minucias de diferente color dependiendo del tipo al que pertenezcan y todo esto sin afectar en nada al funcionamiento del sistema.

Por lo demás, durante toda la realización del proyecto se ha buscado que los métodos sean sencillos y solo realicen una tarea para la simplicidad del mantenimiento.

6.2 ALGORITMOS

Entre los algoritmos más representativos nos encontramos con:

Algoritmo Zhang-Suen de adelgazamiento, es el encargado de generarnos el esqueleto de un pixel de ancho de la huella dactilar para ello itera repetidas veces sobre la imagen mientras produzca cambios, va comprobando si el pixel puede ser borrado y en caso de que sea posible lo elimina, es un algoritmo bastante simple de entender pero realmente complejo de implementar ya que depende del orden en el que compruebes o en el que borres, puedes llegar a situaciones finales diferentes.

Otro algoritmo utilizado, aunque no es realmente tal, es la utilización de los "Crossing Number" para la identificación de las minucias y evitar así los falsos positivos y los cortes debido al ruido, el uso de esta técnica vino forzada al problema de los falsos positivos que se generaban usando otra tras realizar el adelgazamiento con el algoritmo de Zhang-Suen.

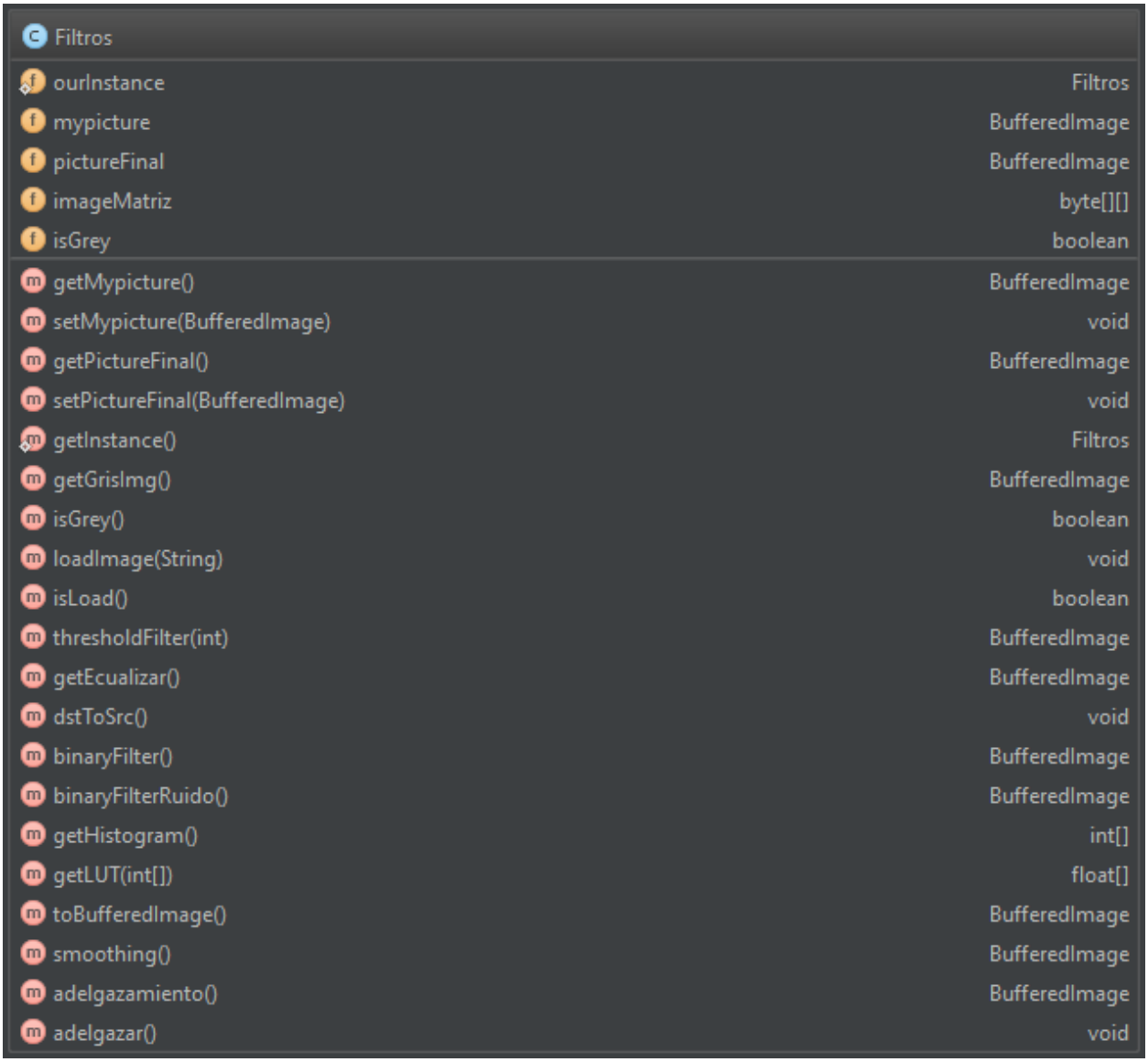
Otro algoritmo, pero tampoco como tal ha sido el que he implementado para el matching de las huellas, para ello obtenemos todos los posibles triángulos generados con las minucias de bifurcación existentes en la huella, siempre y cuando cumplan con unas condiciones de tamaño y posición, ya que un triángulo muy al borde de la huella puede ser obviado si la siguiente vez que se pruebe no se coloca la huella exactamente igual. Una vez tenemos esos triángulos obtenemos los ángulos de dicho triángulo. Dichos valores nos servirán para comprobar con la huella guardada o para guardarlos y verificar las siguientes. En el caso de desplazamientos cartesianos en el plano ni área ni ángulos varían, por lo que es muy eficaz, en el caso de las rotaciones (por ligeras que sean) o rotaciones junto con translaciones sí parecen afectar al método variando aproximadamente algo menos del 10% del área y cambiando los ángulos. No se ha podido indagar más en el motivo o probar diferentes posibilidades debido al reducido tiempo y a los escasos conocimientos matemáticos.

6.3 CLASES IMPLEMENTADAS

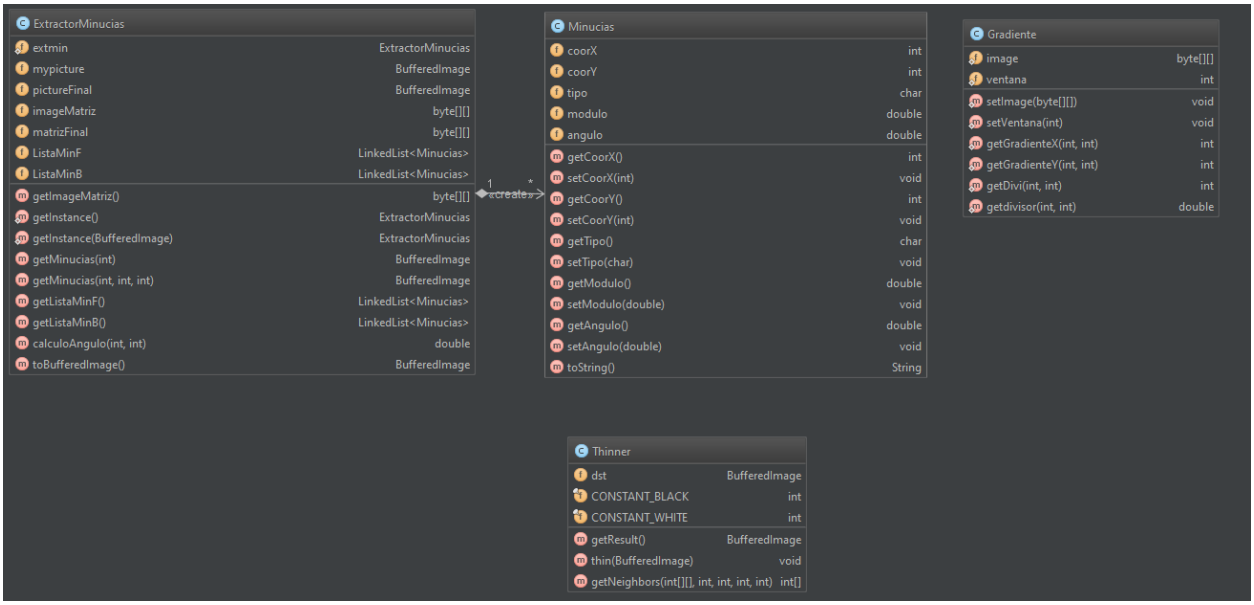
Para este apartado vamos a representarlo con un diagrama de clases y describirlo brevemente.

La aplicación está dividida en 3 paquetes diferenciados por su función.

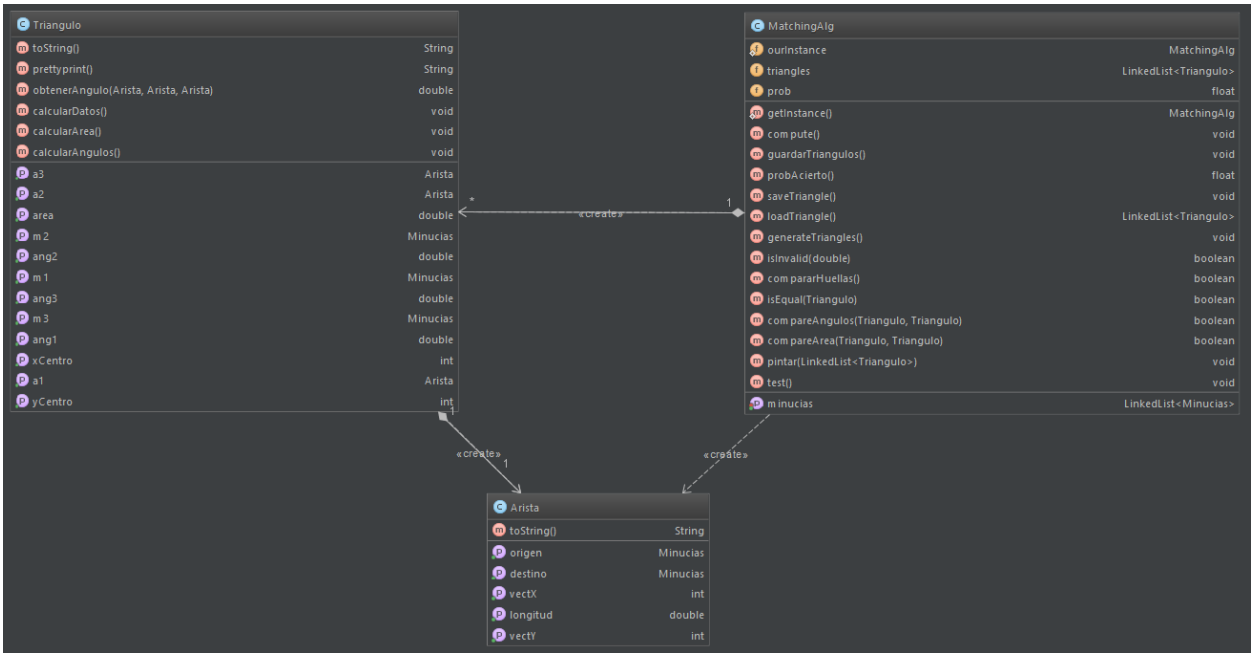
Primer paquete: Filtros. En este paquete se encuentra la clase que realiza todo el preprocesamiento de las imágenes. Se utiliza el patrón Singleton para mantener la coherencia durante todo el proceso y que solo exista una clase durante toda la ejecución.



Ahora veremos el paquete encargado de realizar la obtención de las minucias, en el tenemos la clase Minucias que contendrá la información, la clase Thinner que es la encargada de realizar el adelgazamiento, la clase Gradiente que se encarga de calcular los gradientes para obtener el ángulo de las minucias y la clase ExtractorMinucias que es la encarga de realizar todo el proceso de extracción, creación, almacenamiento y muestra de dichas minucias. Volvemos a valernos del patrón Singleton para mantener solo un extractor de minucias en el sistema.



Por último el paquete encargado de realizar el matching, consta de tres clases, dos de ellas contenedoras de información, la clase Arista, que guarda la información de las aristas del triángulo y la clase Triángulo, que guarda toda la información referente a los triángulos que generamos.



Luego podemos encontrar la clase Main que ejecuta una instancia de la clase Gui que es a través de la cual podemos interaccionar con el sistema.

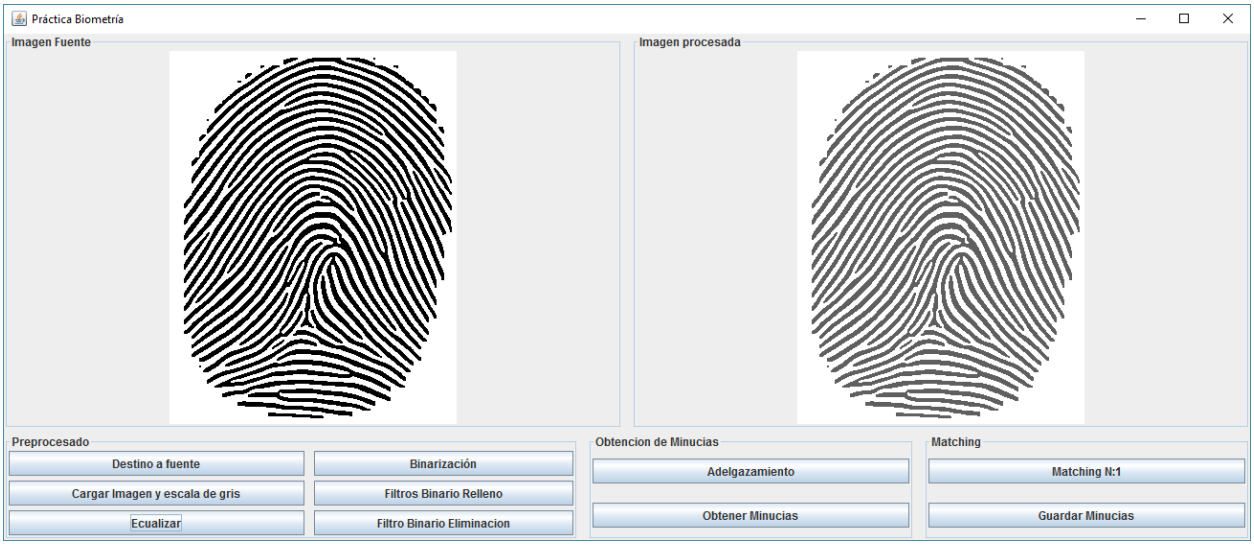
7 EJEMPLO DE EJECUCIÓN

Dividiremos esta sección en dos partes ya que se ha conseguido un matching funcional.

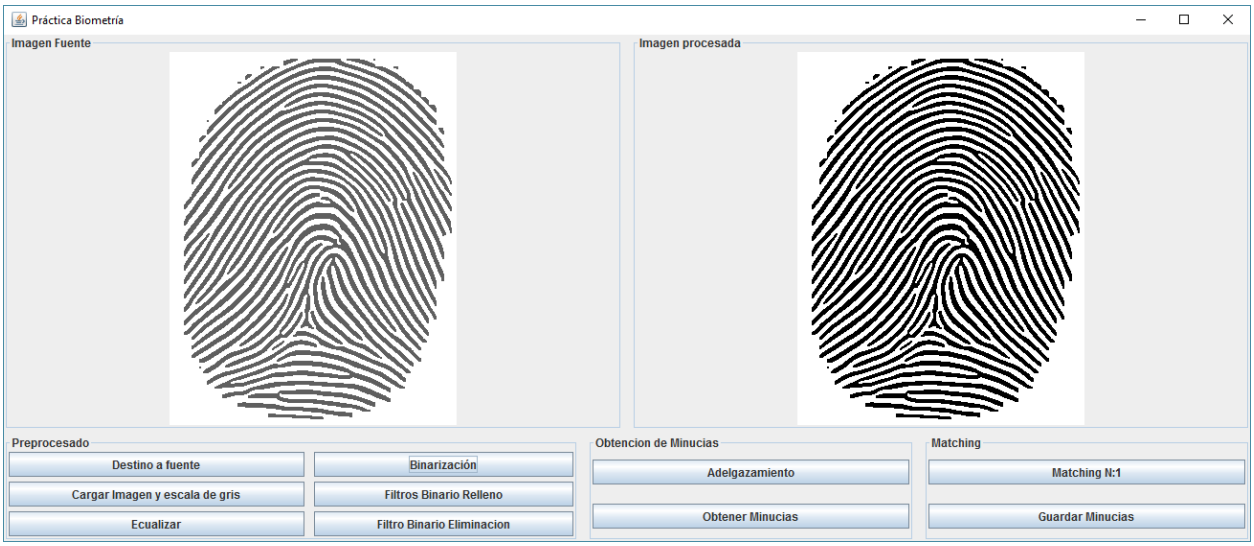
7.1 GUARDADO DE UNA HUELLA

En esta sección se explicará el proceso para guardar los datos de una huella en el sistema para poder comprobar más tarde si una huella es la misma.

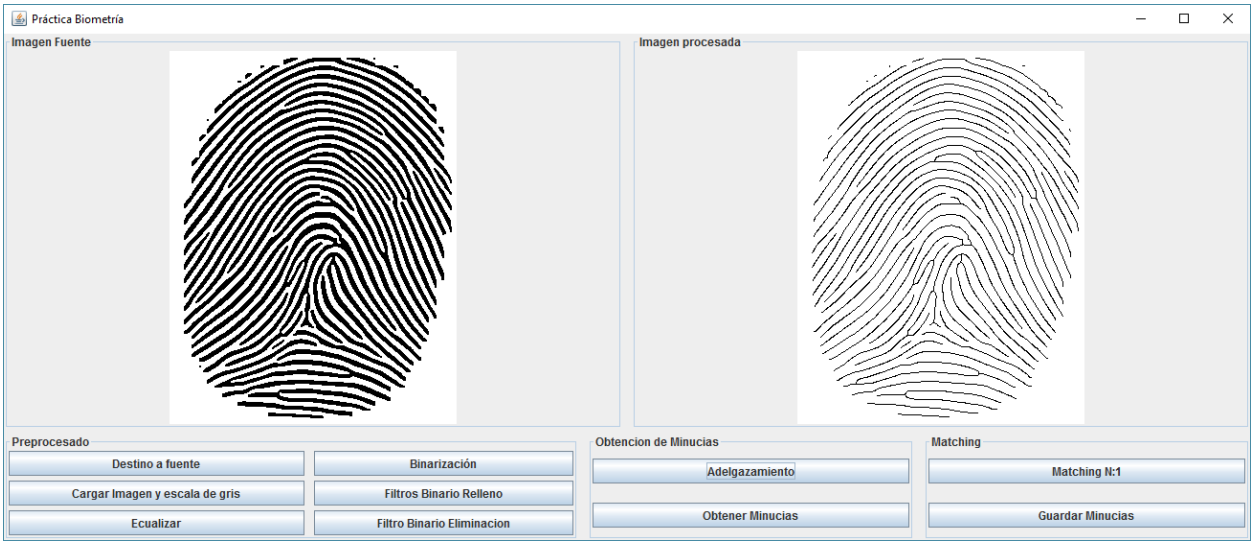
Inicialmente cargaremos la imagen (File_008.png) y le aplicaremos la ecualización.



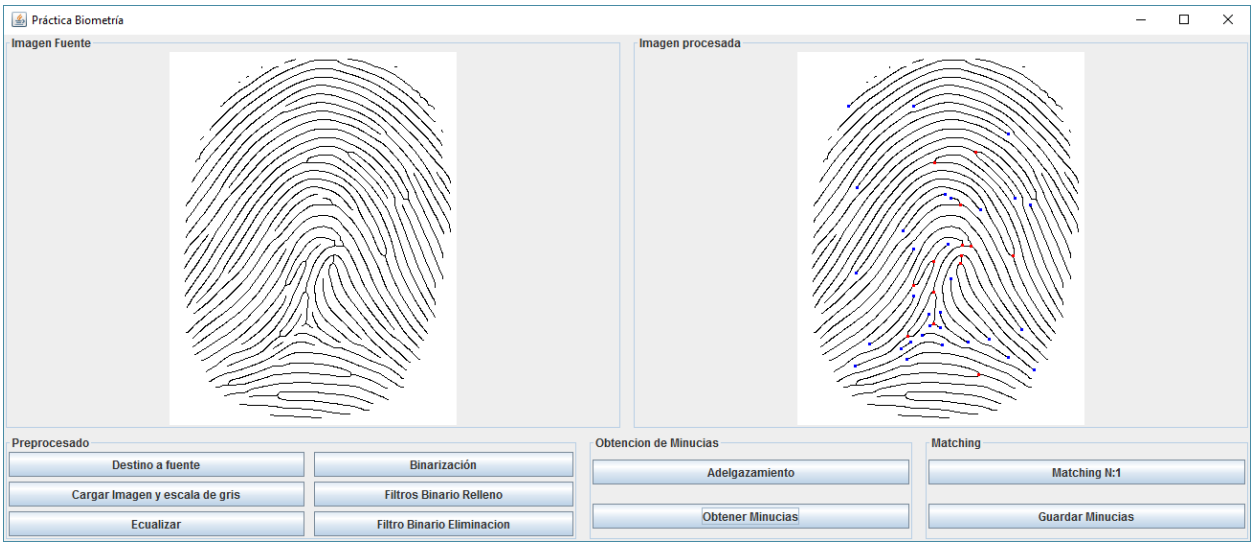
Una vez tenemos la huella en ese estado, seleccionaremos “Destino a fuente” para trabajar sobre la huella procesada, pulsaremos sobre binarización y seleccionaremos de “100” quedándonos lo siguiente.



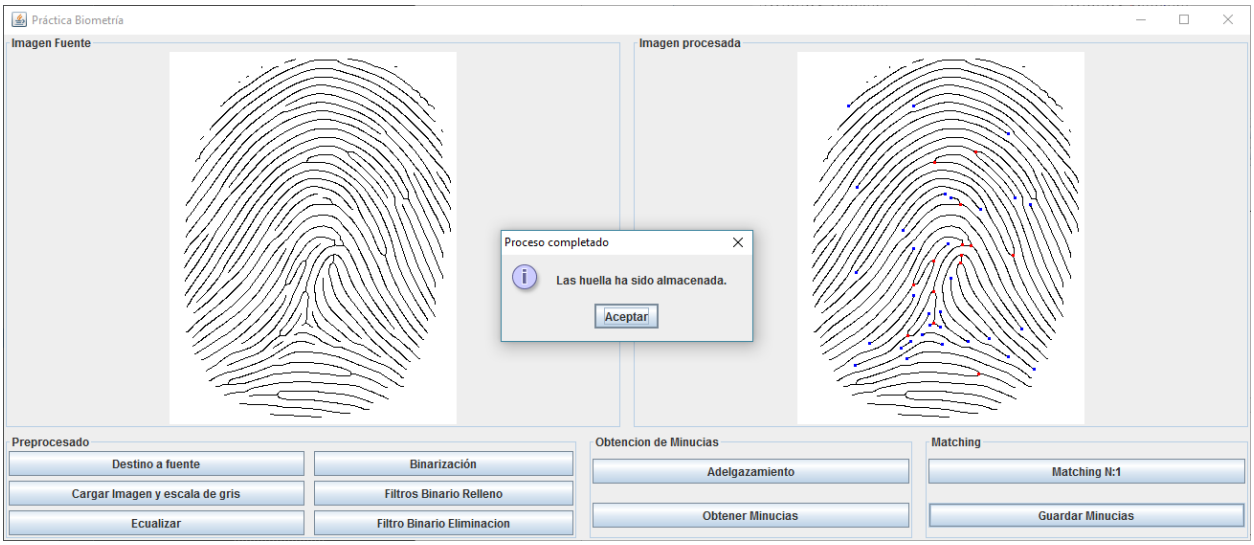
Nótese que es igual que la inicial, esto es debido a que esa imagen ya está preprocesada, una vez aquí volveremos a pulsar sobre “Destino a fuente” y aplicaremos directamente el “Adelgazamiento” porque al ser una imagen de muy buena calidad no es necesario utilizar los filtros.



Podemos ver el esqueleto de la huella perfectamente hecho, volveremos a pulsar en “Destino a fuente” y ahora pulsaremos sobre “Obtener Minucias”, para este paso introduciremos tres veces el valor de “50” para eliminar las minucias de los bordes.



Podemos ver cómo están marcadas las minucias en la imagen. Ya está todo listo para guardar la información de dicha huella en el sistema. Pulsaremos sobre “Guardar Minucias” y nos saldrá una ventana informándonos de que se han guardado con éxito.

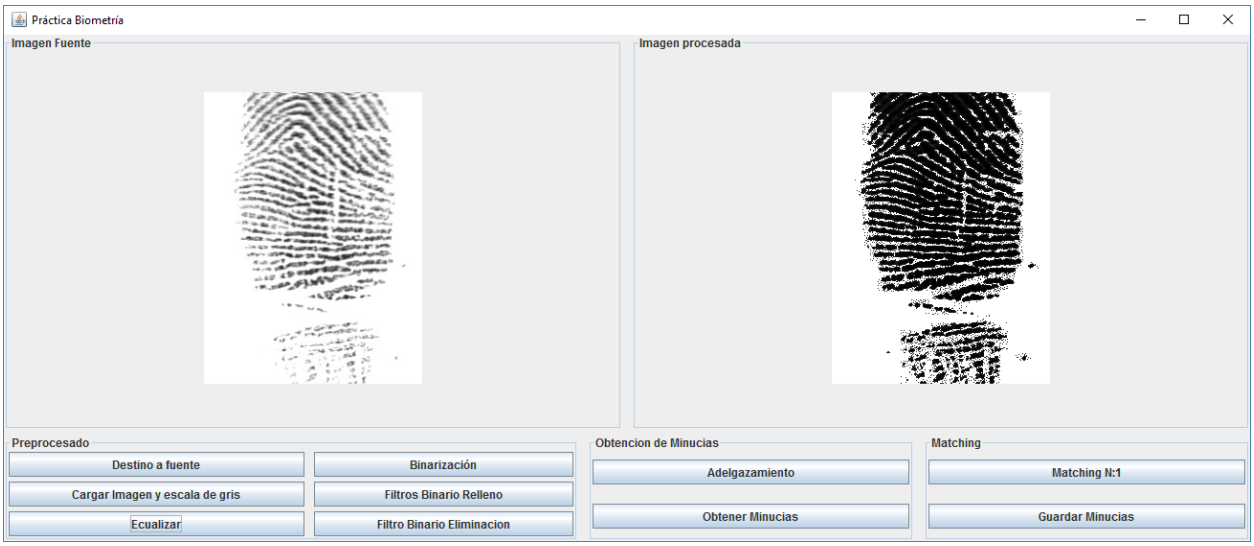


Una vez llegado a este punto ya tenemos la información de la huella almacenada y lista para comparar con otras.

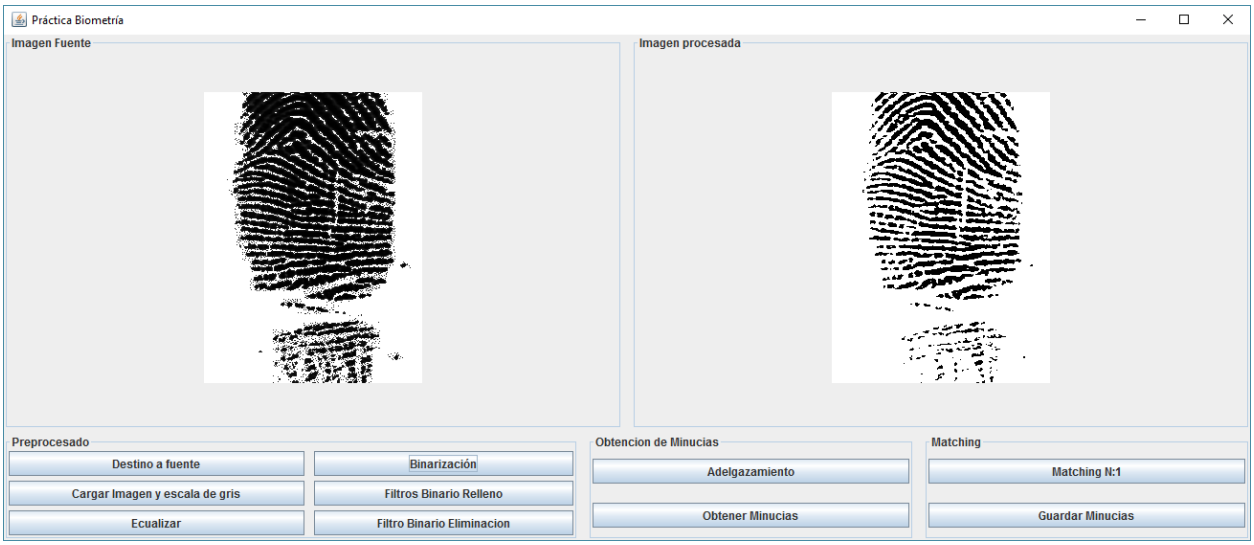
7.2 MATCHING DISTINTA HUELLAS

Como los pasos son los mismos que en el apartado anterior las explicaciones serán mucho menores, al menos hasta el punto donde empiezan a diferenciarse.

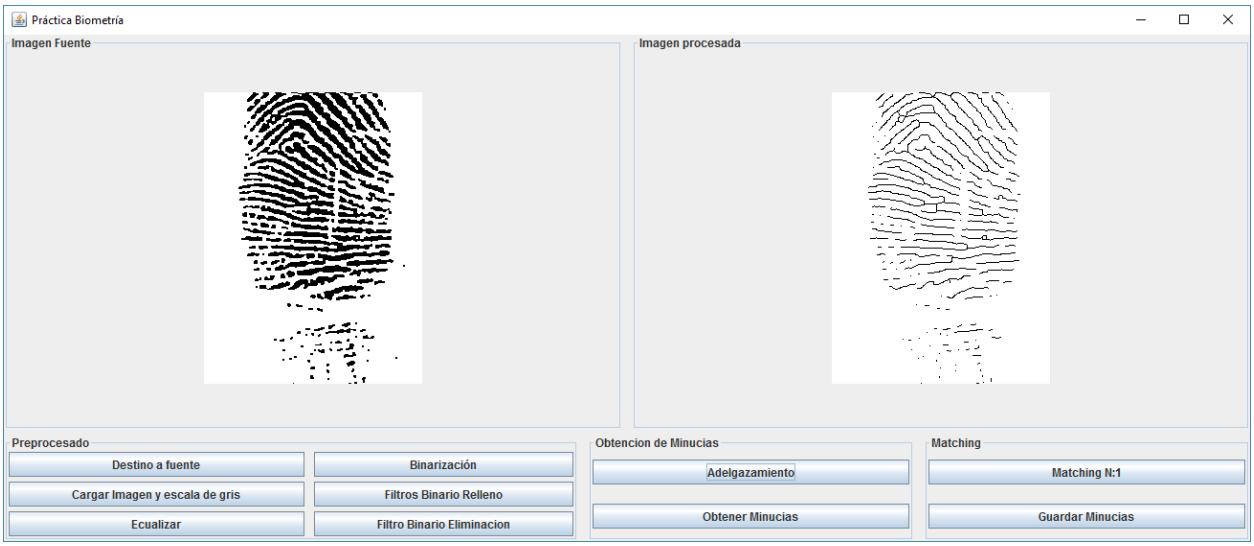
Cargaremos una huella en la aplicación, en este caso 101_1.png y la ecualizamos.



A continuación aplicaremos una binarización con el valor “60” obteniendo lo siguiente.



Una vez tengamos esto, pasaremos otra vez la imagen procesada a la fuente y usaremos los filtros binarios hasta que no produzcan cambios. Una vez hayamos realizado eso, pulsaremos sobre “Adelgazamiento”.



Nos saldrá algo parecido a esto, teniendo el esqueleto lo volveremos a pasar a la fuente y realizaremos la obtención de minucias, como podemos realizarla sucesivas veces, podremos probar distintas ventanas para eliminar minucias en los bordes, en este caso usaremos “50” para los laterales, “30” para la parte superior y “150” para la parte inferior de la imagen.

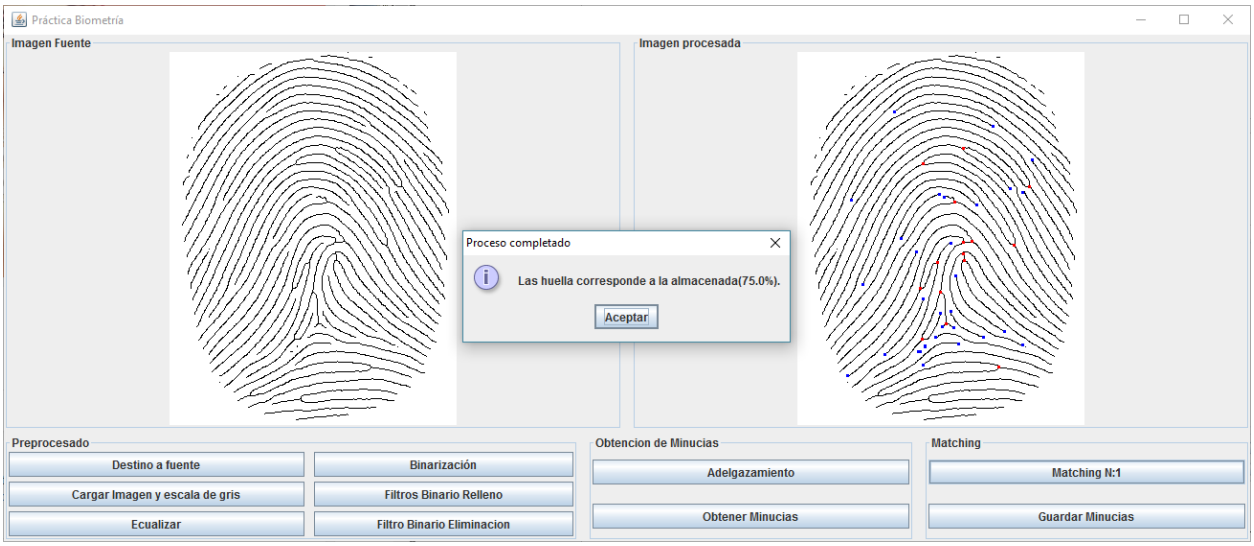
Una vez tengamos esto, que podremos realizar el matching entre esta huella y la almacena en el paso anterior, solo tendremos que pulsar el botón “Matching N:1”.

Obtenemos este resultado, debido a que el algoritmo de matching no es extremadamente bueno, detecta algunas similitudes, pero como se pueden ver son de un 23%, que puede ser que haya encontrado 1 triangulo parecido a alguno de los almacenados.

Ahora se muestra otra prueba en la que omitiremos el paso hasta obtener las minucias sobre la imagen File_008.png rotada 45º y otra con la imagen desplazada horizontalmente.

Con la imagen desplazada horizontalmente, como podemos ver en la siguiente imagen hay un 100% de coincidencia eso es debido a que los triángulos en los que nos fijamos no han sido deformados en ningún momento solo desplazados.

Ahora realizaremos lo mismo con la imagen rotada 45%, como podemos ver la rotación (con el justificado desplazamiento para que siga siendo cuadrada la imagen) si ha modificado algún triángulo, dándonos solo un porcentaje de similitud del 75% (3 de 4 triángulos almacenados).



Se aprecia que al rotarla se ha producido una pequeña deformación en la huella, que todavía ha aceptado la aplicación.

8 ASPECTOS MEJORABLES

Este apartado lo he añadido para remarcar ciertas partes las cuales no he podido llevar a cabo debido al tiempo o a las imposibilidades técnicas.

En primer lugar me hubiese gustado ampliar el matching para que no fuera 1:1 usando una base de datos con ID de usuarios, enlazados los respectivos triángulos generados de sus huellas, para poder así comparar con más de una huella y ver el rendimiento en una situación un poco más exigente. Este apartado no he podido llevarlo a cabo como ya dije anteriormente porque no dispongo de ningún sitio donde montar una base de datos a la que acceder.

El tiempo tampoco ha sido un factor favorable, ya que no he podido implementar una forma de retroceder entre operaciones realizadas para poder así evitar tener que realizar el proceso entero si te equivocas en algún punto.

Testeo y mejora en el algoritmo de matching, el algoritmo actualmente funciona pero solo fue testeado con las imágenes que se adjuntan al proyecto, con ninguna más y aun así no se probó con todas las opciones posibles. Me hubiese gustado perfeccionarlo un poco más antes de entregar el proyecto.

9 CONCLUSIONES

El desarrollo de la práctica ha sido muy interesante y muy instructivo desde el punto de vista de recordar muchos fundamentos matemáticos que parecía que nunca iba a volver a utilizar, pero que han sido muy útiles, además de ver la cantidad de aplicaciones que tienen las matemáticas en el reconocimiento de patrones en imágenes, algo que ya sabía pero no esperaba encontrarme tal nivel matemático para algo que parecía “simple”.

Me gustó mucho también el hecho de que se desarrolle algo que actualmente se usa comercialmente tanto en dispositivos móviles, como en dispositivos de seguridad (aunque a diferentes niveles).

En general ha sido una práctica muy entretenida, aunque con un poco más de tiempo se podría haber avanzado bastante más. Hubiese estado interesante también la de reconocimiento de iris. Es reseñable también que al haber trabajado con imágenes antes no supuso gran problema trabajar con las imágenes.