Sistemas Informáticos
Curso 2002-03

Interfaz de tratamiento de imágenes
SDTI v1.0

Carlos Murciano Díaz
Jose Manuel Alonso Torezano
Sonia Risueño Leonor

Dirigido por:
Prof. Gonzalo Pajares Martisanz
Dpto. de Arquitectura de Computadores y Automática

Facultad de Informática
Universidad Complutense de Madrid

ÍNDICE

.

1.- INTRODUCCIÓN

El proyecto consiste en el desarrollo de un sistema software de tratamiento de imágenes. Este proyecto fue iniciado el año pasado por otro grupo y se denominó ATIM v1.0. Nosotros, basándonos en este sistema hemos desarrollado SDTI v1.0 (Sistema de tratamiento de Imágenes) ampliando su funcionalidad.
Entre las nuevas funcionalidades se encuentran más filtros sobre imágenes, operaciones morfológicas (para el tratamiento de regiones), operaciones de histograma y funcionalidades específicas que podrían tener aplicación en el campo de la medicina, como las binarizaciones de imágenes con distintos parámetros, y cálculo de momentos invariantes para regiones.

Inicialmente el proyecto está orientado al tratamiento específico de imágenes médicas como indica el nombre del proyecto base (Aplicación de Tratamiento de Imágenes Médicas). Sin embargo, hasta el momento todas las operaciones que se han implementado pueden ser utilizadas sobre cualquier tipo de imagen, aunque al final del proyecto se han realizado operaciones que tienen aplicación en el campo de la medicina.

Para la implementación del proyecto se ha empleado el lenguaje de programación JAVA debido a que es un lenguaje multiplataforma que puede ser ejecutado sobre diferentes sistemas. Además, se ha tenido en cuenta que es uno de los lenguajes más utilizados en la actualidad, contando además con una amplia documentación.

El objetivo principal del proyecto ha sido completar una aplicación de tratamiento digital de imágenes de temática genérica. El sistema trata imágenes en color (RGB e HSI), de intensidad de grises e imágenes binarias y soporta distintos formatos de almacenamiento (.jpg, .png, .tif, .bmp, etc...).

Se considera prioritario que la aplicación sea modular y fácilmente ampliable para que el trabajo que ha sido realizado en este proyecto pueda ser continuado en años sucesivos con procesamiento de imágenes avanzado y centrado en temas más concretos.

La realización del proyecto se ha llevado a cabo en tres fases principales. La primera consistió en la comprensión de la arquitectura del proyecto ATIM, precedente del actual. Posteriormente, y respetando dicha arquitectura se fueron incluyendo clases y modificando otras existentes para dotar a la aplicación de las operaciones que serán comentadas en la especificación de requisitos más adelante. En la última fase se incluyeron las operaciones especiales del menú “Aplics.”, con posible aplicación en imágenes médicas.

Para la implementación de las operaciones sobre las imágenes a bajo nivel se han utilizado las librerías JAI, a través de las cuales se consigue un acceso de

lectura y escritura a las imágenes píxel por píxel. De esta manera se puede acceder al formato de color de una imagen y aplicando los diferentes algoritmos se puede implementar cualquier operación sobre dicha imagen.

Por último, se vuelve a insistir en que la vocación del proyecto es que pueda servir como base para proyectos futuros de tratamiento de imágenes y por lo tanto se incluye en esta memoria un apéndice en el cual se dan indicaciones de cómo ampliar la aplicación.

INTRODUCTION

The project consists of development of a software system for digital images treatment. This project was begun the last year for other team and they called it ATIM v1.0. Our team are based on ATIM v1.0 to develop SDTI v1.0, extending the functions of the system. Among the new functions, there are more filters for images, morfologic operations (for the treatment of regions), operations for histogram and specific operations that could be used in medicine like binaries images with several parameters and calculation of invariant moments for regions.

At the beginning, the project was orientated to specific medical images, like the title of the project base indicates. However, at now, all the operations done may be used to treatment of all kind of image, although the last operations that was included in project could be applied in medicine.

We have chosen the JAVA language programming because is a multiplatform language, is a very used language at now and besides there is a lot of documentation of it.

The main objective of the project has been to complete an application for digital images treatment of all kind of thematic. System treats colour images (in RGB and HSI formats), grey images, binaries images and supports several formats of storage (.jpg, .png, .tif, .bmp. etc...).

Another main objective is the modularity. So, the work of enlarging this project with advanced functions for treatment of images (with specific thematics) will be easier.

Therefore, this project is open to be continued the next years. There is a chapter of this report relating to this theme.

This project has been made in three phases. First phase was dedicated to the studing of architecture of ATIM (preceding project). In second place we have included and modified a lot of classes, respecting the original architecture to include the new functions that are described in the specification of requirements. In the last phase, we included the operations of the menu “Aplics.”.

In the implementation of the operations, we have used JAI libraries of JAVA, So, with de apropiated classes, we can read and write all pixels of a image and apply the algorithms.

At last, we insist again in that one of the main objectives of this project is that was the base to others and our work was be continued.

3.- AUTORIZACIÓN

Se autoriza el uso de este proyecto con fines exclusivamente académicos, en la medida que la Facultad de Informática estime oportuno.

Fdo: Carlos Murciano Díaz. DNI: 46888628-T

Fdo: José Manuel Alonso Torezano. DNI: 50319248-D

Fdo: Sonia Risueño Leonor. DNI: 2266605-R

4.- DIEZ PALABRAS CLAVE DEL PROYECTO

- **Binarización:** (con o sin parámetros). Procedimiento según el cuál se transforma una imagen a blanco y negro transformando cada color al extremo más próximo.

- **Filtro:** Transformación de imagen que elimina información.

- **JAI:** (Java Advanced imaging) Librería de componentes JAVA sobre tratamiento avanzado de imágenes.

- **JAVA:** Lenguaje de programación orientado a objetos empleado en el desarrollo de este proyecto.

- **Histograma:** Representación gráfica del análisis de frecuencia de los valores de muestreo estadístico donde los valores de muestreo son los valores de intensidad de color.

- **Operación morfológica:** Simplifican las imágenes y preservan las formas principales de los objetos. Se suelen utilizar para el tratamiento de regiones. Ese tratamiento consiste en determinar como se pueden cambiar, contar y evaluar.

- **Momentos invariantes:** Propiedades de las regiones de una imagen. Son especialmente importantes los momentos invariantes de Hu, que son constantes para las regiones de una imagen en las rotaciones, traslaciones y cambios de escala.

- **Píxel:** Contracción de picture element. Unidad básica de color programable de una imagen digitalizada.

- **RGB:** (Red Green Blue). Formato de representación de color mediante la composición de luminosidades del rojo, verde y azul.

- **Modularidad, reusabilidad y fácil ampliación:** Principios básicos en el desarrollo del proyecto. Se pretende facilitar el trabajo futuro sobre este proyecto.

5.- ESPECIFICACIÓN DE REQUISITOS

La orientación de la funcionalidad del sistema corrió a cargo del tutor del proyecto. Los requisitos funcionales se listan a continuación:

- Lectura de imágenes desde archivo en diferentes formatos: GIF, JPEG, TIFF, PNG, PNM y BMP.

- Escritura de imágenes a archivo en diferentes formatos: JPEG, TIFF, PNG, PNM y BMP. Se ha excluido el formato GIF por estar limitado a 256 colores.

- Control de cambios, esto es, debe permitir deshacer y rehacer.

- Uso de portapapeles local: Cortar, copiar y pegar.

- Transformaciones entre formatos gráficos: RGB, HSI y escala de grises.

- Mecanismos de manipulación de imágenes tales como:
- Transformaciones afines: rotación, inclinado, traslación y escalado;
    - Inversión cromática;
    - Reflexión horizontal y vertical;
    - Binarización y función umbral;
    - Ajuste de brillo, contraste y corrección gamma;
    - Aplicación de filtros: pasa baja, pasa alta y personalizado;
    - Extracción de bordes: Sobel, Prewitt y Laplace;
    - Inserción de texto;
    - Inserción de figuras o formas básicas: líneas, rectángulos y elipses.

- Herramientas de análisis de la imagen como:
    - Obtención de los canales de la imagen;
    - Cálculo y visualización del histograma.

Se desea además que las diferentes adiciones a la imagen (texto, formas, o fragmentos de imagen pegados) puedan ser trasladas y eliminadas individualmente. Esto ha terminado transformándose en que la aplicación tenga gestión de capas, entendiendo por capa una sección flotante de la imagen y siendo la imagen la superposición de todas sus capas.

Requisitos no funcionales.

Aunque los alcances de estos requisitos no queden delimitados completamente, se pedía además que la aplicación fuese:

Multiplataforma y eficiencia razonable en la ejecución de las operaciones.
De fácil uso.

Estos requisitos se han satisfecho de la siguiente forma:

→ El hecho de haber sido programada en Java garantiza su funcionamiento en la totalidad de plataformas de funcionamiento actuales.

→ Todas las manipulaciones de imágenes de tamaño medio consumen menos de 5 segundos en un PC con velocidad aproximada de 500 MHz, con las salvedades:

- Operaciones que involucren el formato HSI, ya que este formato representa los colores con números en coma flotante, y por tanto todas sus operaciones conllevan mayor tiempo de proceso en la unidad aritmético lógica.

- Imágenes que combinen muchas capas, por la complejidad añadida que conlleva la superposición de las mismas.

- Operaciones de binarización con parámetros. En el caso del recorrido en zonas concéntricas a partir de un píxel seleccionado por el usuario con el ratón (binarización "especial" 2), dependiendo del tamaño de la imagen el tiempo de realización de la operación puede llegar a ser de un minuto y medio, debido a la complejidad exponencial que requiere dicha operación.

- Cálculo de los momentos invariantes de regiones seleccionadas por el usuario en imágenes binarias. Si los valores de los parámetros p y q son muy altos, el tiempo de realización se ve incrementado debido a la complejidad del cálculo de exponentes de gran valor. No obstante el tiempo de realización también dependerá del tamaño de la región seleccionada, aunque los valores p y q son más determinantes.

- La ecualización de histograma sobre una imagen es una operación de complejidad elevada por lo que puede llegar a tardar en realizarse sobre 30 segundos dependiendo del tamaño de la imagen

→ La facilidad de uso se consigue a través de un interfaz gráfico similar a los programas de dibujo y tratamiento de gráficos disponibles en el mercado (Paint, Corel Draw, Adobe Photoshop), de modo que cualquier usuario medianamente familiarizado con estas puede empezar a manejar la aplicación sin necesidad de leer ningún manual.

La aplicación es sensible al contexto, de forma que se habilitan las opciones del menú que es posible realizar en cada momento, evitándose así posibles fallos provocados por el usuario. Por ejemplo, si no se ha abierto ninguna imagen en un marco del programa, no estarán habilitadas las opciones relativas a la realización de operaciones morfológicas, de histograma, filtros y demás operaciones sobre imágenes.

En otros casos, las opciones del menú sí están habilitadas, pero en función del tipo de imagen que haya en el frame activo se indicará si es posible o no llevar a cabo la operación. Por ejemplo, si se trata de pulsar la opción "Momentos invariantes" del menú "Aplics." con una imagen en color RGB en el marco activo, el sistema informará de que se requiere una imagen binaria y la operación no se realizará.

También puede ocurrir que en alguna opción se haga un tratamiento por defecto. Si se trata de hacer operaciones morfológicas sobre una imagen en color RGB, el sistema la convierte automáticamente en una imagen de grises y aplica la transformación sobre esa imagen. (Las operaciones morfológicas solo están definidas para imágenes binarias y de intensidad de grises.

Funcionalidades añadidas:

Se han añadido otras funcionalidades que se han considerado de utilidad como transformación por extracción de color, duplicación de una imagen, consulta de las propiedades de la imagen, filtro de la mediana, operaciones aritmético-lógicas entre imágenes (AND, OR, XOR, suma, resta, producto y división), etc

Nuevas funcionalidades:

- Operaciones morfológicas:

  Estas operaciones simplifican las imágenes preservando las formas principales. Se utilizan en el tratamiento de regiones:

  - Dilatación de imágenes binarias y de grises.
  - Erosión de imágenes binarias y de grises.
  - Apertura de imágenes binarias y de grises.
  - Cierre de imágenes binarias y de grises.

- Operaciones de transformación de histograma:

  Consisten en modificaciones del histograma con efectos de aumento de contraste, de brillo, etc sobre la imagen.

  - Expansión de histograma para imágenes de color (RGB) y de grises.
  - Contracción de histograma para imágenes de color (RGB) y de grises.
  - Desplazamientos de histograma para imágenes de color (RGB) y de grises.
  - Ecualización de histograma para imágenes de color (RGB) y de grises.

- Filtros:

Transformaciones de las imágenes con distintas máscaras.

  - Filtro del máximo.
  - Filtro del mínimo.
  - Filtro de la gaussiana.
  - Filtro de Frei-Chen.

- Operaciones especiales orientadas a la medicina:

  - Binarización “especial” 1.

    Seleccionado un píxel de la imagen con el ratón, se binariza la imagen en función de la distancia de los demás píxeles al seleccionado en relación con un umbral seleccionado por el usuario.

  - Binarización “especial” 2.

    Análoga a la anterior pero para el cálculo de las distancias de los demás píxeles en relación al seleccionado, se recorre la imagen en zonas concéntricas. El umbral puede decrementarse o incrementarse en “x” unidades en cada zona, parámetro seleccionado por el usuario. De la misma manera también se requiere un umbral.

    Estas binarizaciones pueden servir para detectar ciertas regiones en imágenes médicas (anteriormente se comentó el ejemplo del colesterol).

  - Cálculo de momentos invariantes.

    Cálculo de los momentos para las regiones seleccionadas con el ratón y con los valores de p y q elegidos por el usuario. Se trata de propiedades importantes puesto que son valores constantes invariantes a rotaciones, traslaciones y cambios de escala.

Todas las operaciones nuevas son explicadas con cierta profundidad y con ejemplos en el apartado de la memoria “Nuevas operaciones sobre imágenes”.

6.- ANÁLISIS DEL SISTEMA

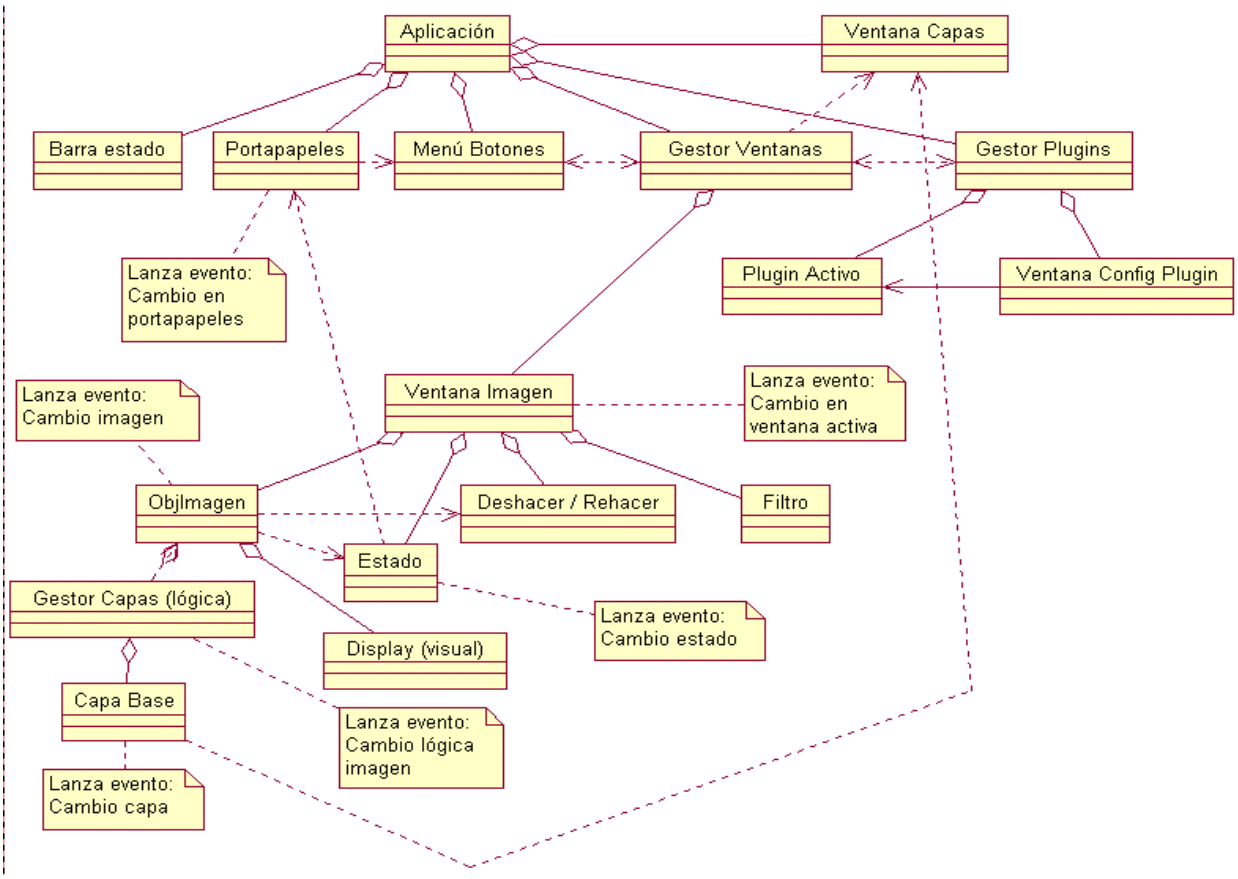
El modelo del sistema ha sido realizado en UML debido a que JAVA es un lenguaje orientado a objetos.

Los diagramas que se adjuntan a continuación muestran la organización lógica del sistema sin entrar en detalles de cómo se ha implementado dicha arquitectura.

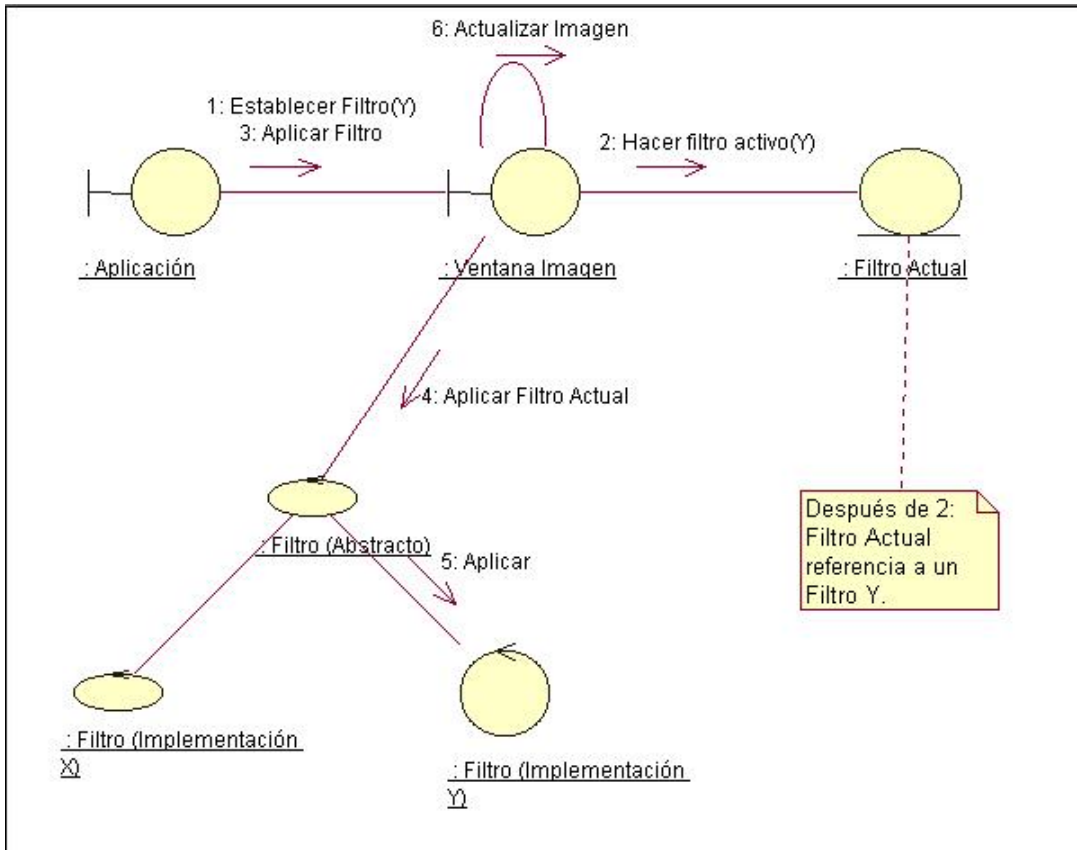
Como se ha dicho anteriormente SDTI v1.0 es un sistema que dota a ATIM v1.0 de una mayor funcionalidad basándose en su estructura por lo cuál el análisis del sistema es el mismo y de la misma manera, el tratamiento de eventos y la forma de aplicar los filtros también se conserva.

Ambos diagramas se muestran a continuación:

→ Aplicación de un filtro genérico sobre una imagen:



→ Aplicación de un filtro cualquiera



→ Uso de Custom Events (eventos personalizados):

7.- DISEÑO DE LA ARQUITECTURA

La arquitectura del sistema ha sido diseñada pensando en la reusabilidad y modularidad. Un principio básico del diseño es que solo se permiten las relaciones hacia abajo y excepcionalmente las laterales, mientras que las relaciones hacia arriba se reemplazan con eventos de usuario (custom events) cuyo funcionamiento se describió anteriormente. La gran ventaja de los custom events es que hacen a los componentes plenamente autónomos, pudiéndose cambiar de lugar (incluso localizaciones remotas con CORBA) manteniendo su plena funcionalidad. También es importante mencionar el diseño modular de los menús que permite añadir nuevas opciones y operatividad fácilmente.
Esto permite una gran modularidad, que como se ha dicho anteriormente es un objetivo principal en el diseño.

La ampliación del sistema se ha realizado siguiendo estos principios de diseño y se invita a las personas que puedan seguir ampliando el sistema y dotándolo de nuevas funcionalidades a seguir esta línea.

- Además, este tipo de desarrollo permite:
  - Vender los componentes desarrollados por separado (JAVA BEANS).
  - Integrar los objetos en un servidor de aplicaciones.
  - Enfocar la aplicación como un sistema distribuido (por ejemplo utilizando CORBA).
  - Solventar en parte uno de los grandes problemas de Java: la eficiencia. Al poder lanzar los eventos desde en un proceso distinto al del objeto que los genera, y a cada uno de los que los “escucha”, podría existir un “demonio” (daemon) dedicado en exclusiva a recibir y procesar estos eventos.

La reusabilidad es otro de los objetivos principales del diseño y facilita notablemente la ampliación de la aplicación.
Las clases abstractas referentes a un formulario estándar y a un filtro estándar son un ejemplo muy claro. En toda la aplicación hay multitud de formularios para la entrada de parámetros por parte del usuario. Al tener una clase abstracta que implementa un formulario genérico, posteriormente para cada formulario concreto se crea una clase que implementa la clase abstracta citada anteriormente y que añade las características particulares a cada formulario. Con los filtros ocurre lo mismo. Hay muchos filtros diferentes realizados con clases diferentes que implementan una clase común y abstracta que define el filtro base.

8.- DISEÑO DETALLADO, CONSTRUCCIÓN, INTEGRACIÓN Y PRUEBAS

Una vez obtenido el diseño del sistema, es decir, los componentes del mismo y sus relaciones, se procede al diseño de cada componente por separado, a su prueba para determinar su corrección y por último a la integración en el sistema.

Toda la funcionalidad que requería el sistema se repartió en distintos componentes, cuyo diseño e implementación fueron asignados a los miembros del grupo.

Cuando un componente está ya implementado se procede a su integración y prueba en el sistema por separado para determinar la corrección. Para integrar y hacer las pruebas se realizan las modificaciones oportunas en el sistema.

Cuando ya están todos los componentes probados, se procede a integrar todos ellos en el sistema.

A continuación se describe a modo de ejemplo el desarrollo del componente TransformacionMorfologica (clase pública incluida en el paquete de la aplicación ATIM.Transformaciones).

Esta clase contiene una serie de métodos especializados en realizar operaciones morfológicas sobre imágenes. Todos ellos reciben como parámetro una imagen de tipo PlanarImage con la imagen original y devuelven de nuevo una PlanarImage modificada en función de la operación que se haya realizado.

Para la implementación de cada una de las operaciones se ha utilizado la JAI (Java advanced Imaging) de JAVA, librerías de tratamiento de imágenes que ofrecen una funcionalidad superior a las que ofrecen Java 2D y JFC (Java Foundation Classes). Sin embargo las operaciones que se han implementado para ampliar ATIM no tienen soporte directo en la JAI, es decir que no existen métodos ya implementados que directamente realicen las operaciones necesarias (por ejemplo, para los filtros ya desarrollados en ATIM, la JAI si daba muchas facilidades).

Por esta razón, todos los métodos han sido implementados directamente y se podría decir que el código desarrollado por el equipo ha sido de casi el 100% en relación al reutilizado de las librerías. La filosofía ha sido implementar todas las operaciones recorriendo las imágenes originales píxel por píxel y aplicando los algoritmos oportunos.

La capacidad de acceder a cualquier píxel de una imagen, obtener sus valores (RGB si es de color en este formato o un valor entre 0 y 255 si se trata de una imagen de grises) y del mismo modo ser capaces de escribir cualquier píxel en una imagen, proviene de la utilización de algunas clases de la JAI.

Una vez que se sabe como acceder y escribir los píxeles de una imagen, se realiza la implementación de las operaciones concretas en base a los algoritmos descritos en uno de los libros que se indicará en la bibliografía (“Visión por Computador”), así como las indicaciones del tutor.

Este componente podría ser integrado y utilizado en cualquier aplicación de tratamiento de imágenes realizado en JAVA que necesitaran tener disponibles operaciones morfológicas sobre imágenes, capacidad que como se ha dicho no proporciona directamente la JAI.

La integración del componente en el sistema requiere varias modificaciones y ampliaciones en el mismo. En primer lugar, hay que añadir una clase que implemente el menú para las operaciones morfológicas. En función de la operación que se quiera realizar se llamará a un método distinto en la clase cl_ventana_imagen que tendrá que tomar la imagen del frame activo, llamar al método adecuado del componente TransformaciónMorfológica, en el cuál se implementa como se ha explicado, la operación propiamente dicha, y que recibida la imagen modificada como resultado, la colocará en el frame activo de nuevo. Todos estos métodos tienen que ser añadidos. Además se requiere también la inclusión de nuevas clases que implementarán distintos formularios para la toma de datos o parámetros por parte del usuario para las distintas operaciones.

En el caso de las operaciones orientadas a la medicina, que como se verá requieren el tratamiento del ratón y los valores del píxel seleccionado, hay que añadir más cosas relacionadas con la gestión de eventos. Es decir que se requiere la modificación y ampliación de más clases.

Cuando el componente ha sido integrado individualmente en el sistema pasa a ser probado. Se comprueba con distintas imágenes que los efectos en las imágenes resultado son los esperados. En los casos en los que se detectan anomalías, estas se van resolviendo conforme avanza el periodo de pruebas.
En el momento en que el componente ha quedado totalmente probado, queda asegurada la funcionalidad del sistema, cubriéndose así una serie de requisitos funcionales (en este caso que estamos comentando, todas las operaciones morfológicas).

Por último, cuando se han desarrollado ya todos los componentes por separado y han sido probados individualmente se procede a la integración de todos ellos en el sistema. Así quedan cubiertos todos los requisitos funcionales especificados.

9.- TRABAJO FUTURO

Como ya se ha dicho, este proyecto ha sido construido sobre ATIM v1.0, realizado el año pasado.

Por lo tanto en el punto de partida ya se contaba con el interfaz gráfico implementado (con JAVA SWING) y algunas operaciones sobre imágenes, que se indican en la especificación de requisitos.

Nuestra labor ha consistido en ampliar la funcionalidad operativa de ATIM con las operaciones indicadas por el tutor. Han sido integradas en el sistema respetando la arquitectura diseñada y siguiendo las indicaciones de ampliación del

año pasado. Para ello se ha ampliado el interfaz y se han modificado y añadido las clases oportunas.

En este momento, SDTI v1.0 es un interfaz que puede realizar la gran mayoría de operaciones básicas de procesamiento de imágenes.
Sin embargo el proyecto no está cerrado puesto que su vocación inicial es que pudiera ser ampliado con nuevas funcionalidades. El desarrollo futuro puede ir encaminado a la especialización del sistema en procesamiento avanzado de imágenes en campos concretos. A continuación se citan algunos ejemplos:

Podría desarrollarse un **sistema de visión estéreo** con el que se pudiera calcular la distancia a objetos por ejemplo. Se tendría que hacer un tratamiento de imágenes 3D.

Otro ejemplo de procesamiento avanzado y muy específico sería la **fusión de imágenes**. Por ejemplo, hay un tipo de imágenes denominadas PET que se obtienen de pacientes con cáncer. Para tomar este tipo de imágenes, los pacientes toman glucosa, de forma que al ser el tumor una zona muy activa, consume una parte muy importante de glucosa. Al tomar la imagen del tumor con las técnicas apropiadas, la zona tumoral aparece coloreada. Sin embargo, para situar la zona coloreada sobre la anatomía del paciente es necesario fusionar la imagen PET con otra imagen de la anatomía de misma zona para que la zona coloreada se superponga en la imagen original.

Las **imágenes obtenidas por satélite** también podrían recibir un tratamiento específico. Por ejemplo es posible que interesara localizar en este tipo de imágenes aquellas regiones que tuvieran una mayor temperatura, para localizar los objetos o personas que se estimasen oportunos.

Éstos, solo son algunos ejemplos de las múltiples posibilidades que se abren a partir de este momento, para continuar con el desarrollo del proyecto.

Para realizar la ampliación del sistema se aconseja leer atentamente las indicaciones que se adjuntan a continuación.

Para la implementación de operaciones específicas sobre las imágenes con las JAI aconsejamos el estudio de las operaciones ya implementadas. Algunas de ellas utilizan facilidades de la JAI, y otras tratan las imágenes al más bajo nivel, accediéndolas píxel a píxel. Con este último método y el algoritmo adecuado se puede implementar cualquier operación sobre las imágenes.

10.- NUEVAS OPERACIONES SOBRE IMÁGENES

10.1.- OPERACIONES MORFOLÓGICAS.

Las operaciones morfológicas simplifican las imágenes y preservan las formas principales de los objetos. Se suelen utilizar para el tratamiento de regiones. Ese tratamiento consiste en determinar como se pueden cambiar, contar y evaluar.

Podemos utilizar la morfología para:

- suavizar los bordes de una región (mejorar un borde ruidoso que se ha obtenido por técnicas de segmentación estándar),
- para separar determinadas regiones unidas tras un proceso de segmentación,
- Y para unir regiones que han sido separadas durante la segmentación.

En definitiva, las operaciones morfológicas facilitan el cómputo de regiones en una imagen. La principal aplicación de estas operaciones matemáticas es el tratamiento de imágenes binarias puesto q los fundamentos matemáticos fueron concebidos desde el punto de vista de la posición y no de la intensidad. Sin embargo las definiciones también se extienden para imágenes de grises.

La morfología matemática se basa en la modelización de las imágenes utilizando conjuntos de puntos de cualquier dimensión. Desde esta perspectiva se consideran operaciones habituales como son la inclusión, unión, intersección, complementario y conjunto vacío. De este modo, una imagen binaria será un conjunto de puntos (pares de coordenadas) de aquellos píxeles de la imagen de valor"1" (255) respecto de un sistema de coordenadas de referencia. Normalmente el píxel de la esquina superior izquierda corresponde a las coordenadas (0,0).

Una transformación morfológica consistirá en la relación de una imagen X caracterizada como un conjunto de puntos con un elemento estructural B, también constituido por un conjunto de pares. La aplicación del elemento estructural sobre los píxeles de la imagen X dará como resultado la imagen modificada X + B.

Se han implementado las operaciones de dilatación, erosión, apertura, cierre, contorno y esqueleto sobre las imágenes binarias y de grises.

La JAI (Java Advanced Imaging) de JAVA, en este caso no ofrece clases ni métodos que faciliten la implementación de los algoritmos correspondientes a las operaciones morfológicas, por lo que se ha optado por tratar las imágenes accediéndolas píxel a píxel. Evidentemente, para poder acceder a una imagen píxel por píxel, se han utilizado clases de la JAI, como Raster, TiledImage, WritableRaster o RenderdImage.

La aplicación de las distintas operaciones morfológicas sobre las imágenes es contextual al tipo de imagen que esté presente en el frame activo en un determinado momento en la aplicación.

Es decir, si la imagen es binaria, el pulsar sobre cualquiera de las operaciones mencionadas anteriormente en el menú provoca la aparición de un formulario en el que se da a elegir el elemento estructural a utilizar en la operación. Los dos elementos estructurales posibles son:

a) b)

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$ Operador isótropo y $$\begin{bmatrix} *1 & 1 \end{bmatrix}$$

Si la imagen contenida en el frame activo es una imagen de grises, entonces al pulsar cualquiera de las opciones del menú Morfología aparecerá un formulario distinto indicando un único elemento estructural posible para llevar a cabo la operación. Ese operador estructural es:

$$\begin{bmatrix} *1 & 1 \\ 1 & 1 \end{bmatrix}$$

Si la imagen es de color (RGB), la aplicación realizará previamente una conversión a grises puesto que las operaciones morfológicas no están definidas para imágenes de color.

10.1.1.- Dilatación sobre imágenes binarias.

La operación morfológica de la dilatación combina dos conjuntos utilizando la adición de vectores. La dilatación X + B es el conjunto de puntos de todas las posibles adiciones vectoriales de pares de elementos, uno de cada conjunto X (imagen) y B (elemento estructural)

$$X + B = \{d \text{ de } E^2 : d = x+b \text{ para cada } x \text{ de } X \text{ y } b \text{ de } B\}$$

Se va a utilizar la siguiente imagen como ejemplo (original y binarizada).

Fig OM1
Imagen original

Fig OM2
Imagen binarizada

Fig OM 2
Imagen binaria.

Fig OM 3
Dilatación de OM 2.
Con elemento est b)

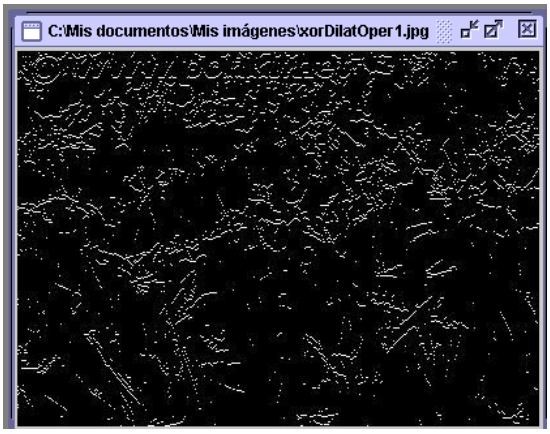


Fig OM 4
XOR entre fig OM 2 y
Fig OM 3

Fig OM 2
Imagen binarizada.

Fig OM 5
Dilatación de OM2
Con elm.Isotropo a)

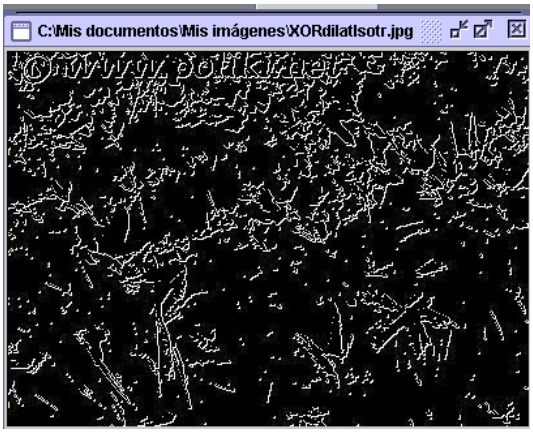


Fig OM 6
Imagen binaria.
XOR entre fig OM2 y
Fig OM 5

Las imágenes de la página() corresponden a operación morfológica de dilatación empleando el elemento estructural:

[*1 1]

Si se observa la imagen binarizada (OM 2) y se compara con la dilatada con este operador (OM 3), se puede apreciar que las regiones blancas (hojas caídas en el suelo en la imagen original) “han crecido” pero no de forma igual en todas las direcciones.

La imagen OM 4 es una operación XOR entre las dos imágenes anteriores y muestra por tanto las diferencias entre ambas. Los píxeles blancos en esta nueva imagen simbolizan precisamente la zona de crecimiento de las regiones blancas de la imagen dilatada respecto de la binarizada. (Los píxeles distintos serán “1”-blancos- en la XOR por que “1” XOR “0” = “1” y “0” XOR “1” = “1”).

Podemos ver que debido a las características del elemento estructural, el crecimiento de las regiones blancas ha sido pequeño y al mismo tiempo algo desigual.

A continuación se verá el efecto del operador isótropo y se comparará con el anterior.
El efecto del operador isótropo se puede apreciar en la página anterior. La figura OM 5 muestra como las regiones blancas son ahora bastante más grandes. Se puede comparar directamente figura OM5 con la OM3 y determinar que el crecimiento de las regiones ha sido mayor.
Además, ahora el crecimiento de las regiones blancas ha sido el mismo en todas las direcciones (dilatación isotrópica). Esto se puede observar perfectamente viendo las diferencias entre la imagen binaria original y la dilatada con el elemento isótropo. Estas diferencias se plasman en la imagen resultante de hacer la XOR de ambas, en la figura OM6. Se observa que en este caso las regiones blancas han crecido un píxel en todas las direcciones en contraste con lo que ocurría en la figura OM4.

Debido a que los dos tipos de operadores estructurales también pueden ser elegidos en las operaciones de erosión, apertura y cierre, no se volverá a hacer hincapié en las diferencias entre ambos y para la explicación de las siguientes operaciones morfológicas solo se pondrán ejemplos del operador isótropo.

10.1.2.- Erosión sobre imágenes binarias.

La operación morfológica de la erosión combina dos conjuntos utilizando la substracción de vectores.

$$X \times B = \{ d \text{ de } E^2 : d + b \text{ de } X \text{ para cada } b \text{ de } B \}$$

Fig OM 2
Imagen binarizada original.

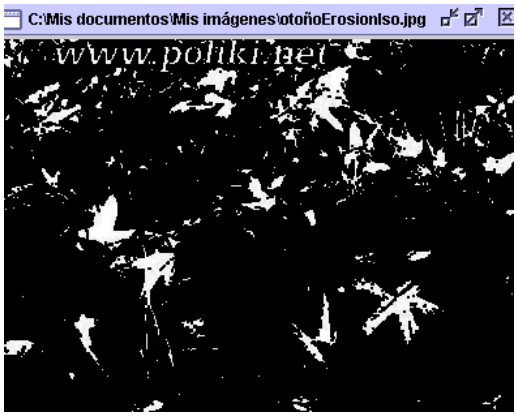


Fig OM 7
Imagen erosionada con el operador

Podemos observar el efecto de la erosión en las dos figuras anteriores. Las regiones blancas han reducido su tamaño en todas las direcciones, puesto que el operador aplicado es isótropo.

Tanto la dilatación como la erosión son operaciones no invertibles.

10.1.3.- Apertura sobre imágenes binarias.

La apertura es una operación morfológica que consiste en una erosión seguida de una dilatación. La imagen original no se recupera y el resultado es una imagen más simplificada y menos detallada que la original.

X ap B=(X x B) + B

Fig OM 2
Imagen binaria.

Fig OM 8
Aplicación del elem
Isótropo en la

10.1.4.- Cierre sobre imágenes binarias.

El cierre es una operación morfológica que consiste en una dilatación seguida de una erosión. La imagen original no se recupera y el resultado es una imagen más simplificada y menos detallada que la original.

X crr B = (X + B) x B

Fig OM 2
Imagen binaria.

Fig OM 9
Cierre con un elemento isótropo

La apertura y cierre con un elemento estructural isótropo se utiliza para eliminar detalles específicos de la imagen más pequeños que el elemento estructural. La forma global de los objetos no se distorsiona.

Como se puede observar en las imágenes de las dos páginas anteriores, el cierre conecta objetos que están próximos entre sí, rellena pequeños huecos y suaviza el contorno del objeto rellenando los pequeños valles mientras que la apertura produce el efecto contrario. Los conceptos de pequeño y próximo están relacionados con la forma del elemento estructural.

10.1.5.- Contorno sobre imágenes binarias.

Las operaciones morfológicas básicas pueden usarse para encontrar los contornos de los objetos. Una forma de obtenerlos es mediante la operación lógica XOR de la imagen original con la imagen dilatada.

Veamos un ejemplo:

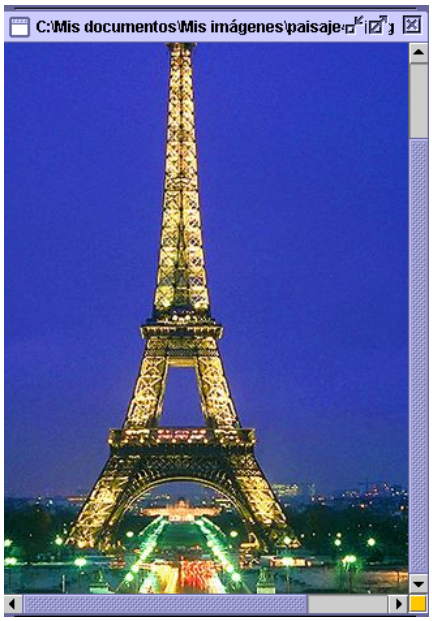


Imagen original

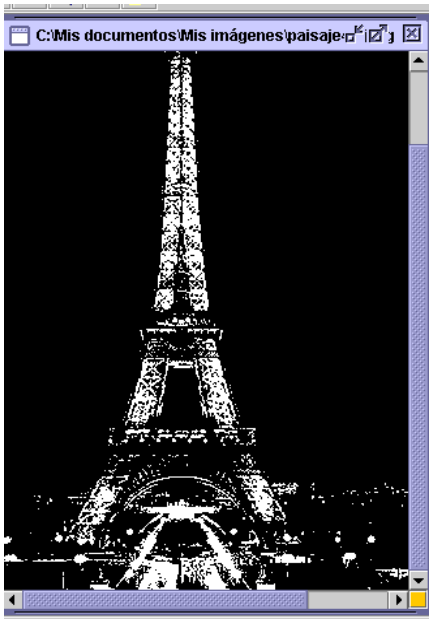
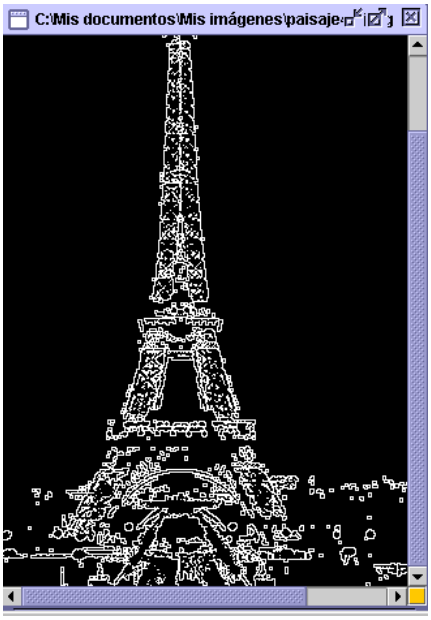


Imagen binarizada



Contorno de la imagen binarizada con operador isótropo.

10.1.6.- Esqueleto sobre imágenes binarias.

Esta operación ofrece la posibilidad de obtener los bordes de las regiones combinando apertura y erosión mediante una resta.

La operación es la siguiente:

$$Bordes = X \circ B - X \otimes B$$

Podemos ver la diferencia que existe entre la imagen obtenida en el contorno y el esqueleto. El contorno ofrece una imagen mucho más detallada que el esqueleto, puesto que la operación de esqueleto realiza una apertura, con lo cual se obtiene una imagen más simplificada y menos detallada con la original, y una erosión, con la que las regiones blancas reducen su tamaño en todas las direcciones puesto que el operador utilizado es isótropo.

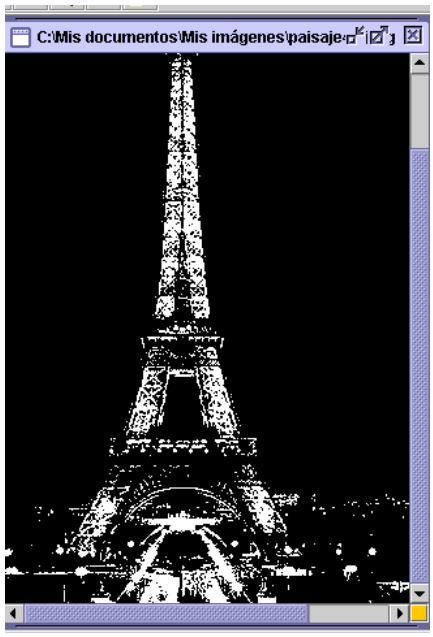
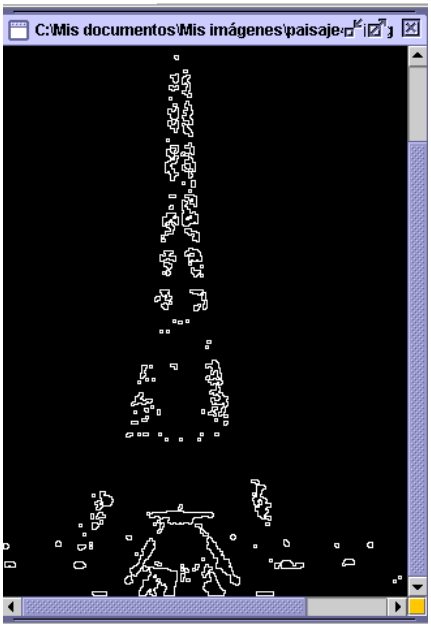


Imagen binarizada



Esqueleto de la imagen binarizada

10.1.7.- Dilatación y erosión sobre imágenes de grises.

Las imágenes en niveles de gris vienen dadas como f(x, y) de dimensión M x N y el elemento estructural como b(i, j), que es realmente una subimagen de dimensión m x n.

La dilatación se define como:

$$(f + b)(x,y) = \max \{ f(x-i, y-j) + b(i,j) \}$$

con $0 <= i <= m-1$ y $0 <= j <= n-1$.

La erosión se define como:

$$(f \times b)(x,y) = \min \{ f(x+i, y+j) - b(i,j) \}$$

con $0 <= i <= m-1$ y $0 <= j <= n-1$.

El elemento estructural utilizado en la implementación es una matriz de 2x2 llena de unos.

Imagen original.

Efecto de la dilatación.

En la página anterior se puede observar el efecto de una operación de dilatación sobre una imagen de niveles de gris.

Puesto que la dilatación está basada (según la ecuación antes expuesta) en la elección del valor máximo de f + B en una vecindad definida por la forma del elemento estructural, el efecto general al realizar la dilatación en la imagen de gris consiste en que la imagen dilatada es más brillante y los detalles oscuros son reducidos o eliminados dependiendo del elemento estructural utilizado.

Se puede ver también como la dilatación ha afectado a los letreros del edificio de la imagen “Grundig” y “Sweppes”.

A continuación se analiza el efecto de la erosión sobre la imagen de grises original:

Imagen original.

Efecto de la erosión.

Puesto que la erosión está basada (según la ecuación antes expuesta) en la elección del valor mínimo de f – B en una vecindad definida por la forma del elemento estructural, el efecto general al realizar la erosión en la imagen de gris consiste en que la imagen erosionada es más oscura y los detalles brillantes son reducidos dependiendo del elemento estructural y los niveles de gris que rodean a los detalles brillantes.

Se puede ver también como la erosión ha afectado a los letreros del edificio de la imagen “Grundig” y “ Sweppes”.

10.1.8.- Apertura y cierre sobre imágenes de grises.

La definición de estas operaciones es análoga al caso de las imágenes binarias utilizando las definiciones de dilatación y erosión correspondientes a las imágenes de niveles de gris.

A continuación se observa el efecto de la apertura en la imagen original:

Imagen original.

Efecto de la apertura.

La apertura provoca la reducción de los detalles brillantes pequeños, sin un efecto apreciable sobre los niveles de gris oscuros.

En la siguiente imagen se aprecian los efectos de una operación de cierre:

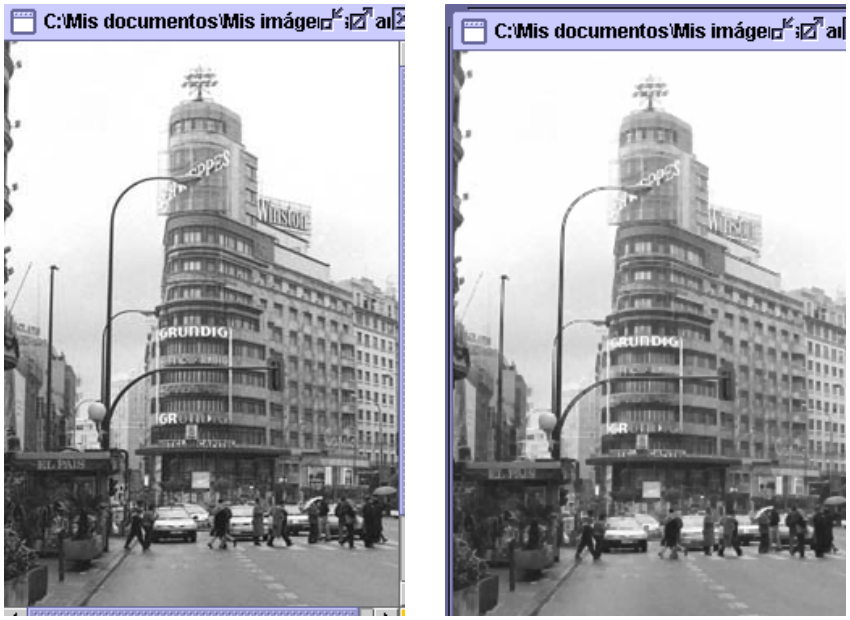


Imagen original. Efecto del cierre.

La apertura provoca la reducción de los detalles oscuros pequeños, sin un efecto apreciable sobre las características brillantes.

En algunos de los ejemplos las diferencias son muy sutiles pero al aplicar la operación correspondiente usando la aplicación se advierten perfectamente los cambios sobre la imagen original.

10.1.9.- Contorno y Esqueleto sobre imágenes de grises.

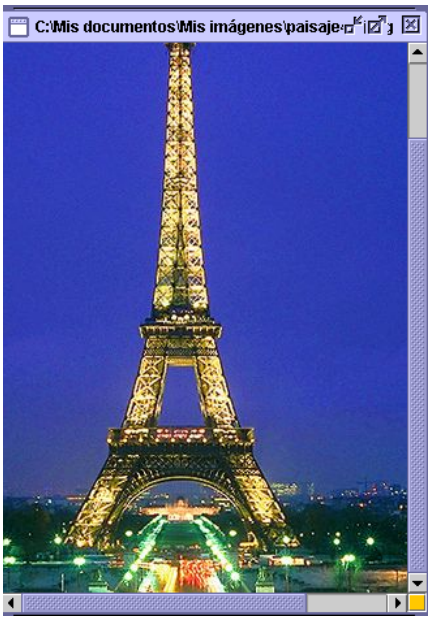


Imagen original

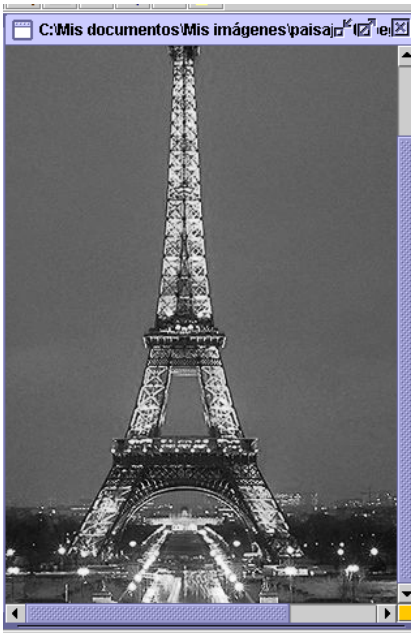


Imagen de grises.

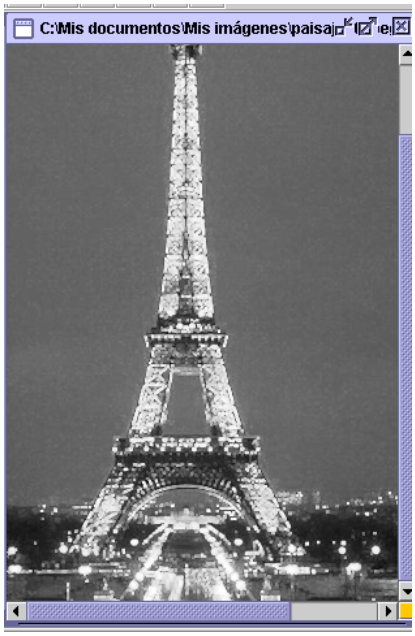
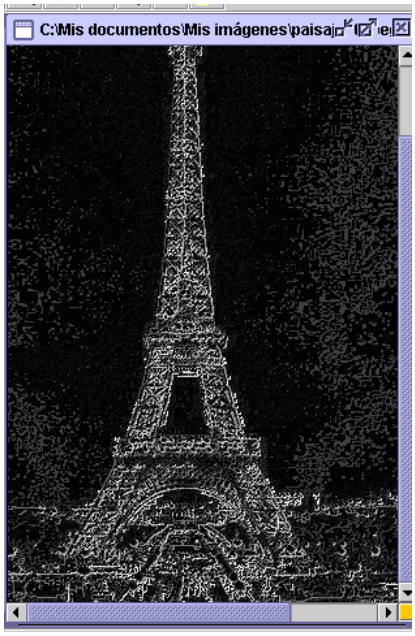
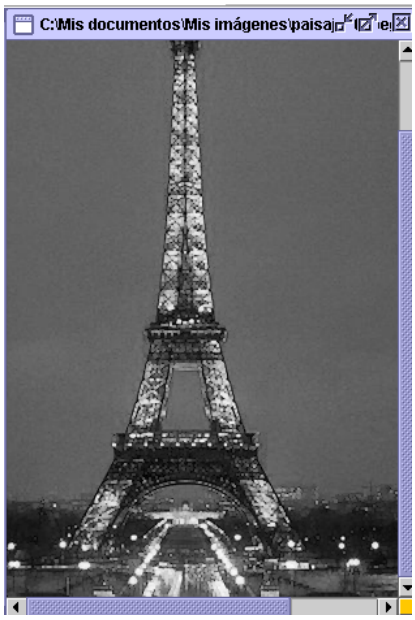


Imagen de grises dilatada.

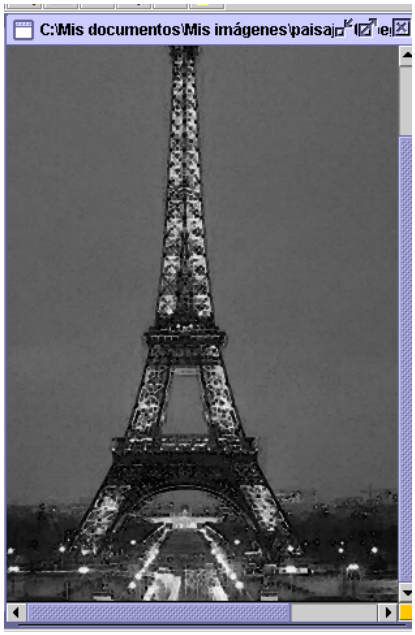


Contorno de la imagen de grises: Imagen original XOR Imagen dilatada.

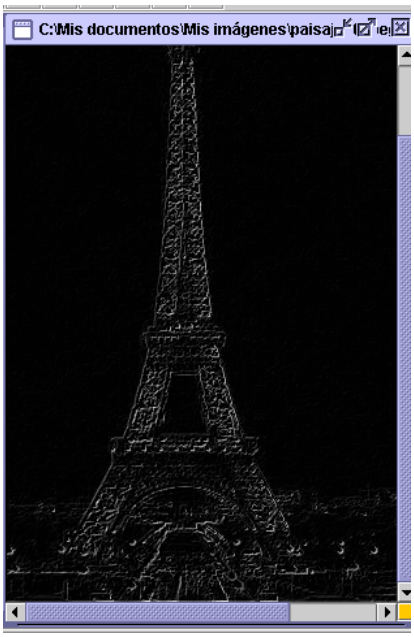
Como vemos la imagen dilatada es más brillante y los detalles oscuros son reducidos o eliminados, al hacer la XOR de la imagen dilatada con la imagen original obtenemos una imagen mucho más oscura y con mucho ruido.



Apertura sobre la imagen de gris.



XOR de la imagen dilatada con la original (de grises)



Esqueleto:
Imagen dilatada –
Imagen erosionada.

Como podemos ver con la resta de la imagen dilatada con la imagen erosionada obtenemos los bordes de la torre de una forma muy fina y el fondo negro, esto es debido a que sustracción muestra la diferencia entre ambas imágenes y como la primera es más clara que la segunda (aunque no mucho), los valores obtenidos son muy pequeños y por tanto negros, de aquí el resultado obtenido. Como la sustracción trabaja con números positivos, si al hacer la resta obtenemos números negativos estos se ponen como 0 (es decir, negros) y si superan el 255 se ponen como 255 (es decir, blancos). Aunque también se puede optar por coger el valor absoluto del resultado.

10.2.- NUEVOS FILTROS

10.2.1.- Filtrado de Máximos y Mínimos.

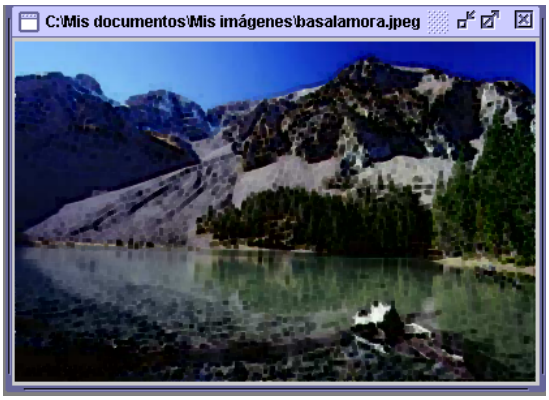
Los filtros máximo y mínimo son dos filtros de orden que se pueden utilizar para eliminar ruido de tipo impulso (sal y pimienta). El filtro máximo selecciona el mayor valor dentro de una ventana ordenada de valores de píxeles, mientras el de mínimo selecciona el valor más pequeño. El filtro de mínimos trabaja cuando el ruido es exclusivamente de tipo sal (valores altos), y el filtro de máximo a la inversa, es decir con ruido de tipo pimienta (valores pequeños).

Veamos como se comportan ambos filtros con un ejemplo:

Imagen original.

Filtrado de Máximos sobre la imagen original.

Como podemos observar el filtrado de máximos selecciona el valor mayor dentro de la ventana de píxeles y por ello la imagen se vuelve más clara, ya que cuanto mayor es el valor más se aproxima a 255 y por tanto a blanco.



Filtrado de mínimos sobre la imagen original.

Como podemos observar el filtrado de mínimos selecciona el valor menor dentro de la ventana de píxeles y por ello la imagen se vuelve más oscura, ya que cuanto menor es el valor más se aproxima a 0 y por tanto a negro.

Exactamente los mismos resultados se producen para imágenes de grises y binarias, el filtro máximo aclara la imagen y el filtro mínimo la oscurece.

10.2.2.- Filtrado de Gauss.

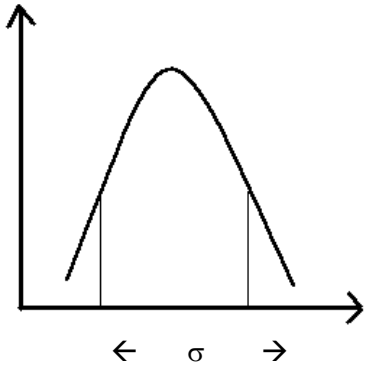
El filtro de Gauss es una de las mejores operaciones de suavizado que existen. El suavizado se utiliza fundamentalmente para eliminar ruido.

La máscara que aplicamos a la imagen se obtiene de la siguiente fórmula:

$$G(x,y) = e^{-(x^2 + y^{2)}/2\sigma}$$

En dos dimensiones la gráfica obtenida sería como un sombrero mejicano. La dimensión de la máscara elegida y el valor de sigma (σ) son elegidos por el usuario, siendo la dimensión un número impar, puesto que la máscara es una matriz simétrica n x n, que va desde –n / 2 hasta n / 2, por tanto basta calcular los valores para uno de los cuadrantes y luego calcular el resto de la matriz.

La forma de la gráfica obtenida para este filtro de suavizado es la siguiente:



Cuanto mayor sea el valor de sigma mayor el ancho de la campana y por tanto pasan más valores. Una vez obtenida la máscara sumamos todos los valores obtenido y dividimos cada valor por esta suma, para que los valores no se salgan de rango, si no obtendríamos un efecto demasiado luminoso (muy blanco).

Comprobamos el efecto de aplicar la convolución de esta máscara a la imagen y el resultado obtenido para dimensión 9 y σ = 0.6 es el siguiente:

Imagen original.

Imagen suavizada

Aunque en el documento no se aprecia con mucha claridad el efecto del filtro, en la aplicación se aprecian mucho mejor los cambios originados. La imagen se ve más borrosa. El borde negro que aparece en la imagen obtenida se origina por la forma que tienen las JAI de aplicar la convolución, ya que no se la aplica a los bordes.

Los resultados obtenidos para imágenes de grises son similares a los de color.

10.3.- OPERACIONES DE TRANSFORMACIÓN DE HISTOGRAMA

Estas técnicas van principalmente enfocadas a mejorar la visualización de una imagen. El histograma de una imagen es un gráfico que ofrece una descripción global de la apariencia de la misma, en el eje de abcisas se representa el rango de valores de píxeles, mientras que el eje de ordenadas se representa el rango de valores que pueden tomar esos píxeles.

El histograma de una imagen nos da la distribución de sus niveles de gris o color, es una función discreta que representa el numero de píxeles en la imagen en función de los niveles de intensidad, g. La probabilidad P(g) de la ocurrencia de un determinado nivel g se define como,

$$P(g)=\frac{N(g)}{M}$$

Donde M es el número de píxeles en la imagen y N(g) es el número de píxeles en el nivel de intensidad g. Al ser una función de probabilidad todos los valores de P(g) son menores o iguales que 1 y la suma de todos los valores de P(g) es 1.

Para la realización de cada una de estas técnicas de expansión de histograma hemos utilizado las funciones que nos aporta la **JAI (Java Advanced Imaging)** de Java para un histograma que están ya implementadas lo que facilita su manejo aunque para la representación de los resultados de la transformación del histograma y su plasmación en la respectiva nueva imagen hemos tenido que tratar las imágenes píxel a píxel de la misma forma que en las operaciones morfológicas. De igual forma para acceder píxel a píxel a una imagen se han utilizado las clases de la JAI como Raster, RenderedImage..., realizando recorridos sobre la matriz de píxeles que compone la imagen.

También de la misma forma que en las operaciones anteriores, la aplicación de las distintas operaciones de transformación del histograma es contextual a la imagen que se encuentre en el *frame activo* en un determinado momento de la aplicación.

Para cada una de las distintas operaciones se sigue el mismo esquema de presentación de manera que en primer lugar se muestra el histograma referido a la imagen que se encuentra en el *frame activo,* posteriormente se aplica la transformación seleccionada por medio del menú y por último se muestra de

nuevo el histograma modificado y se modifica la imagen mostrando los cambios en el *frame* actual.

También es de mencionar que para todas éstas operaciones los cambios aplicados al histograma son válidos tanto en el caso de imágenes grises como de imágenes a color RGB, y no se aplican a imágenes binarias ya que no tiene mucho sentido pues solo se representarían dos valores en el histograma blanco y negro y sobre éstos no hay cambios que realizar en el histograma.

Para que se pueda apreciar claramente los cambios que se van produciendo en el histograma y en la respectiva imagen a la que representa hemos elegido una imagen de muestra e iremos mostrando los resultados que se producen sobre la misma al aplicar las distintas técnicas de modificación del histograma. Presentamos la imagen original y el histograma que representa tanto para su versión de grises como para color RGB, y posteriormente la imagen modifica junto con su histograma también modificado.

10.3.1 Expansión de histograma:

La expansión de histograma es la operación que toma una imagen de entrada f y expande el histograma a lo largo del rango de valores completo de los niveles de gris o color. Como tal esto provoca el efecto de incrementar el contraste de una imagen de bajo contraste. Si se desea que la expansión no cubra el rango total posible de niveles de gris o color, podemos especificar unas cotas o valores máximo y mínimo para los niveles de gris o color, tomados como MAX y MIN.

Así la expansión de un histograma para una imagen de grises es una función que se puede definir de la siguiente manera:

$$g(i,j) = \left[\frac{f(i,j) - f(i,j)_{MIN}}{f(i,j)_{MAX} - f(i,j)_{MIN}} \right] [MAX - MIN] + MIN$$

donde:
f(i,j) es el nivel del gris de la imagen de entrada;
$f(i,j)_{MAX}$ es el mayor valor del nivel de gris en la imagen de entrada f;
$f(i,j)_{MIN}$ es el menor valor del nivel de gris en la imagen de entrada f;

MAX y MIN corresponden al máximo y mínimo de los valores posibles de los niveles de gris(para una imagen de 8 bits serían 0 y 255).

En la mayoría de las imágenes tratadas algún valor cae en los niveles máximo y mínimo del histograma de forma que éste abarca todo el rango de posibles valores por lo que una expansión pura del histograma no mejora ni cambia la imagen ya que el límite MAX y MIN es casi siempre el máximo y mínimo posible. Por ello hemos recortado el nivel de gris de los extremos mas bajo y mas alto. De esta forma hemos realizado una comprobación en la operación de expansión del histograma y es que si en el rango de valor de los píxeles que van de 0 a 10 (es decir en los más extremos), el número de píxeles encontrados es mayor que una cantidad mínima(en nuestro caso hemos acotado a 100) entonces recortamos los niveles de gris de nuestra imagen a 50 por abajo y 200 por arriba y solo consideramos los píxeles encontrados en esa región. De esta forma expandimos el histograma hasta ocupar el rango de valores total de 0 a 255 observándose los cambios en la imagen.

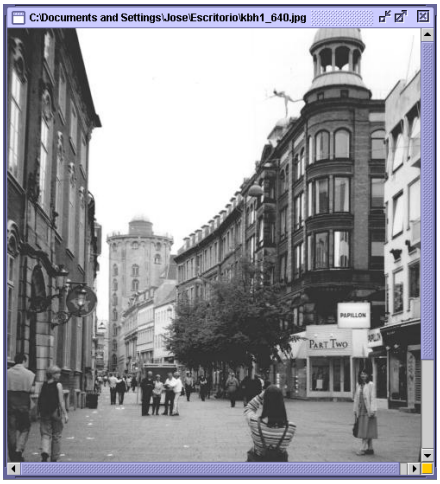


Fig HIS 1
Imagen grises original

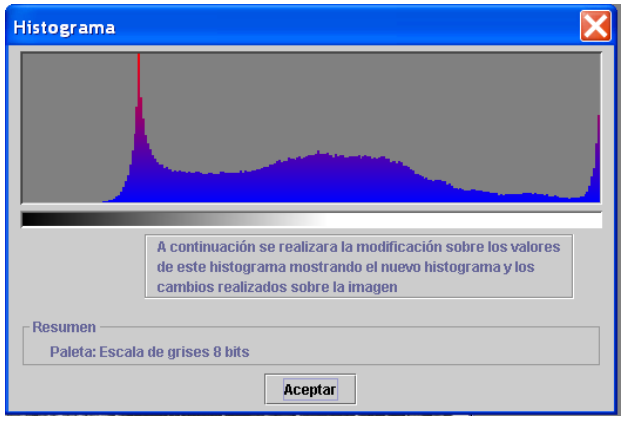


Fig HIS 2
Histograma
Imagen grises
original

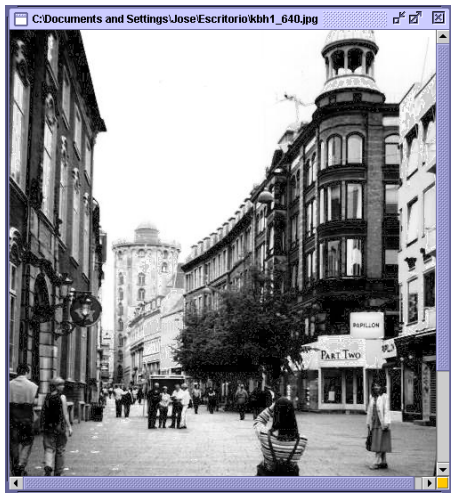


Fig HIS 3
Imagen grises
expansión

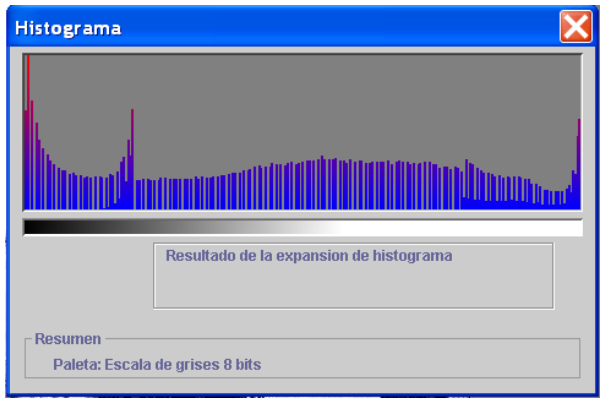


Fig HIS 4
Histograma
Imagen grises
expansión

En el caso de una imagen a color el procedimiento es similar con la diferencia que en la imagen de color RGB se aplicarían los cambios a casa uno de los componentes distintos de la imagen (al componente R, al componente G y al componente B), y de nuevo la combinación de los tres produciría la nueva imagen modificada apreciando los cambios.

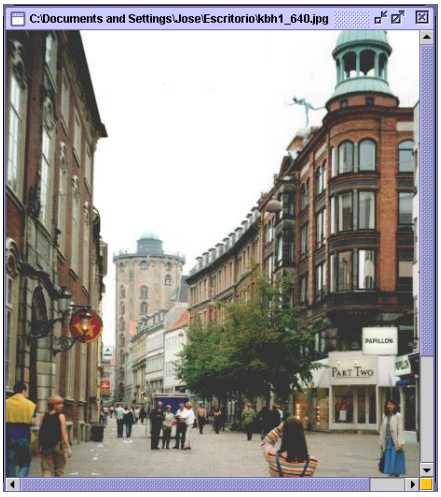


Fig HIS 5
Imagen color
RGB original

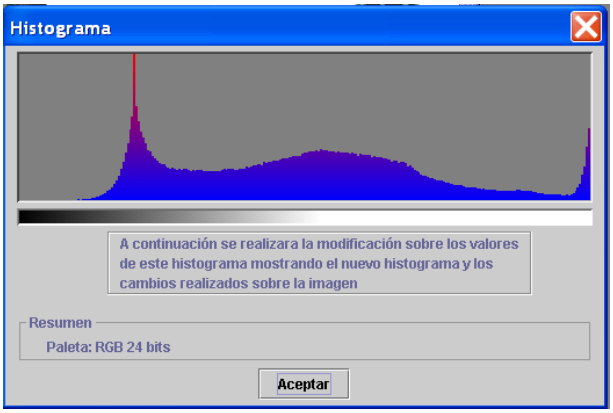


Fig HIS 6
Histograma
Imagen color
RGB original

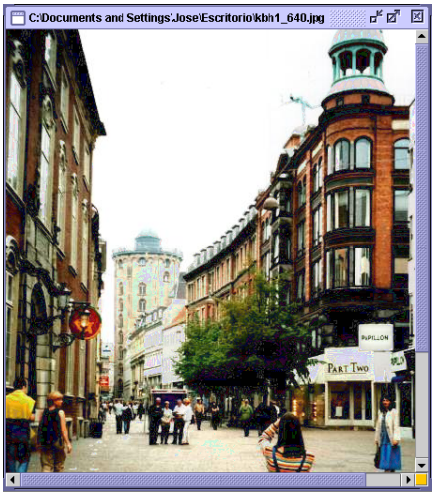


Fig HIS 7
Imagen color
RGB expansión

Fig HIS 8
Histograma
Imagen color
RGB expansión

Como se puede observar el efecto es incrementar el contraste de la imagen tanto para la imagen de grises como la de color al cubrir todo el rango posible de colores.

- Para las siguientes técnicas de modificación de histograma hemos elegido una imagen común para todas y sobre la que se observan muy bien los cambios. Únicamente hemos elegido otra como se ha visto anteriormente para el caso de la expansión del histograma ya que la imagen original abarca prácticamente todo el rango de valores de 0 a 255 y el cambio al expandir el histograma apenas se iba a notar.

  Así primeramente vamos a presentar la imagen original junto con su histograma tanto para grises como para color RGB ya que es común a todas las demás técnicas que vamos a presentar a continuación.
  En cada técnica mostraremos únicamente la imagen ya transformada y el histograma también modificado con la aplicación de la modificación de histograma que corresponda:

Fig HIS 9
Imagen grises original

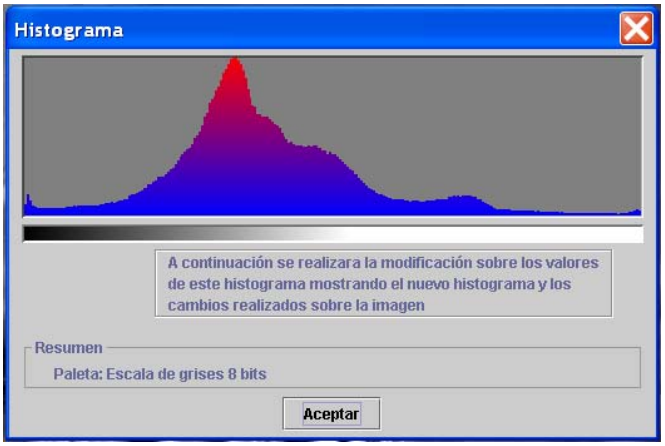


Fig HIS 10
Histograma
Imagen grises
original

En el caso de la misma imagen pero en color RGB

Fig HIS 11
Imagen RGB
original

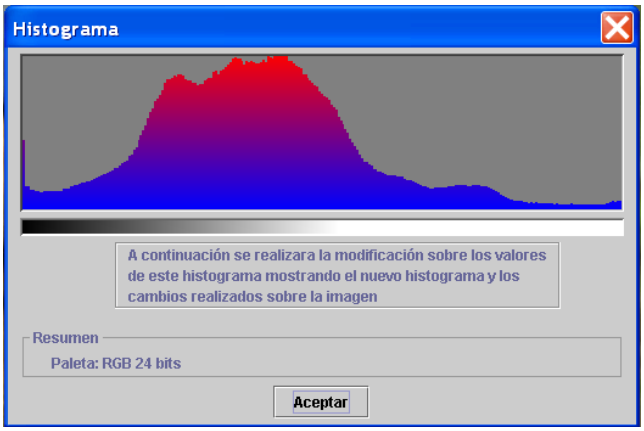


Fig HIS 12
Histograma
Imagen RGB
original

10.3.2 Desplazamiento del histograma:

El desplazamiento del histograma se usa para aclarar u oscurecer una imagen pero manteniendo la relación entre los valores de los niveles de gris o color.

Modificar el brillo de una imagen es añadir una cantidad constante al valor del tono de cada uno de los píxeles de la misma. Si este valor es positivo estamos aumentando el brillo de la imagen, mientras que, si por el contrario es negativo, el brillo disminuye. Evidentemente si al sumar la constante se pasa de 255, se queda el píxel en ese valor. Lo mismo si baja de 0 se queda en negro. Por tanto las modificaciones de brillo corren el riesgo de perder información de la imagen.

Esta operación puede llevarse a cabo por la simple adición o sustracción de un número fijo a todos los valores del nivel de gris mediante la función siguiente:

$$f(i,i) = h(i,j) + DESP$$

donde:
f(i,j) es el nuevo valor del píxel
h(i,j) es el antiguo valor del píxel
DESP es el desplazamiento que le aplicamos a cada uno de los píxeles para desplazar el histograma

Se asume aún con pérdida de información que los valores que sobrepasan el máximo y el mínimo se redondean al máximo y al mínimo de los valores permitidos. Un valor DESP positivo incrementa el brillo de la imagen mientras que un valor DESP negativo la oscurece.

Por ello hemos considerado los dos tipos de desplazamiento posibles hacia la izquierda y hacia la derecha y un ejemplo de aplicación de los mismos es el siguiente:

10.3.2.1.-Desplazamiento a la izquierda de histograma:

Fig HIS 13
Imagen grises desplazada a la izquierda

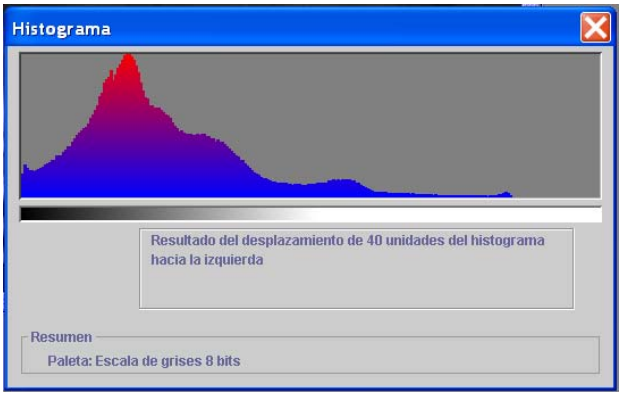


Fig HIS 14
Histograma imagen grises desplazada a la izquierda

Como aparece indicado en la nueva imagen del histograma se le ha aplicado al histograma un desplazamiento de 40 unidades a la izquierda y el resultado en la imagen como se puede observar es un oscurecimiento ya que desplazamos todos los valores 40 unidades hacia la región de los valores oscuros.

En el caso de la imagen RGB el procedimiento es similar pero aplicado a cada una de las regiones R, G y B que componen la imagen como se ha comentado anteriormente. El resultado sobre la imagen y el histograma es el siguiente:

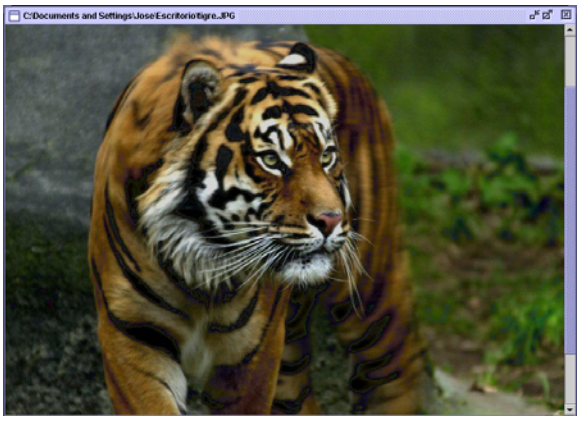


Fig HIS 15
Imagen color RGB desplazada a la izquierda

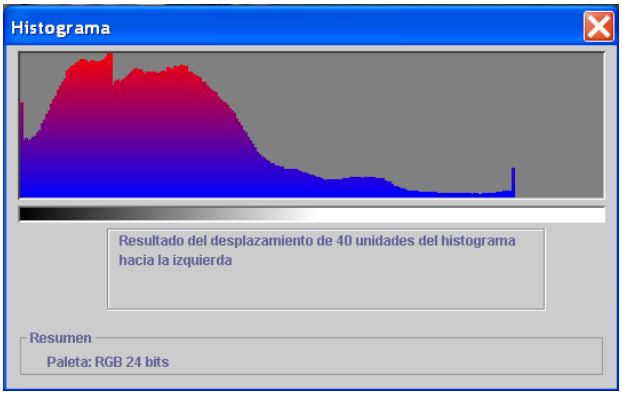


Fig HIS 16
Histograma imagen color RGB desplazada a la izquierda

10.3.2.2.-Desplazamiento a la derecha de histograma:

Fig HIS 17
Imagen grises desplazada a la derecha

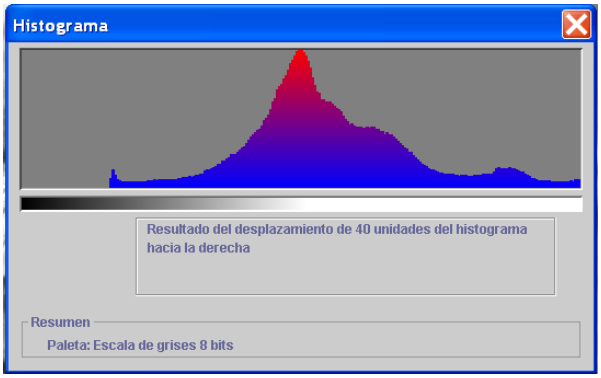


Fig HIS 18
Histograma imagen grises desplazada a la derecha

De la misma forma que el caso anterior el desplazamiento del histograma 40 unidades a la derecha produce un aclaración de la imagen al desplazar todos los píxeles 40 unidades hacia la zona de los valores mas claros.
Lo mismo ocurre en la imagen a color RGB apreciando una aclaración de la imagen y el correspondiente desplazamiento de los niveles de color del histograma.

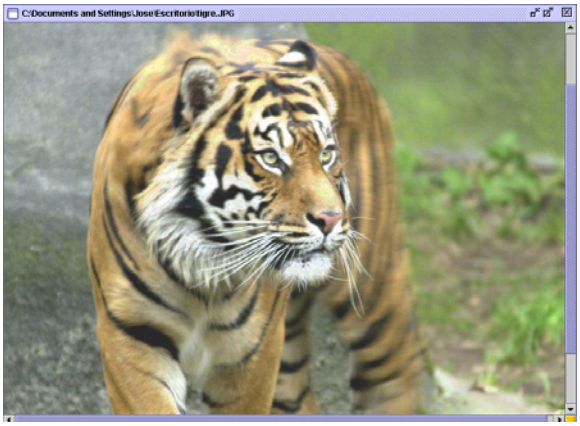


Fig HIS 19
Imagen color RGB desplazada a la derecha

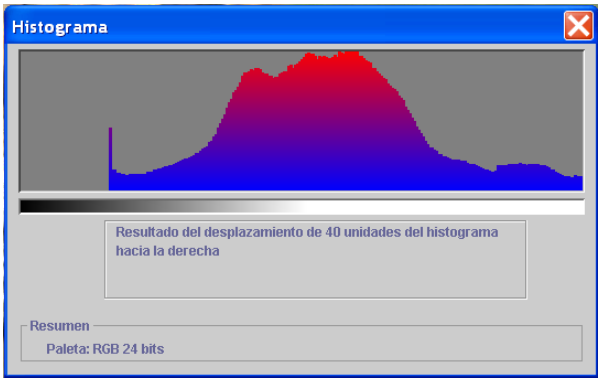


Fig HIS 20
Histograma imagen color RGB desplazada a la derecha

10.3.3.-Contracción del histograma:

La técnica es similar a la expansión del histograma pero de efecto contrario. No produce un realzado de la imagen sino una disminución del contraste de la imagen.
La función que define la transformación que realiza es la siguiente

$$g(i,j) = \left[\frac{C_{MAX} - C_{MIN}}{f(i,j)_{MAX} - f(i,j)_{MIN}} \right] [f(i,j) - f(i,j)_{MIN}] + C_{MIN}$$

donde:
f(i,j) es el nivel del gris de la imagen de entrada;
$f(i,j)_{MAX}$ es el mayor valor del nivel de gris en la imagen de entrada f;
$f(i,j)_{MIN}$ es el menor valor del nivel de gris en la imagen de entrada f;

C_{MAX} y C_{MIN} corresponden al máximo y mínimo valores deseados en la compresión del histograma.

Fig HIS 21
Imagen grises contracción

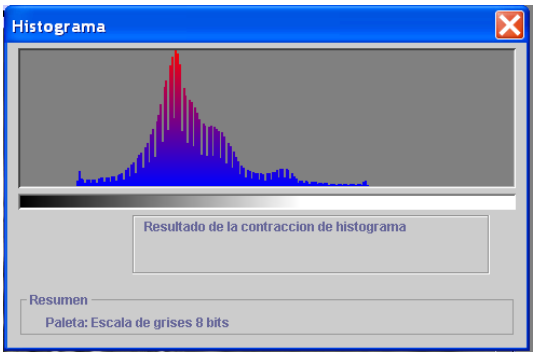


Fig HIS 22
Histograma
imagen grises
contracción

Como se observa el resultado es que la imagen pierde parte de su contraste apareciendo como más difuminada. En el caso de una imagen a color RGB la técnica es la misma pero aplicando la operación sobre las tres matrices de píxeles correspondientes a R,G y B como se comentó para las otras técnicas.
El resultado que se aprecia es el siguiente.

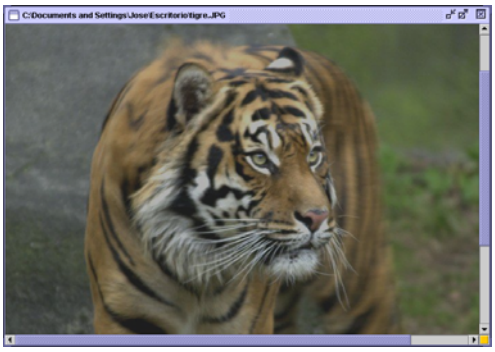


Fig HIS 23
Imagen color RGB
contracción

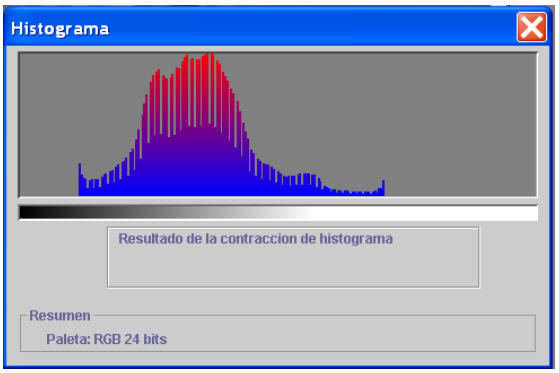


Fig HIS 24
Histograma
imagen color RGB
contracción

10.3.4.-Ecualización de histograma:

La ecualización de histograma es una de las técnicas más usadas para la mejora del contraste de la imagen original. Esta técnica realza la imagen original mediante una determinada modificación o transformación del histo_
grama denominada ecualización o igualación del histograma.

Lo que se busca es encontrar una función F(g) que realce el contraste general en la imagen original expandiendo la distribución de los niveles de gris o para los niveles R,G y B en el caso de una imagen a color RGB.
Esta expansión debe de ser suave en el sentido que idealmente debería haber el mismo número de píxeles por niveles de gris. El objetivo de una ecualización de un histograma es distribuir los niveles de gris (ó color en el caso RGB) de una manera uniforme a lo largo de todo el rango de valores de los niveles de gris.

Considerando los valores de intensidad máximo y mínimo en el rango de niveles de gris de la imagen dada, g_{max} y g_{min} respectivamente, y considerando una función de distribución de probabilidad uniforme,

$$F(g) = [g_{max} - g_{min}] P_g(g) + g_{min}$$

siendo $P_g(g) = \sum_{g=0}^{s} p(g)$

En términos generales y a modo de resumen cuando aplicamos una ecualización de histograma sobre una imagen original el efecto que se produce con una distribución de densidad uniforme es una imagen realzada.

Fig HIS 25
Imagen gris ecualización

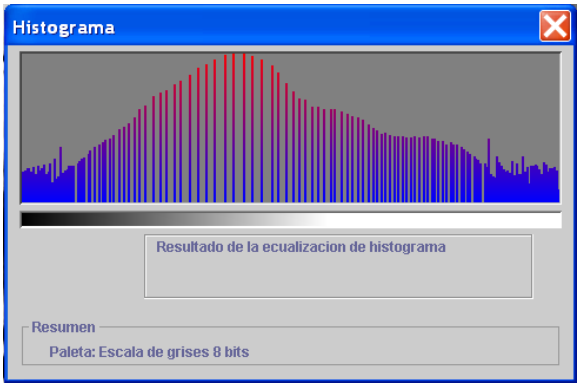


Fig HIS 26
Histograma
imagen gris
ecualización

Para el caso de la imagen a color RGB el resultado es el mismo apreciándose una variación de color hacia tonos mas fuertes y mas realzados.

Fig HIS 27
Imagen color RGB
ecualización

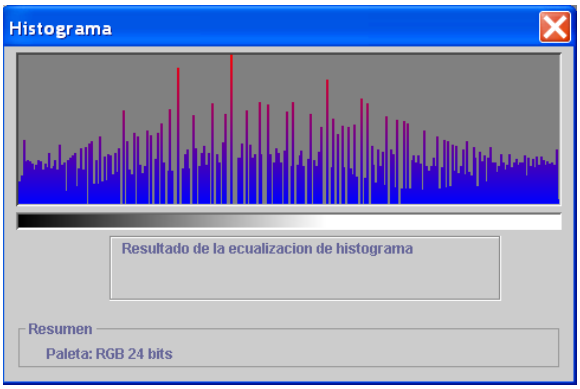


Fig HIS 28
Histograma
imagen color RGB
ecualización

10.4.- OPERACIONES ESPECIALES.

Conforman la parte opcional del proyecto y están contenidas en el menú "Aplics." del menú principal del sistema. Como se explicará mas adelante estas operaciones pueden tener aplicación en el campo de la medicina.

Se han implementado 4 operaciones especiales que se pueden dividir en dos grupos, las binarizaciones especiales, que consisten en binarizar imágenes con distintos parámetros y criterios y el cálculo de momentos invariantes para regiones especificadas en imágenes binarias.

Al igual que todas las operaciones disponibles en el sistema, las implementadas en el menú "Aplics" se realizarán sobre la imagen contenida en el marco o *frame* activo. Así, si tenemos varios marcos abiertos con distintas imágenes, tendremos que seleccionar primero el que contiene la imagen sobre la cuál queremos realizar la operación y después seleccionar la opción con la operación que se quiera en el menú.

A continuación se explican las operaciones especiales implementadas:

10.4.1.- Binarización especial 1.

Esta operación consiste en la binarización de la imagen teniendo en cuenta dos parámetros seleccionados por el usuario. Solamente se puede realizar sobre imágenes de color RGB de forma que si se intenta aplicar con otro tipo de imagen en el frame activo, se mostrará un cuadro de dialogo informando de la situación y la operación no se llevará a cabo.

El primer parámetro es el formado por los valores RGB de un píxel seleccionado por el usuario. Antes de pulsar sobre la opción del menú correspondiente a esta operación, el usuario clickeará con el ratón sobre el píxel de la imagen que desee (botón izquierdo) y se capturarán las coordenadas x e y así como los valores R (rojo), G (verde) y B (azul).
En la barra de estado se mostrarán las coordenadas del píxel clickeado.
El segundo de los parámetros es un umbral, elegido por el usuario entre 0 y 255. Los errores de todo tipo están controlados y el sistema informa cuando el valor no es adecuado y está fuera de rango, no se introduce un número, etc...

Una vez indicados estos parámetros se lleva a cabo la operación. Consiste en recorrer la imagen de arriba abajo y de izquierda a derecha y calcular la distancia de cada uno de los píxeles que se va recorriendo con los valores RGB seleccionado con el ratón por parte del usuario. Esa distancia se calcula de la siguiente forma:

Distancia = | x - R | + | y – G | + | z – B |

siendo los valores RGB los correspondientes al píxel seleccionado y siendo x, y, z respectivamente los valores RGB de los píxeles que se van recorriendo.

Sabiendo la distancia del píxel seleccionado a todos los de la imagen se produce la binarización de la misma en función de si esa distancia es mayor o no que el umbral definido por el usuario. De esta forma, aquellos píxeles con distancia mayor o igual al seleccionado que el umbral serán blancos en la imagen resultado y los que no serán negros.

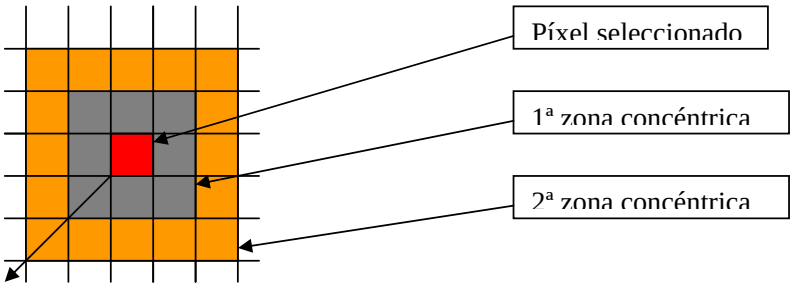
10.4.2.- Binarización especial 2 (Umbral decreciente).

Al igual que la operación anterior se requiere una imagen de color RGB, si no es así el programa lo indicará y la operación no se realizará.

Se trata de otra binarización parametrizada. De la misma manera que en el caso anterior, el usuario tiene que seleccionar un píxel con el ratón antes de elegir la opción del menú correspondiente a esta operación así como un umbral entre 0 y 255.

Lo que diferencia esta operación de la anterior es el recorrido de la imagen, que en este caso se realiza en “zonas concéntricas” alrededor del píxel que ha sido seleccionado con el ratón por el usuario. Además a medida que en ese recorrido nos vamos alejando del píxel seleccionado, el umbral elegido se irá decrementando. El tercer parámetro de esta operación es precisamente el número que se quiere que se decremente el umbral por cada nueva “zona concéntrica” que se recorre.

Esquema del recorrido:



A medida que se recorre una nueva zona de píxeles el umbral va disminuyendo X unidades, siendo X el valor especificado por el usuario.

Este tipo de recorrido tiene una complejidad exponencial de forma que dependiendo del tamaño de la imagen, puede durar alrededor de un minuto y medio. El sistema informa convenientemente de esta circunstancia para que no se crea que se ha bloqueado. En el frame activo, aparece la imagen resultado binarizada, transcurrido el tiempo.

10.4.3.- Binarización especial 2 (Umbral creciente).

Se trata de una operación análoga a la anterior. La única diferencia es que a medida que se avanza el recorrido en zonas concéntricas el umbral

definido se va incrementando X unidades en cada nueva zona, siendo X especificada por el usuario. El efecto de estas binarizaciones es por tanto distinto a las del tipo 1.

Este tipo de binarizaciones de las imágenes puede tener su aplicación en el campo de la medicina. Por ejemplo se puede someter a una de estas operaciones una imagen de una vena o arteria con colesterol en las paredes y determinar con distintos umbrales y clickeando en distintos píxeles en que zonas se encuentra el colesterol respecto del fondo.

En este sentido se pueden utilizar para obtener regiones claramente diferenciadas en las imágenes con distintos criterios.

10.4.4.- Momentos invariantes de regiones.

Es la última opción del menú de operaciones especiales y requiere imágenes binarias. Es lógico, si se trata de calcular características y momentos de las regiones de una imagen, habrá que proporcionar una imagen en la que se puedan estudiar esas regiones. Si no se dispone de una imagen binaria en el frame activo a la hora de seleccionar esta opción del menú se informa con un cuadro de diálogo.

Si la imagen si es binaria, al pulsar esta opción del menú, el sistema entra en un modo especial, del que se informa al usuario.
A partir de este momento el usuario deberá seleccionar aquellas regiones blancas de las cuales desee conocer los momentos. Si se selecciona alguna región negra se informará que hay que seleccionar obligatoriamente regiones blancas. La selección se realizará pinchando con el botón izquierdo del ratón sobre la región deseada. En la barra de estado se indican las coordenadas del píxel seleccionado.

Cuando se selecciona una región, esta se colorea de gris para indicar con un cierto contraste la selección realizada (muchas veces las regiones se extienden mas de lo que parece que percibimos). Acto seguido aparece un
formulario en el que se pide que el usuario introduzca unos valores “p” y “q” mayores o iguales a 1 y que son parámetros necesarios para el cálculo de los momentos.

Una vez realizada la selección de los parámetros para una región determinada, ya coloreada en gris, el sistema muestra un formulario con toda la información calculada que se comentará posteriormente.

Se pueden seleccionar todas las regiones que uno desee para ver los momentos de cada una con diferentes parámetros p y q, de modo que siempre que se selecciona una nueva región, esta se colorea en gris mientras que la anterior pasa a tener su antiguo color blanco. De esta manera queda resaltada en todo momento la región seleccionada en último lugar.

Si estando en el modo del sistema indicado anteriormente, es decir que hemos pulsado la opción del menú relativa a momentos invariantes con una imagen binaria, no se saldrá de dicho modo hasta que el usuario lo haga expreso. Después de haber visto las propiedades de las regiones deseadas pinchando en ellas con el botón izquierdo del ratón, se abandonará este "modo" pulsando el botón derecho del ratón sobre cualquier parte de la imagen del frame activo. El sistema informará de que se abandona este modo y a partir de este momento el botón derecho no tendrá ninguna funcionalidad a menos que volvamos a entrar de dicho modo y la pulsación del izquierdo tampoco mostrará los momentos calculados. Únicamente se mostrarán las coordenadas del píxel clickeado con el botón izquierdo en la barra de estado del sistema (parte inferior de la pantalla).

Todas las propiedades calculadas para las regiones y mostradas en el formulario al que se alude anteriormente se encuentran descritas en el libro "Visión por computador" indicado en la bibliografía, por lo que no se va a explicar aquí como se realiza su cálculo. Sin embargo si se exponen a continuación aquellos valores calculados, y que se muestran en el formulario cuando se pincha una región con el botón izquierdo del ratón y se eligen los valores p y q:

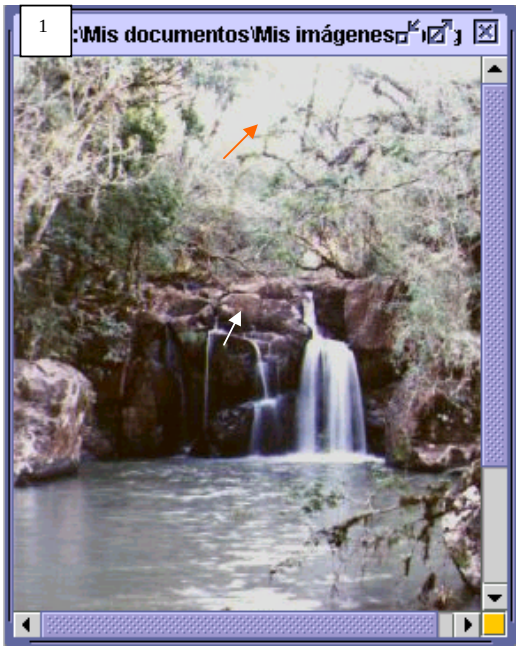
- Media de las X's
- Media de las Y's
- Momento de orden p+q m(p,q)
- Momento de orden central u(p,q)
- Momento de orden central normalizado n(p,q)
- Y los 7 momentos invariantes de Hu. (Hu(1).....Hu(7))

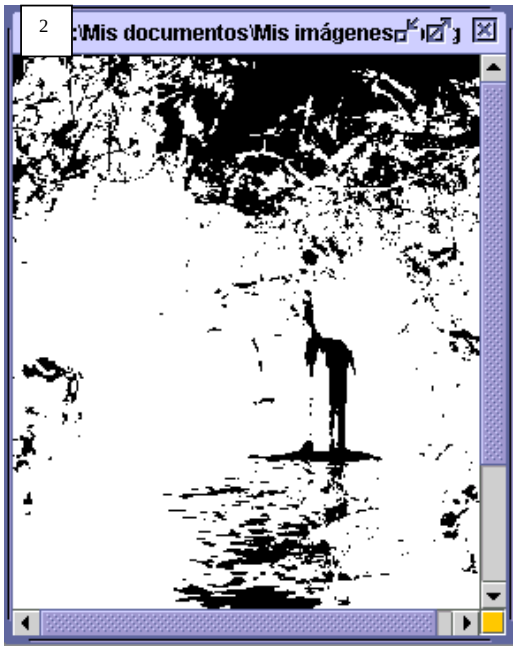
La particularidad de los momentos de Hu es que son invariantes a las rotaciones, traslaciones y cambios de escala de la imagen, lo cual hace que sean unas propiedades importantes de las regiones. Sus valores serán constantes para la misma región aunque sometamos a la imagen en la que se encuentre dicha región a las operaciones mencionadas de rotación, traslación o cambio de escala.

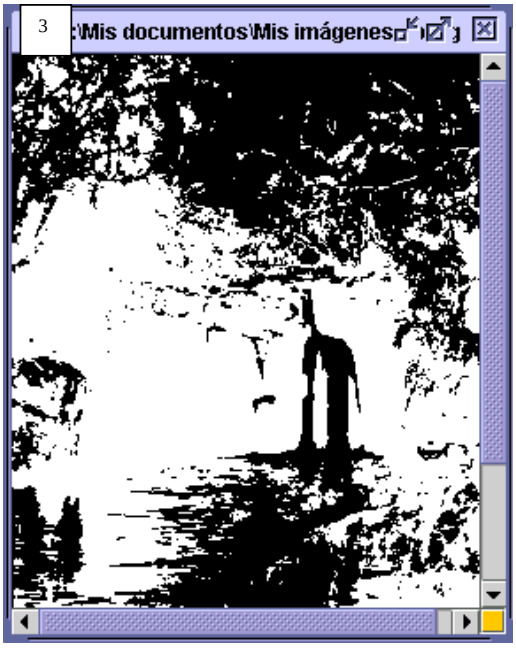
5.- Ejemplos de operaciones sobre imágenes.

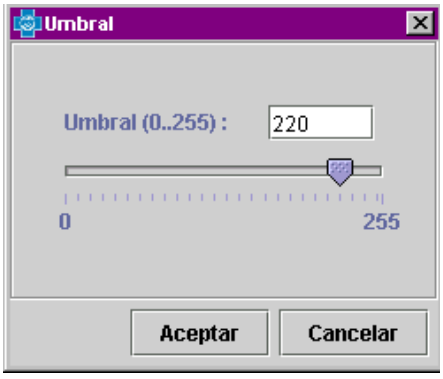
A continuación se muestran ejemplos de operaciones especiales sobre imágenes y se comentan determinados aspectos.

Ejemplo de **binarización especial 1**:



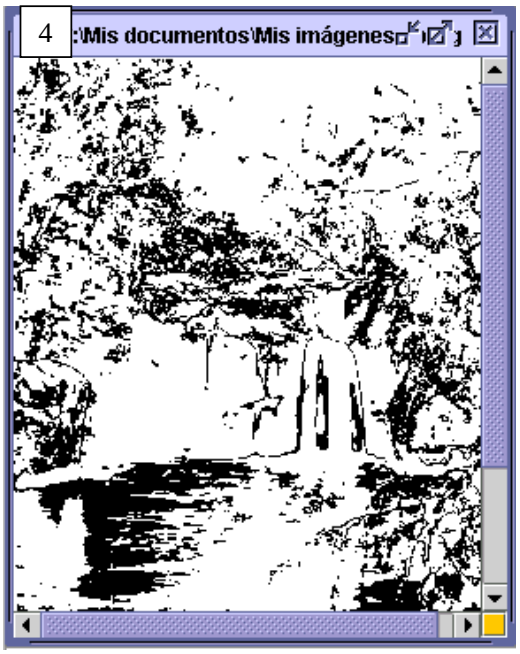


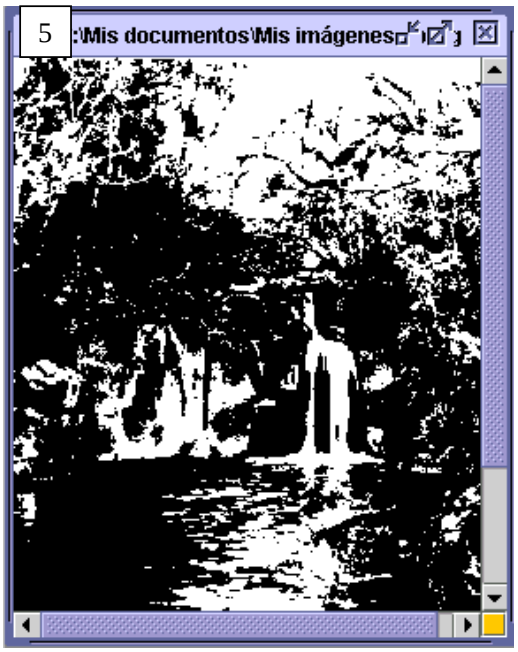




En la página anterior se muestra la imagen original indicada con un **1** en la que se ha seleccionado el píxel (x=125, y=40), punto que esta marcado con una flecha de color rojo.

También se muestra un cuadro de diálogo para la selección del umbral por parte del usuario. En la imagen nº**2** el umbral seleccionado es 100 mientras que en la imagen nº**3**, el umbral es 220. Se puede observar como en la imagen 3 las regiones blancas son menores que en la 2. Esto es así porque el umbral es mayor en el segundo caso. Recordando que la binarización convierte a blancos los píxeles con distancia mayor o igual que el umbral, si el umbral es mas grande el número de píxeles blancos será menor.





En las dos imágenes de arriba se ha realizado la binarización pinchando en otro píxel, en este caso del de coordenadas (x=127, y=138) señalado aproximadamente en la imagen original (página anterior nº 1) con una flecha de color blanco. Al igual que con el ejemplo del píxel anterior, en la imagen **4** se considera un umbral de 100 y en la imagen **5** se considera el umbral de 220.
El píxel que se seleccionó en primer lugar es verde claro en contraposición al que se eligió en segundo lugar, marrón oscuro. Lógicamente las distancias de los demás bits son distintas en cada caso y se aprecia que las binarizaciones son distintas.

En el caso de la imagen 4 el color oscuro del píxel seleccionado implica que sus valores RGB son bajos y las distancias de los demás con el son grandes. Si tenemos en cuenta que el umbral no es muy alto (100) y casi todas las distancias son grandes el resultado es que la mayoría de píxeles son blancos puesto que su distancia con el píxel seleccionado es superior al umbral. Al aumentar el umbral en

la imagen nº**5** crecen las regiones negras. También se observa que la catarata por ejemplo antes era negra (respecto del píxel verde indicado con la flecha roja) y ahora es blanca (respecto del píxel señalado con la flecha blanca). Esto es consecuencia de que el segundo píxel es mas oscuro y sus valores RGB son muy bajos. Al ser la catarata de color blanco o casi blanco, sus valores RGB serán muy cercanos a 255. Esto implica que en las imágenes 4 y 5 las distancia de los píxeles de la catarata respecto del seleccionado es muy grande y supera en los dos casos el umbral (100 y 220) siendo por lo tanto en la imagen resultado, la catarata de color blanco.

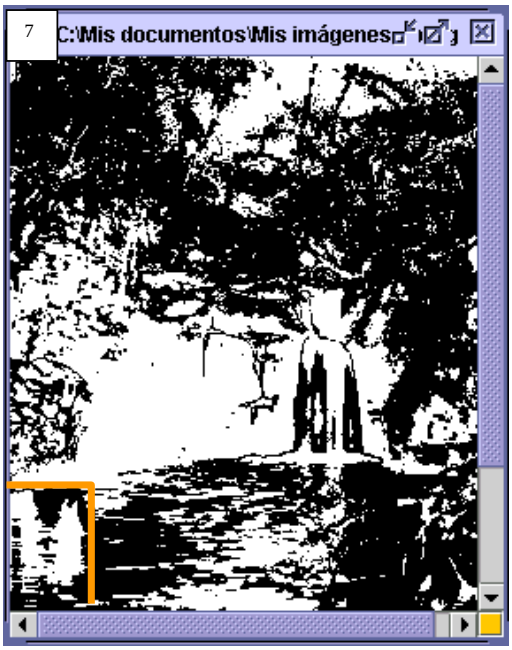
Sin embargo en las imágenes 2 y 3, la catarata en la imagen resultado es negra porque al ser los valores RGB del píxel seleccionado bastante mas grandes, las distancias respecto de este de los demás píxeles es menor. Al no sobrepasar esas distancias el valor del umbral, los píxeles aparecen negros en la imagen resultado.

A continuación se muestra algún ejemplo de la **operación de binarización especial 2:**

La flecha roja indica el píxel seleccionado, de coordenadas (x=42, y=276). Para realizar la operación se ha elegido un umbral inicial de 25 y se ha seleccionado el incremento en +3 por cada nueva zona de píxeles concéntricos que se va recorriendo. Se observa que las regiones correspondientes a las copas de los árboles y las zonas del agua en las que no hay reflejo de la luz se convierten a negros al se la distancia de sus píxeles inferior al umbral

En torno al píxel seleccionado se puede ver una pequeña región blanca que indica que los píxeles en torno al seleccionado son muy parecidos en valores. El

umbral aumenta a medida que nos alejamos de la flecha. Llega un momento que aunque los píxeles cercanos al seleccionado son parecidos en valores, al haber crecido el umbral su distancia es menor al mismo y por tanto se convierten en negros.



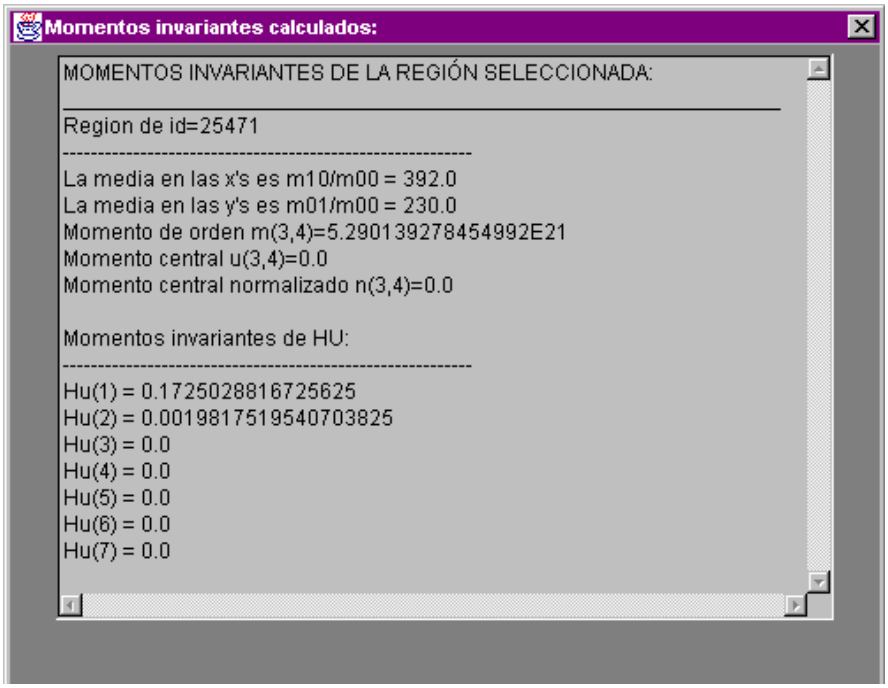
La diferencia de la imagen **7** con la **6** es que el incremento del umbral a medida que se avanza es +1, es decir, un incremento 3 veces menor al caso de la imagen 6. Se puede ver que la región blanca alrededor del píxel seleccionado en este caso es bastante mas grande porque aunque el umbral vaya creciendo lo hace mas lentamente lo que permite que mas píxeles superen el umbral en las cercanías del seleccionado y por lo tanto la región blanca en torno a el sea mayor

Para hacer notar la circunstancia comentada, las regiones en torno al píxel seleccionado aparecen rodeadas de un recuadro de color naranja.

Por último se expone a continuación un ejemplo del **cálculo de momentos invariantes** para una región. En este punto hay que destacar un aspecto que no se comentó anteriormente cuando se explicó la operación. Para poder calcular los momentos relativos a cualquier región seleccionada con el ratón el sistema hace una representación de la imagen binaria que consiste en un algoritmo de etiquetado tras el cual todos los píxeles blancos de la imagen tienen un identificador distinto de 0. Todos los píxeles de una misma región tienen la misma etiqueta, distinta de las demás regiones. De esta forma se controla en todo momento los píxeles que pertenecen a cada región.

En la página siguiente se muestra una imagen original correspondiente a una bandada de aves. Después se muestra dicha imagen convertida en binaria. Las regiones blancas se corresponden con las aves y la que aparece en color gris es la que ha sido seleccionada para el cálculo de sus momentos invariantes.

Los valores P y Q elegidos son 3 y 4 respectivamente. En la página siguiente se muestra el formulario con los cálculos que el sistema muestra cuando se han introducido los valores de P y Q.



En este caso se ve como muchos de los valores son tan pequeños que la p y la q se han aproximado por 0. Si los valores de p y q son muy grandes el tiempo de cálculo se incrementa notablemente. Para valores relativamente pequeños la respuesta es rápida. También se puede ver el identificador de la región, la etiqueta que tienen todos sus píxeles y mediante la cual son reconocidos para hacer los cálculos.

11.- LAS BIBLIOTECAS JAI (JAVA ADVANCED IMAGING) DE SUN MICROSYSTEM

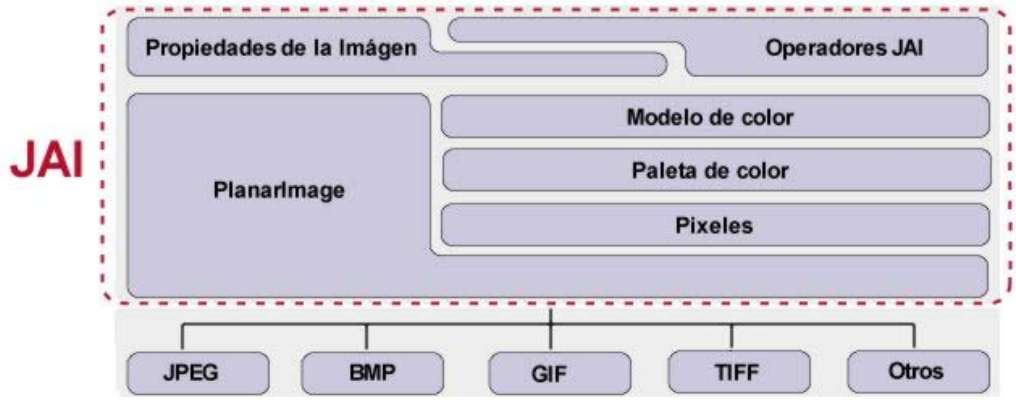
11.1.- CARACTERÍSTICAS GENERALES

Java ofrece diversas características que lo hacen especialmente apto para el desarrollo de una aplicación de estas características: Java es seguro, escalable, independiente de la plataforma, e ideado para la red. Existen dos tecnologías diferentes en la actualidad proporcionadas por Sun Microsystems para la gestión de la imagen: **Java Media Framework** (para la captura, procesado y transmisión de datos de audio y video en tiempo real) y **Java Advanced Imaging** (para el procesado avanzado de imágenes independientes).

Las primeras versiones de Java AWT proporcionaban un simple paquete de *renderización* adecuado para construcción de páginas HTML comunes, pero sin las características necesarias para imágenes complejas. Los primeros AWT permitían la generación de imágenes sencillas mediante el dibujo de líneas y formas. Un número muy limitado de ficheros de imagen, tales como GIF y JPEG podían ser leídas mediante el uso de un objeto Toolkit. Una vez leída, la imagen podía ser mostrada, pero esencialmente no había operadores de manipulación de imágenes.

El API Java 2D extendió esos primeros AWT añadiendo soporte para operaciones más generales sobre manipulación y *render* de gráficos. Java 2D añadió clases gráficas especiales para la definición de primitivas geométricas, trazado de texto y definición de fuentes, espacios de color y *renderizado* de imágenes. Las nuevas clases soportaban un conjunto limitado de operaciones de proceso de imágenes para realizar transformaciones geométricas, suavizado y perfilado, mejora del contraste y definición del umbral. Las extensiones de Java 2D fueron añadidas al núcleo de Java AWT a partir de la versión 1.2 de la plataforma Java.

El API Java Advanced Imaging (JAI) extiende más allá la plataforma Java (incluyendo el API Java 2D) permitiendo que operaciones de proceso de imágenes sofisticadas y de alto rendimiento sean incorporadas en aplicaciones (locales o remotas) y applets. JAI es un conjunto de clases que proporcionan funcionalidades de tratamiento de imágenes más allá de aquellas del Java 2D y de las Java Foundation Classes (JFC), aunque es compatible con ellos.



JAI pretende satisfacer las necesidades de cualquier aplicación que manipule imágenes. El API pretende ser fácilmente extensible, permitiendo que nuevas operaciones de proceso de imágenes sean añadidas de manera que aparenten ser parte nativa de él. De esta manera, JAI beneficia virtualmente a todos los desarrolladores Java que quieren incorporar imágenes en sus aplicaciones y applets.

JAI encapsula los diferentes formatos de imagen y las invocaciones a métodos remotos con un objeto de imagen, permitiendo un archivo de imagen, un objeto de imagen de red, o una imagen transmitida en tiempo real procesándolas todas ellas de la misma forma. De esta manera, JAI representa un modelo simple de programación ahorrándonos los complejos mecanismos internos del tratamiento de imágenes.

11.2.- LAS JAI EN NUESTRA APLICACIÓN

Dentro de la aplicación SDTI hemos empleado varias bibliotecas de JAI, en concreto y principalmente las referidas a las clases PlanarImage y TiledImage, así como las operaciones de bajo nivel que proporciona, divididas en varias categorías principales:

- Operadores de punto (Point Operators): Las operaciones de punto transforman una imagen de entrada en una imagen de salida de manera que cada píxel de la imagen de salida depende solamente del correspondiente píxel de la imagen de entrada.
- Operadores de área (Area Operators): Los operadores de área realizan transformaciones geométricas, que resultan en el reposicionado de píxeles dentro de una imagen. Utilizando transformaciones matemáticas, se obtienen las coordenadas de los píxeles en la imagen de salida a partir de las

coordenadas x e y de la imagen de entrada.
Hay dos tipos básicos de operaciones de área: Lineares y no lineares. Las operaciones lineares incluyen la traslación, la rotación, y el escalado. Las operaciones no lineares, también conocidas como *warping transformations*, introducen curvaturas y ángulos a la imagen procesada.

- Operadores geométricos (Geometric Operators): Los operadores geométricos permiten modificar la orientación, tamaño y forma de una imagen.
- Operadores de cuantificación de color (Color Quantization Operators): La cuantificación de color, también conocida como dithering, se emplea para efectuar transformaciones de color sobre los pixeles de la imagen de entrada.
- Operadores de fichero (File Operators): Los operadores de fichero se emplean para leer o escribir ficheros de imagen.
- Operadores de frecuencia (Frequency Operators): Los operadores de frecuencia se usan para descomponer una imagen desde su forma de dominio espacial a una forma del dominio de la frecuencia. También se proporcionan operadores para realizar la transformación inversa, en la cual la imagen se convierte desde el dominio de la frecuencia de nuevo al dominio espacial.
- Operadores estadísticos (Statistical Operators): Los operadores estadísticos proporcionan los medios para analizar el contenido de una imagen.
- Operadores de extracción de bordes (Edge Extraction Operators): Los operadores de extracción de bordes permiten el realzado de los bordes de una imagen. El realzado de bordes reduce una imagen para mostrar solamente sus detalles de bordes. El realzado de bordes se implementa a través de filtros espaciales que detectan una específica pendiente de brillo de píxel en un grupo de píxeles de una imagen. Una pendiente de brillo acusada indica la presencia de un borde.
- Operaciones de histograma (Histogram Operators): Operadores que permiten la generación y determinadas modificaciones de histograma. A partir de éstas operaciones podemos obtener por ejemplo el número de regiones, el número de píxeles que componen cada región del histograma y operaciones de manipulación.
- Otros operadores (Miscellaneous Operators): Aquí se engloban otros operadores que no encajan en ninguna de las categorías previas.

Empleando estos operadores, la filosofía que pretende seguir la API de JAI es proporcionar un sistema que permita evitar tratar directamente con el formato interno de la imagen, es decir, el formato de los píxeles, el modelo de color y la paleta de color.
Esta idea, que puede parecer la aproximación más adecuada a la hora de enfrentarnos al diseño de una aplicación de manipulación de imágenes, si bien resulta apropiada a la hora de manipular o realizar operaciones básicas, como las anteriormente comentadas, demuestra sus carencias, complejidad e ineficiencia al intentar implementar operaciones más complejas, ya que tanta encapsulación y aislamiento, perjudican a una aplicación de este tipo que busca tratar con el formato interno en muchas ocasiones. Este es el motivo que ha dado lugar a que para la implementación de la mayoría de las operaciones de complejidad mayor

Comentario [t1]: Sergio, el Histograma sería un buen ejemplo de esto?

como operaciones morfológicas, de histograma, operaciones especiales...aparte de estos operadores mas genéricos y proporcionados por el paquete de las JAI hemos empleado ciertas clases que ofrecen utilidades para acceder a la estructura píxel a píxel de la imagen. Entre estas están RenderedImagen, TiledImagen, Raster... que permiten lectura y escrituro concreta sobre un píxel de la imagen, implementando luego nosotros el algoritmo al completo de recorridos y modificación de estos píxeles.

Otro problema que hemos encontrado a la hora de emplear la API de JAVA es que al tratar nosotros con imágenes de diferentes formatos por ejemplos imágenes binarias, RGB, grises... los algoritmos y el tratamiento son distintos(una imagen de grises es una matriz de píxeles y una RGB son tres matrices de píxeles) implementando diferentes algoritmos según el tipo de imagen.

Así los diferentes paquetes que se presentan en la JAI son:
- javax.media.jai: contiene las principales interfaces y clases de JAI
- javax.media.jai.iterator: contiene las interfaces de los iteradores especiales y clases que se usan para escribir operadores de extensión.
- javax.media.jai.operator: contiene las clases que describen todos los operadores de imagen.

javax.media.jai.widget: contiene interfaces y clases para crear convoluciones sobre imágenes y ventanas para la representación de imágenes.

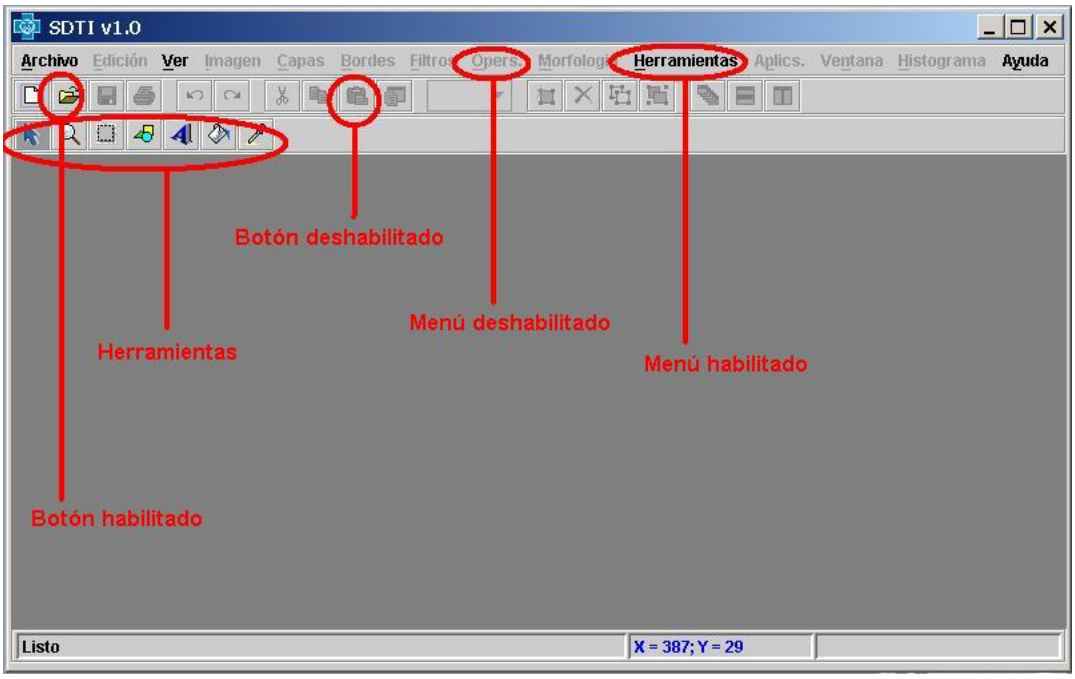
Para concluir no todo van a ser ventajas y uno de los inconvenientes de las JAI es la velocidad. la velocidad máxima de procesado no supera los 7 fotogramas por segundo. JAI optimiza cada operación básica, pero para el procesado de interés se ha debido construir una cadena bastante larga de operaciones simples, que hacen que el procesado total sea lento.
Las aplicaciones basadas en JAI tienen que ser capaces de tolerar pausas relativamente largas en el procesado (del orden de un segundo). JAI se puede usar para aplicaciones con procesado a baja velocidad pero por el momento no para aplicaciones en tiempo real, aunque se está avanzando en este campo.

12.- AMPLIACIÓN DE LA APLICACIÓN

12.1.- AÑADIR NUEVOS ELEMENTOS A LA INTERFAZ DE LA APLICACIÓN.

Con el objeto de minimizar la cantidad de errores que pudiera provocar el usuario al interactuar con la aplicación, decidimos que la interfaz permitiera hacer sólo aquello que tuviera sentido en cada momento. De este modo, si no se está procesando ninguna imagen en la aplicación – por ejemplo -, no se permite grabar, imprimir, aplicar filtros, etc.

Para ello debíamos conseguir que todos los elementos de la interfaz (menús, botones, items y herramientas) fueran sensibles tanto a los cambios de ventana dentro de la aplicación como a los cambios en las imágenes contenidas en las ventanas. Esto lo hemos conseguido con la creación del evento personalizado cl_eventoActivada y la interfaz – en el sentido de *esqueleto de clase Java* – i_listenerEventoActivada.



Los eventos los lanza el gestor de ventanas - cuando se cambia de ventana o se realiza una modificación en una de ellas - y contiene la información relevante que precisan los elementos de la interfaz para saber si deben habilitarse o deshabilitarse.

De este modo, haciendo que los elementos de la interfaz implementen un solo método – el definido en i_listenerEventoActivada - y declararlos como oyentes del gestor de ventanas para recoger los eventos, conseguimos que modifiquen dinámicamente su estado en función de la información contenida en los mismos.

Insertar un nuevo menú en la barra de menús.

Visto lo anterior, para añadir un nuevo menú a la barra de menús lo único que hay que hacer es:

1.- Crear una subclase de JMenu – la clase Java que implementa los menús - que implemente la interfaz i_listenerEventoActivada.

2.- Declarar en cl_ventanaPadre – la ventana que contiene la aplicación – el nuevo menú como oyente del gestor de ventanas.

Insertar un nuevo item en un menú.

Se hace igual que en cualquier otra aplicación Java con la salvedad de que se debe introducir en el método de configuración del menú (el que habilita y deshabilita sus elementos) el código para configurar el nuevo item.

Insertar un nuevo botón en la barra de botones.

Somos de la opinión de que todo botón debe corresponderse con algún item de algún menú. Así pues, aparecen cuatro problemas de mantenimiento de coherencia:

1.- Entre el estado del item y el del botón: o bien ambos están habilitados o bien están ambos inhabilitados.

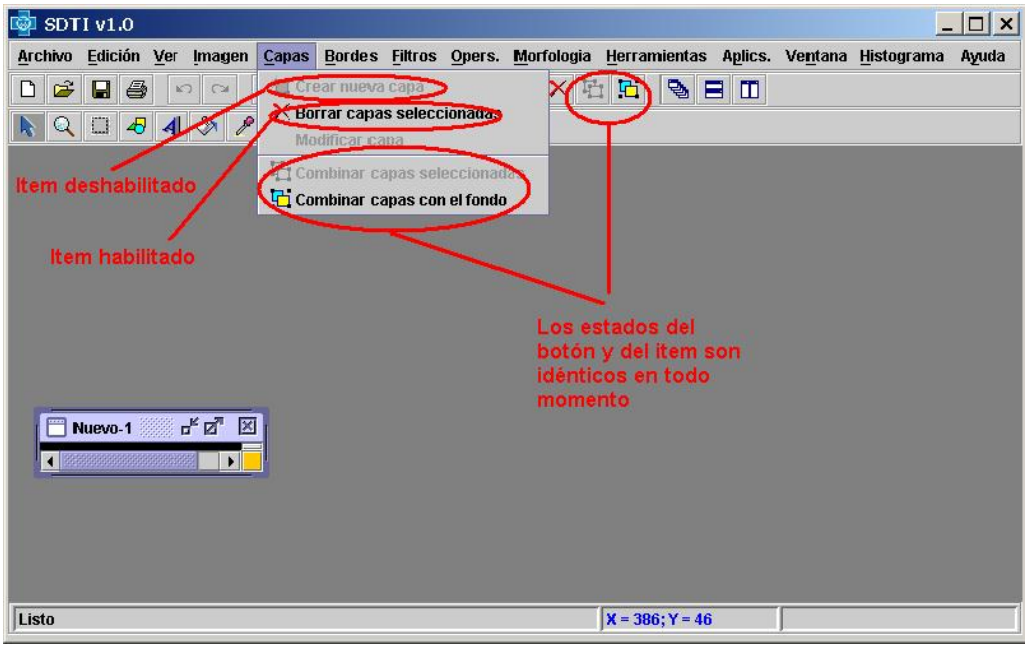
2.- Entre los nombres del item y del botón, así como del texto que aparece en la barra de estado: estos tres nombres son idénticos.

3.- Entre los iconos del item y del botón: deben ser idénticos.

4.- Entre las acciones a realizar al seleccionar el elemento: el resultado será el mismo tanto si seleccionamos el botón como si seleccionamos el item.

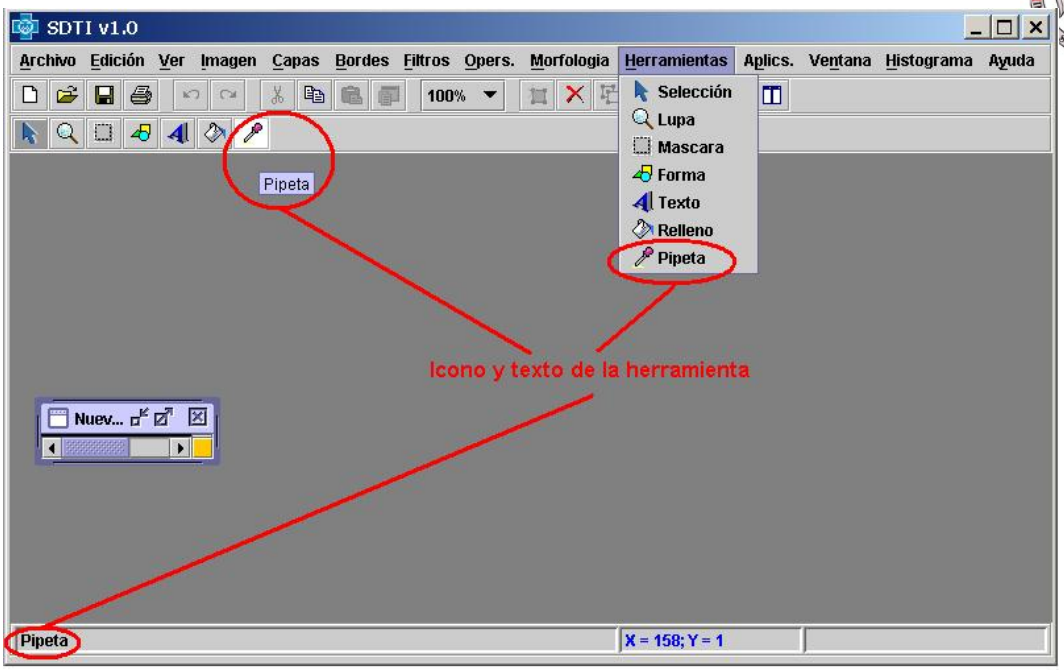
Para asegurarnos de estas tres características deseables se ha creado la clase cl_itemBoton, que reúne en un solo elemento lógico los dos elementos visuales de la interfaz a la vez que se encarga de mantener la coherencia deseada.

Así pues, para añadir un nuevo botón a la barra de botones, lo que haremos será declarar el item asociado como cl_itemBoton en lugar de cómo un JMenuItem – item simple de Java – y proceder como en el caso anterior.



Insertar una nueva herramienta.

Mención aparte merecen las herramientas de nuestra aplicación. Se han desarrollado las herramientas como Java Plug-Ins, esto es, clases que directamente se *acoplan* a cualquier aplicación gracias a que se autogestionan. Por lo tanto, para agregar una nueva herramienta - al menú y a la barra de herramientas - lo que debemos hacer es agregar un cl_itemHerramienta – análogo a cl_itemBoton, pero utilizando los datos del Plug-In para decidir la acción a realizar al seleccionar los elementos – al menú de herramientas como si fuera un item corriente.



12.2.- AÑADIR UN NUEVO PLUG-IN A LA APLICACIÓN

Los Plug-In representan una manera de añadir funcionalidades a la aplicación, por lo que este es otro de los aspectos que cabe considerar a la hora de continuar el trabajo sobre la aplicación, y por tanto el hecho de añadir un Plug-In debería ser lo más sencillo posible.

Así, el diseño de SDTI hace que para añadir un nuevo Plug-In a la aplicación baste con seguir cuatro sencillos pasos:

1. *Crear una clase que implemente los métodos abstractos de la clase base de los Plug-In's* (ATIM.plugIn.cl_pA_plugInBase). Esta nueva clase deberá formar parte del paquete ATIM.plugIn

2. *Crear un panel de configuración del Plug-In*.
   Este panel se integrará automáticamente dentro de la ventana de configuración de los Plug-In's.
   Una vez registrado el Plug-In en el gestor de Plug-In's, el gestor se encargará de colocar el panel de configuración dentro de la ventana de herramientas de la aplicación. Para facilitar su distinción, se pondrá el nombre del Plug-In como título de la ventana.

3. En la figura 1, se puede observar la ventana de configuración, que incluye el panel del Plug-In “Lupa”.

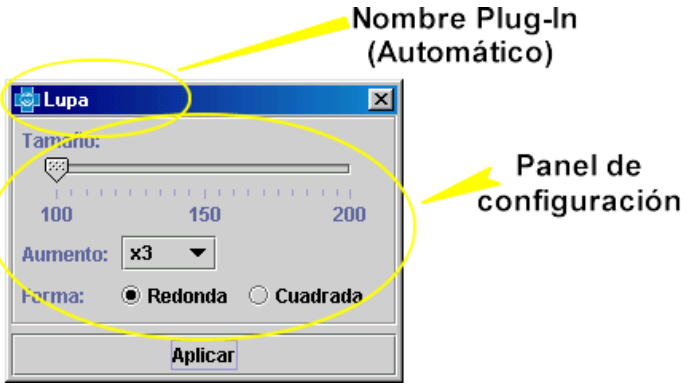


Figura 1

4. *Registrar el Plug-In en el gestor de Plug-In’s*

5. *Dar de alta el nuevo servicio en la aplicación*
   Para ello tan solo será necesario crear un nuevo botón y un nuevo menú (véase Anexo C).
   La figura 2 muestra los Plug-In que incorpora la aplicación.

Figura 2

Así pues siguiendo una serie de pasos indicados se pueden añadir nuevos Plug-In a la aplicación, completándola y ampliándola sin apenas necesidad de modificaciones sobre el código existente, con lo se simplifica al máximo la tarea de ampliación por parte de un equipo ajeno al responsable del desarrollo inicial.

12.3.- AÑADIR UNA NUEVA TRANSFORMACIÓN A LA APLICACIÓN

Entendemos por transformación cualquier operación que se pueda realizar sobre una imagen, ya sea el paso de un filtro, la detección de bordes, su binarización, etc.

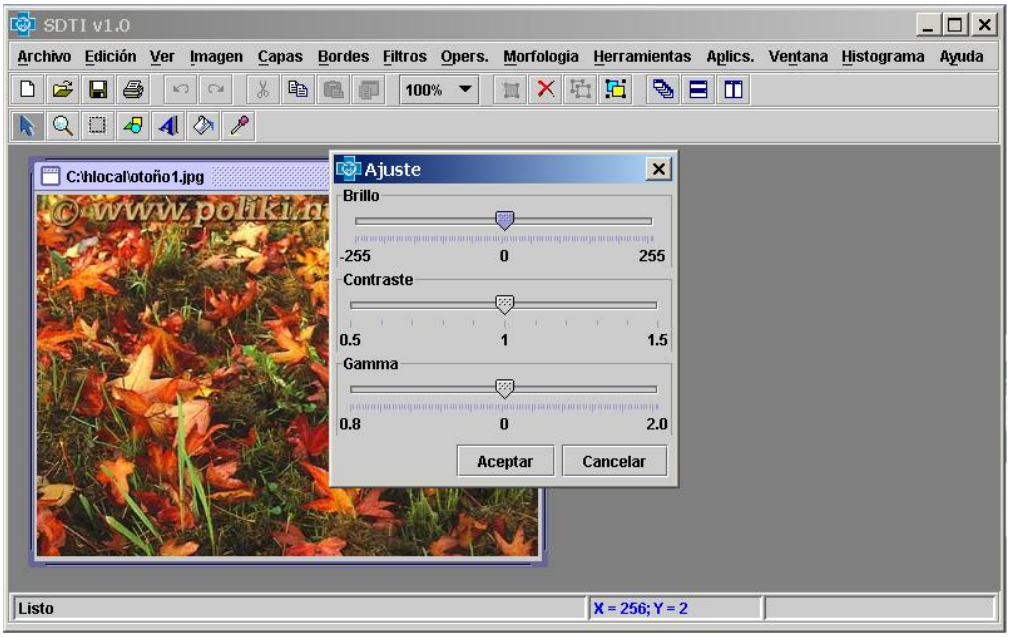
Las transformaciones serán fundamentales a la hora de añadir nuevas funcionales a la aplicación. Por ello hemos intentado que su forma de uso fuese lo más sencilla y abstracta posible, para así facilitar su uso y el futuro trabajo que se pueda realizar sobre la aplicación.

Para añadir una nueva transformación a la aplicación hay q seguir estos sencillos pasos:

1. *Crear una clase que implemente la clase base de las transformaciones* (ATIM.filtro.cl_pA_filtroBase). Para ello simplemente deberemos implementar la constructora y los métodos: m_pA_b_cofigurarFiltro y m_p_v_aplicarFiltro. El primero de dichos métodos configura la transformación, mediante la petición de datos al usuario usando un formulario, por ejemplo, y el segundo contiene el código en si de la transformación. Esta nueva clase deberá alojarse en el paquete ATIM.filtro

2. *Crear un formulario para configurar el filtro, si es necesario*. El formulario deberá formar parte del paquete ATIM.formularios.interfaz. Dicho formulario deberá tomar como base la clase cl_pA_formularioBase (declararse hijo de ella), lo cual nos facilitará en gran medida la creación del nuevo formulario. En la figura 1 se puede ver un ejemplo de la utilización de un formulario para configurar la transformación.

Con estos dos pasos ya tendríamos creada la transformación. Ahora quedaría integrarla en la aplicación, lo cual, debido a la abstracción utilizada es muy sencillo y transparente para la aplicación. Es decir, la aplicación trata cualquier transformación del mismo modo, no necesita saber que transformación va a aplicar. A continuación detallamos como integrar el filtro en la aplicación.



3. *Crear una nueva opción de menú desde donde poder invocar la aplicación de una transformación*. Ver Anexo C.

4. *Al activarse dicha opción de menú debemos aplicar la transformación que deseamos se realice*. Para ello en primer lugar creamos una instancia de la clase que implemente la transformación que deseamos realizar, que por supuesto, será una clase hija de la clase ATIM.filtro.cl_pA_filtroBase. A continuación realizamos una llamada al método m_v_aplicarFiltro de la clase cl_utilidades (alojada en el paquete ATIM.utilidades), pasándole como argumento dicha clase.

   Mediante esta llamada la aplicación se ocupa de aplicar la transformación. En ningún paso siguiente la aplicación sabrá que transformación está aplicando. Sólo lo sabe en el paso 4.

Como se ha podido ver añadir una nueva transformación es muy sencillo. Esto, creemos, facilitara la inserción de nuevas funcionalidades en la aplicación, lo cual simplificara en gran medida el trabajo futuro que otro equipo pueda realizar sobre la aplicación.

13.- GLOSARIO:

- **SDTI**: Sistema de tratamiento de imágenes, el proyecto actual.
- **ATIM**: Acrónimo de **A**plicación para el **T**ratamiento de **I**mágenes **M**édicas, precedente del proyecto actual.
- **Banda**: Cada una de las matrices de valores que conforman una imagen. La composición de los valores de las diferentes matrices para una misma fila y columna da como resultado un píxel.
- **Binarización:** Procedimiento según el cual se transforma una imagen a blanco y negro transformando cada color al extremo más próximo.
- **Binarización especial 1:** Binarización de una imagen con dos parámetros seleccionados por el usuario, los valores RGB del píxel seleccionado y un umbral. Para realizarse, se recorre la imagen original de arriba hacia abajo y de izquierda a derecha.
- **Binarización especial 2:** Binarización de una imagen con tres parámetros seleccionados por el usuario: valores RGB del píxel seleccionado, un umbral y el incremento o decremento de ese umbral. El recorrido de la imagen original se hace en zonas de píxeles concéntricas en torno al píxel seleccionado.
- **BMP:** Acrónimo de **B**it **M**a**p**, formato de almacenamiento de gráficos sin compresión de información.
- **Brillo:** Luminosidad del color negro en un monitor. En los equipos modernos se sustituye este término por el de *nivel de negro*.
- **Canal:** Banda.
- **Capa:** cada una de las imágenes atómicas de que se compone una imagen en la aplicación.
- **Contraste:** factor de ganancia que se aplica a las señales roja, verde y azul de una imagen.
- **Corrección gamma:** Exponente de corrección que se introduce para corregir la no linealidad entre voltaje de entrada y luminosidad de salida producida en los monitores.
- **Distancia entre dos píxeles:** Dados dos píxeles a y b y sus correspondientes valores RGB:

  $$\text{Distancia} = |R(a)-R(b)| + |G(a)-G(b)| + |B(a)-B(b)|$$

- **Evento:** mecanismo por el que un objeto notifica de manera indirecta su estado al resto de los objetos.
- **Extracción de color:** Procedimiento según el cual se transforma una imagen a blanco y negro aplicando a cada banda una transformación umbral (o su inversión) y computando la AND lógica de las bandas.
- **Extracción de bordes:** Transformación de imagen basada en el estudio del gradiente de luminosidad que detecta los bordes o fronteras entre zonas de colores.
- **Filtro:** Transformación de imagen que elimina información.

- **Filtro pasa baja:** Filtro que elimina los elementos de alta frecuencia (pasan los de baja frecuencia).
- **Filtro pasa alta:** Filtro que elimina los elementos de baja frecuencia (pasan los de alta frecuencia).
- **Frame:** ventana que está activa en un momento concreto de la ejecución del programa.
- **GIF:** Acrónimo de CompuServes **G**raphics **I**nterchange **F**ormat, formato de compresión de archivos gráficos sin pérdida de información limitado a gráficos con un máximo de 256 colores.
- **GUI:** Acrónimo de **G**raphic **U**ser **I**nterface, interfaz gráfica de usuario.
- **Herramienta:** Utilidad genérica de gestión, observación o manipulación de imágenes.
- **Inversión:** Transformación que proporciona el negativo cromático de una imagen dada.
- **Histograma:** Representación gráfica del análisis de frecuencia de los valores de muestreo estadístico. En tratamiento de imágenes, los valores de muestreo son los valores de intensidad de color.
- **HSI:** Acrónimo de **H**ue - **S**aturation - **I**ntensity, formato de representación del color mediante su tono, saturación e intensidad.
- **Intensidad:** Luminosidad de un color.
- **JAI:** Acrónimo de **J**ava **A**dvanced **I**maging, librería de componentes Java sobre tratamiento avanzado de imágenes.
- **JFC:** Acrónimo de **J**ava **F**oundation **C**lasses, extensión de Java que proporciona componentes para la construcción de interfaces gráficas de usuario (GUI).
- **JPEG:** Acrónimo de **J**oint **P**hotographic **E**xperts **G**roup, formato de compresión de archivos gráficos con pérdida de información.
- **Momentos invariantes:** Propiedades de las regiones de una imagen. Son especialmente importantes los momentos de Hu porque son valores constantes para la región aunque se haga una rotación, traslación o cambio de escala sobre la imagen.
- **Pixel:** Contracción de **pic**ture **el**ement, unidad básica de color programable de una imagen digitalizada.
- **Plug-In:** Clase Java autogestionada diseñada para ser directamente *acoplada* a cualquier aplicación gracias a su inmediata interfaz.
- **Portapapeles:** Almacén local para intercambio de información.
- **PNG:** Acrónimo de **P**ortable **N**etwork **G**raphics, formato de compresión de archivos gráficos sin pérdida de información.
- **PNM:** Acrónimo de **P**ortable a**nym**ap, formato de compresión de archivos gráficos sin pérdida de información.
- **Realzado:** aumento del contraste entre las partes mas claras y mas oscuras de una imagen.
- **Reflexión:** Transformación especular de la imagen.

- **Región de una imagen:**

- **RGB:** Acrónimo de **R**ed - **G**reen - **B**lue, formato de representación de color mediante la composición de luminosidades en rojo, verde y azul.
- **Saturación:** Índice de pureza de un color representado por la proporción no gris del mismo.
- **SOAP:** Acrónimo de **S**imple **O**bject **A**ccess **P**rotocol, protocolo de comunicación XML entre objetos distribuidos.
- **TAC:** Acrónimo de **T**omografía **A**xial **C**omputerizada, técnica de diagnóstico que permite visualizar imágenes seccionadas por planos de todo el cuerpo.
- **TIFF:** Acrónimo de **T**agged **I**mage **F**ile **F**ormat, formato de compresión de archivos gráficos sin pérdida de información.
- **Tono:** Representación de la tonalidad de color real.
- **Transformación:** En esta aplicación, manipulación de una imagen.
- **Transformación afín:** Manipulación algebraica de la imagen considerada como espacio euclídeo. En nuestra aplicación, son transformaciones afines la rotación, traslación, inclinación y escalado.
- **Transformación umbral:** Procedimiento según el cual se transforma una imagen a blanco y negro transformando cada color al blanco, si su luminosidad es mayor que un umbral dado, o al negro en caso contrario.

14.-BIBLIOGRAFÍA.

- " Visión por computador "
  Gonzalo Pajares.
  J.M. de la Cruz.
  Ed: Ra-ma (2001).

- " Java. Manual de referencia".
  Patrick Naughton
  Herbert Schildt
  Ed:Osborne/McGraw-Hill (1997).

- Documentación del proyecto Atim (precedente de este proyecto).

- Documentación on-line de JAVA:

  http://developer.java.sun.com/developer/infodocs

- Documentación on-line de JAI:

  http://java.sun.com/products/java-media/jai/forDevelopers/jai1_0_1guide-unc/

APÉNDICE: MANUAL DE USUARIO

SDTI es una Aplicación para el Tratamiento de Imágenes desarrollada por alumnos de la Facultad de Informática de la Universidad Complutense de Madrid. Esta aplicación pretende dar de un modo fácil e intuitivo a los usuarios de la misma una forma de manipular imágenes. La aplicación está pensada para facilitarle al usuario la manera de conseguir observar, retocar, moldear e incluso aplicar operaciones a nivel de píxeles de la imagen.

Con la facilidad de manejo de cualquier programa de Microsoft Windows es posible realizar multitud de acciones por parte del usuario. Presenta un interfaz gráfico que ofrece simplicidad de uso ya que el comportamiento de cada una de las opciones que se ofrecen es muy intuitivo. La vista general del programa justo en el momento en que arranca la aplicación es la siguiente. Indicamos en ella las diferentes partes de que consta el interfaz gráfico.

En las siguientes secciones se procede a detallar los diferentes menús a los que podemos acceder.

El menú Archivo

El menú de archivo es muy parecido a cualquier aplicación de tratamiento de imágenes. Al pinchar sobre el menú Archivo se observan las siguientes opciones.

Nuevo

Si se pincha en la opción nuevo se abre un archivo con una imagen vacía en la que se puede elegir el ancho y el largo de la misma. Incluso se puede elegir el color del fondo de la imagen.

Abrir

Si se pincha Abrir la aplicación presenta un cuadro de diálogo donde se puede elegir el archivo que se desee. Este cuadro de diálogo incorpora en el lado derecho un preview de la imagen. Se pueden abrir simultáneamente varias imágenes si son seleccionadas. Al pulsar Aceptar a imagen es abierta. En la figura mostramos el cuadro de diálogo.

Reabrir

Si se pulsa la opción reabrir se abre una imagen que recientemente el usuario a tenido abierta. Se muestra una lista con las últimas ocho imágenes.

Guardar

Al pulsar la opción Guardar pueden ocurrir dos cosas. Si el archivo había sido previamente guardado o existe en la memoria física (disco duro) un archivo con el contenido de la imagen en cuestión, se sobrescribe el archivo de imagen abierto. Si por el contrario se pulsa guardar y no existe ningún archivo desde el cual se haya recuperado la imagen a guardar se abre la opción de Guardar como.

Guardar como

Si se pulsa la opción Guardar como, se abre un cuadro de diálogo muy similar al abierto en la opción abrir. Se puede elegir un archivo ya existente o introducir un nombre para el archivo y especificar, con el filtro disponible en el cuadro la extensión del archivo que queremos crear**.**

Imprimir

Al pulsar al opción Imprimir se hace una llamada al cuadro de impresión del sistema y se puede realizar una impresión adecuada de la imagen seleccionada.

Salir

La opción salir cierra la aplicación. Se cierran antes todas las ventanas de las imágenes. La aplicación preguntará, en caso de encontrar alguna imagen que ha sido modificada, si se quieren salvar los cambios accediéndose en este caso a la opción de menú de Guardar o Guardar como según sea el caso.

El menú Edición.

El siguiente menú que encontramos en la barra de menús es el de Edición que consta de las siguientes opciones.

Deshacer

Si se pincha en la opción deshacer el programa elimina el último cambio realizado en la imagen dejando a ésta exactamente igual de cómo estaba antes de haber realizado dicho cambio.

Rehacer

Esta opción produce el efecto contrario a la anterior. Al pinchar en Rehacer lo que se consigue es provocar en la imagen una transformación que se había realizado anteriormente pero que se había eliminado pulsando Deshacer.

Cortar

La opción cortar realmente se realiza con uno de los plug-in que hemos definido. Se pincha en el plug-in de selección de máscara y se selecciona la parte de la imagen que queremos cortar

Copiar

Si se pulsa copiar la aplicación almacena en el portapapeles la imagen activa, que luego se podrá recuperar con la opción pegar.

Pegar

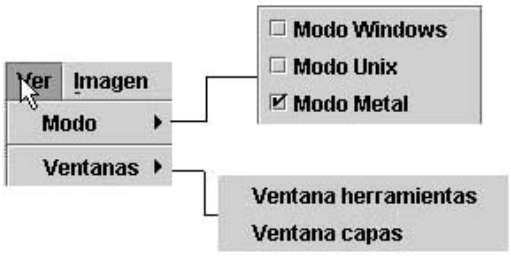
La opción pegar permite recuperar la imagen anteriormente copiada en el portapapeles y poderla pegar de dos formas diferentes en la aplicación. Una de las formas genera un nuevo archivo con la imagen y la otra nos pega la imagen del portapapeles como una nueva capa en la imagen que se tiene activa en la aplicación.

Seleccionar todo

Por el momento esta funcionalidad no tiene implementación.Se ha mantenido como posibilidad para el trabajo futuro con capas.

El menú Ver.

El tercero de los menús que presenta la aplicación es el menú Ver.



Modo

Si se pincha sobre Modo nos aparece un submenú que nos permite elegir el modo visual en el que se puede observar la aplicación entre uno de estos tres: Modo windows, Modo Unix o Modo Metal (este último seleccionado por defecto al comenzar la aplicación).

Ventanas

La opción de Ventanas abre también otro submenú. En el submenú se pude elegir ver la ventana de herramientas o bien la ventana de capas. Cada una de estas ventanas está descrita en la sección correspondiente. Más adelante se puede ver para que se utiliza cada una de ellas.

El menú Herramientas

SDTI ofrece la posibilidad de modificar imágenes utilizando ciertas herramientas. En el menú de herramientas se puede elegir cada una de estas herramientas. La aplicación también incorpora una barra de herramientas desde la cual aplicar las mismas. Para cada una de las herramientas y desde el menú ver se puede obtener una ventana para configurar la herramienta (si es que dispone de ella).Las herramientas que se pueden utilizar son:

Selección

 Selecciona una determinada región de la imagen. Después de pulsar sobre esta herramienta se pasa a la ventana activa y se pincha con el ratón. Se mantiene pulsado el ratón para delimitar una zona con forma rectangular en la imagen. Esta opción no es configurable, por tanto la ventana de herramienta no tiene ninguna opción.

Lupa

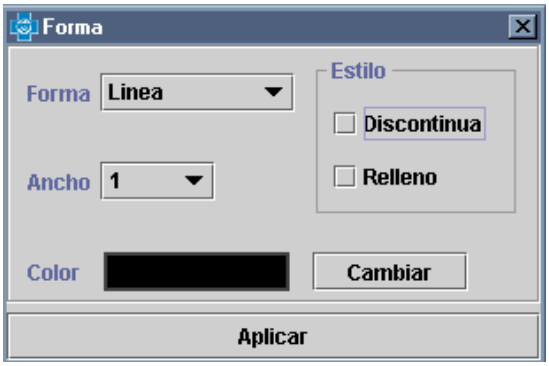
 Al pinchar en la opción lupa aparece en la ventana activa una lupa que aumenta la zona de la imagen por donde pasa. Se puede arrastrar esa lupa a través de toda la imagen. Si se muestra la ventana de herramientas el usuario puede configurar tanto el tamaño de la lupa, como su forma y el aumento que se desea. La ventana de configuración de la herramienta tiene el siguiente aspecto**:**

Máscara

 Con la opción de máscara se logra introducir en el portapapeles una selección realizada por el usuario de la imagen. El proceso es el mismo que el de la herramienta de selección pero con la opción máscara se introduce dicha selección en el portapapeles. Esta herramienta no es configurable por lo que no tiene ventana de configuración.

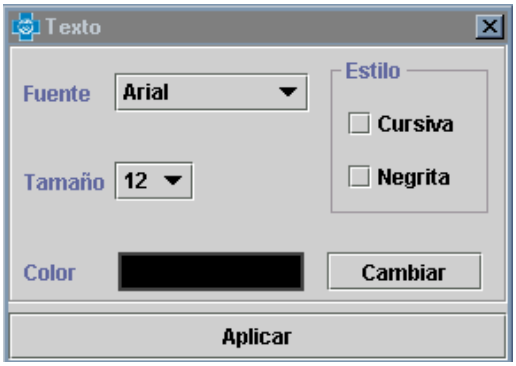
Forma

 Esta herramienta lo que permite es el dibujo sobre la imagen de figuras geométricas sencillas. En concreto las figuras que permite dibujar son: una línea recta, un rectángulo, un rectángulo redondeado y una elipse. Mediante a ventana de configuración que tiene el siguiente aspecto se puede elegir también el color, el ancho y el estilo (contorno discontinuo y rellenado):



Texto

 También en el menú de herramientas se tiene la opción de introducir texto en la imagen. La ventana de configuración es la siguiente:



Como se puede observar el usuario es capaz de configurar el tipo de fuente, el tamaño, el color e incluso se puede poner en negrita o cursiva.

Relleno

 La opción de relleno permite que una zona delimitada se pinte de un mismo color seleccionado. No tiene ventana de configuración.

Pipeta

 Permite pintar píxel a píxel en la imagen. No tiene ventana de configuración.

El menú Ventana

El menú ventana es el menú que permite la colocación de las ventanas de imagen dentro de la aplicación así como operaciones típicas de una ventana como son maximizar, minimizar o cerrar.

En cuanto a las posibles ordenaciones de las ventanas que se pueden elegir son: cascada, mosaico horizontal y mosaico vertical. Y en cuanto a operaciones típicas de una ventana se tiene: maximizar todas, minimizar todas, cerrar activa y cerrar todas.

El menú Ayuda

Al pulsar ayuda se ofrece al usuario una ventana con el título de la aplicación, el logotipo de la misma y los alumnos encargados del desarrollo del entorno SDTI.

Operaciones sobre imágenes en general

Para ejecutar cualquiera de estas operaciones hay que tener abierta en un marco al menos una imagen. Después, cuando se pulse la operación deseada, esta se llevará a cabo sobre la imagen contenida en dicho marco.

Si se quiere comparar la imagen original con la imagen resultado de la aplicación de la operación puede ser conveniente duplicar la imagen y ejecutar la operación en un segundo marco. También se pueden apreciar los cambios producidos por la operación en un mismo marco utilizando las opciones "hacer" y "deshacer".

Si se quiere conocer en profundidad el funcionamiento de algunas de estas operaciones como aspectos de su implementación, análisis de los resultados etc, se debe consultar el capítulo 10 del índice.

El menú Imagen.

El siguiente menú que podemos encontrar en la barra de menús de aplicación es el de Imagen. En el se agrupan todas las transformaciones geométricas que se pueden realizar. Por norma general cada una de las opciones que da el menú Imagen dispone de un cuadro de configuración de la misma que se explicará en cada caso.

Invertir

Si se pulsa en Invertir lo que se consigue es la inversión de cada uno de los píxeles de la imagen con su color opuesto (por ejemplo, los píxeles negros pasan a ser blancos y viceversa).

Reflejar

Al pulsarlo se puede elegir entre vertical u horizontal. El efecto que se consigue es una reflexión sobre una línea imaginaria que atraviesa la imagen horizontal o verticalmente según el usuario elija.

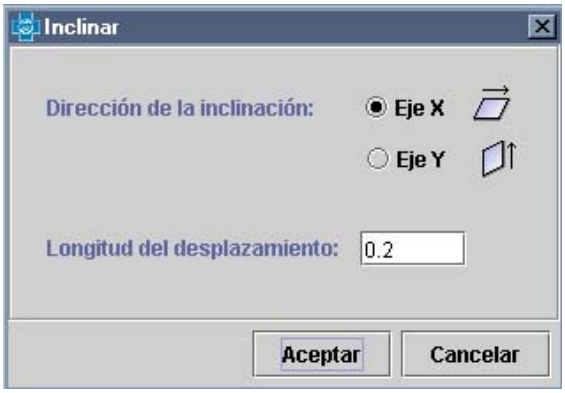
Rotar

La opción rotar permite girar una imagen un cierto ángulo determinado. Se podrá seleccionar un ángulo predefinido (45º, 90º, etc...) o bien elegir el ángulo concreto mediante la introducción del mismo en el siguiente formulario.

Se introduce el ángulo en el formulario, se elige la dirección hacia la que rotarlo y se pulsa Aceptar para que automáticamente la imagen quede rotada.

Inclinar...

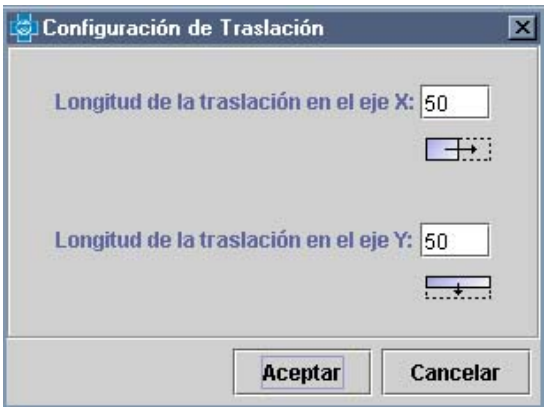
Para inclinar una imagen se pincha esta opción que nos muestra el siguiente formulario.



En este formulario se puede seleccionar si la inclinación será con respecto al eje X o con respecto al eje Y. Además podemos introducir la longitud de desplazamiento (por defecto 0.2).

Trasladar...

Otra de las operaciones geométricas que SDTI permite es la traslación. Consiste en un movimiento de la imagen en los dos ejes en la dirección derecha en el eje X y hacia abajo en el eje Y. Se toma como origen de coordenadas la esquina superior izquierda de la imagen. La elección de estos parámetros se realiza en el siguiente formulario.



Transformar a

La opción 'transformar a' permite cambiar la paleta de colores a una de las siguientes: IHS, RGB, Grises y Binaria. Consiste en una transformación píxel a píxel de la imagen.

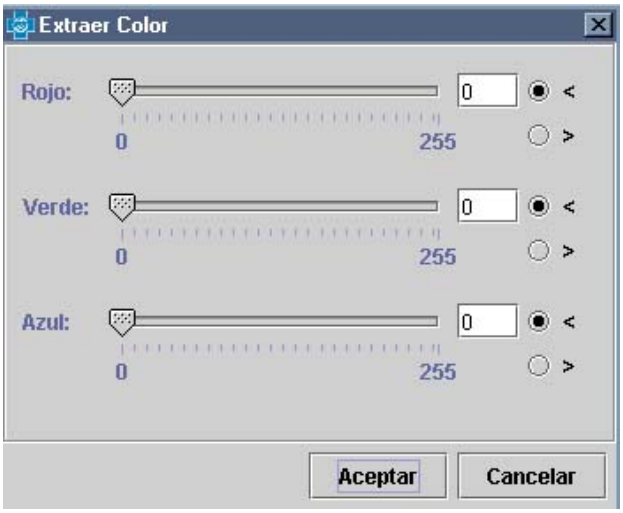
Umbral

Si se pincha en umbral nos aparecerá un formulario como el que sigue:

En la barra deslizable se elige el parámetro que se quiere aplicar a la transformación. El rango para este parámetro está entre 0 y 255. Una imagen tendrá por defecto 0

Extraer color

Mediante la opción extraer se consigue modificar los valores de las paletas RGB de la imagen. Con el siguiente formulario se configura la transformación:



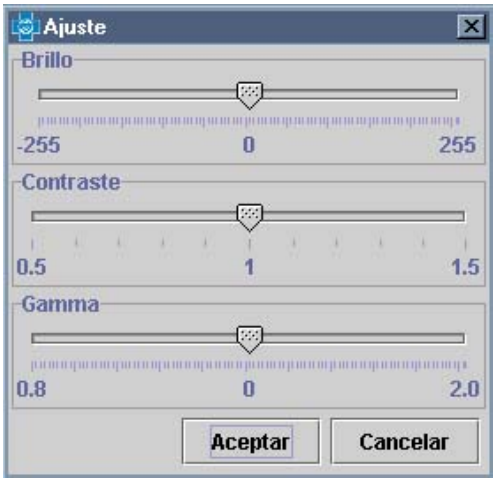
Para introducir un determinado valor se deslizan las barras y se elige si ese valor se quiere añadir o quitar seleccionando los símbolos '<' o '>' respectivamente.

Separar canales

Al pulsar esta opción lo que se consigue es extraer las tres bandas RGB de una imagen e introducir cada una de ellas en una ventana de imagen. La ventana de imagen tendrá el mismo nombre de la ventana padre añadiendo el color de la banda extraida.

Ajustar...

Con la opción Ajustar lo que se consigue es modificar los valores de brillo, contraste y factor γ de una imagen. Tendremos para configurar esta transformación el siguiente formulario:



Con las barras deslizantes elegiremos los valores para cada ajuste posible. Al pulsar Aceptar y la transformación se realiza en la ventana activa.

Duplicar...

Esta opción consigue realizar una copia exacta de la imagen que posee la ventana activa e introducirla en otra ventana de imagen.

Histograma...

El histograma de una imagen es una gráfica que contrasta el color de los píxeles con la cantidad de los mismos, es decir se consigue tener una visión de la cantidad de píxeles que tiene cada determinado color de la paleta RGB.

Propiedades...

Por último, la opción propiedades nos muestra una ventana con las medidas de la imagen como se observa a continuación:

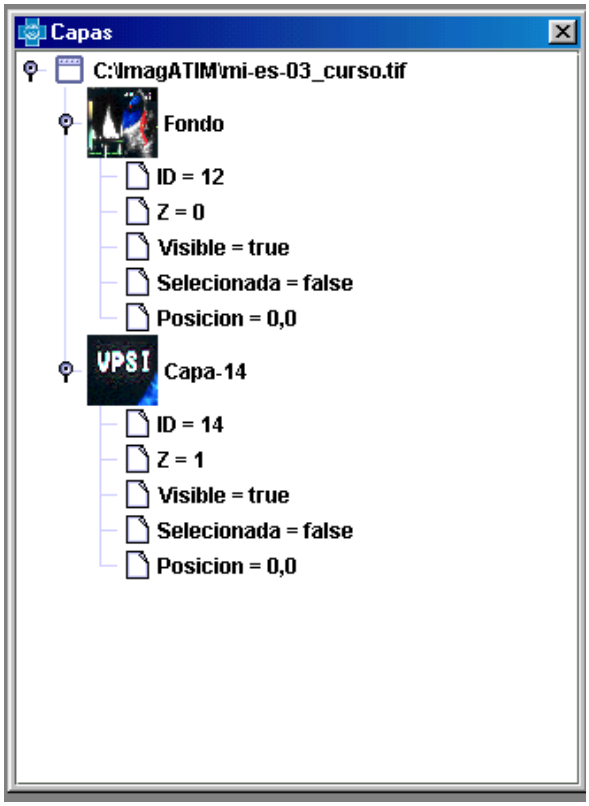
Desde esta ventana se pueden modificar ambos valores. También se tiene la opción de mantener las proporciones para que la imagen no se deforme marcando la casilla correspondiente. Además la ventana muestra la paleta de la imagen.

El menú Capas

SDTI ofrece la posibilidad de introducir capas en la imagen. Desde el menú *ver* se puede mostrar la ventana de capas. Esta ventana muestra siempre una imagen de fondo y las demás capas existentes. Los atributos que podemos observar (y modificar) de cada capa son:

- el índice Z, índice que muestra la profundidad en la que se encuentra esta capa en la pila de capas (la imagen de fondo tendrá siempre Z = 0).
- El identificador de capa ID.
- El atributo visible, que indica si la capa se puede ver o no.
- El atributo seleccionada, que indica si la capa está seleccionada.
- La posición de la capa dentro del lienzo utilizado.

Un ejemplo donde se pueden ver como está construida la ventana de capas es el que sigue:



Dentro del menú capas existen las siguientes opciones:

Crear nueva capa

Esta opción crea una nueva capa. Por el momento la aplicación tan solo crea nuevas capas que provengan de pegar, pero su funcionalidad está prevista como trabajo futuro.

Borra capas seleccionadas

Si se selecciona una determinada capa dentro de una imagen se puede borrar esa capa (o incluso varias si hemos seleccionado más de una) de la imagen.

Modificar capa

Se pincha en esta opción y se puede cambiar los atributos de cierta capa que previamente se ha seleccionado.

Combinar capas seleccionadas

Si se pincha esta opción se consigue generar una capa que sea el resultado visual de combinar todas las capas que se han seleccionado. La nueva imagen no modificará la capa de fondo y genera una nueva capa con el resultado de la combinación de las elegidas.

Combinar capas con el fondo

Esta opción es análoga a la anterior. Únicamente varía con ella en que se combinan todas y cada una de las capas de la imagen quedando dicha combinación como el fondo de la imagen.

El menú Bordes.

SDTI incorpora una opción para poder realizar un análisis píxel a píxel de la imagen y detectar los bordes de una imagen (contornos que definen y diferencian objetos dentro de una imagen). Esta opción se basa en determinados algoritmos que por normal general analizan un píxel teniendo en cuenta los píxeles que tiene alrededor y dependiendo de ese valor le asigna al píxel analizado un nuevo valor.

Los diferentes algoritmos que el usuario puede aprovechar para la detección de los bordes de la imagen son los siguientes:

Sobel,Prewitt,Laplaciano y Kirsch.

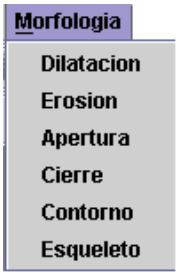
Todos ellos son algoritmos conocidos y de probada validez cuyos detalles de funcionamiento pueden consultarse en la bibliografía.

El menú Filtros

Otro de los menús que la aplicación incorpora es el menú de filtros. Los filtros son, en definitiva transformaciones a nivel de píxeles de la imagen. Utilizando un modificador en forma de matriz para los píxeles se logran efectos de suavizado, de perfilado y similares. Cada uno de los filtros definidos aplicará por tanto una transformación a la imagen. Los filtros disponibles son los siguientes:

Low Pass (Suavizado),High Pass (Perfilado),Mediana, Personalizado, Máximo, Mínimo y Gauss.

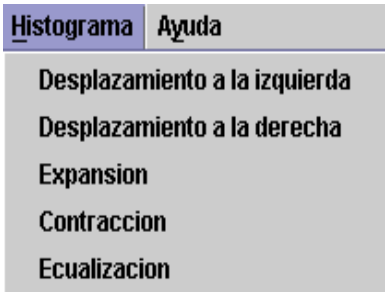
El menú Morfología



En el menú morfología de la aplicación se encuentran operaciones para extraer y tratar regiones de las imágenes seleccionadas. Todas ellas están definidas para imágenes binarias y de grises. Si se trata de realizar alguna de estas operaciones sobre una imagen de color RGB , la imagen será en primer lugar transformada a la escala de grises y posteriormente se llevara a cabo la operación. Este tipo de operaciones no están definidas para imágenes de color RGB.

Solamente hay que pulsar la opción del menú deseada teniendo un marco activo en el programa.

El menú Histograma:



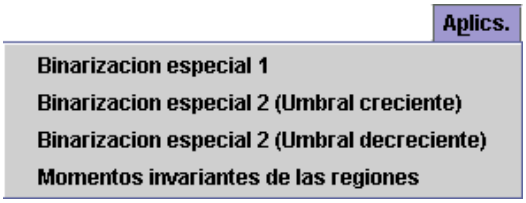
Nos proporciona operaciones que nos permiten realizar cambios en una imagen a través de transformaciones de histograma. En este caso y para cada una de las distintas operaciones que aparecen en el menú se presenta en primer lugar el histograma de la imagen original aun sin aplicar ningún cambio y posteriormente el histograma ya modificado así como la nueva imagen resultante de la aplicación de la operación.

Dentro de este menú aparecen:

- Desplazamiento a la izquierda: Como su nombre indica realiza un desplazamiento de todos los píxeles del histograma hacia la izquierda.
- Desplazamiento hacia la derecha: Análogo al anterior pero desplazando los píxeles a la derecha.
- Expansión: El resultado es que el histograma se expande hacia los niveles máximos permitidos
- Contracción: El histograma concentra los píxeles en las regiones centrales.
- Ecualización: Aplicando el algoritmo apropiado produce un realzado de la imagen.

Para todas estas operaciones de modificación del histograma los tipos de imágenes que contemplan son imágenes a color RGB e imágenes de grises ya que en el caso de las imágenes binarias no tienen sentido los cambios de histograma puesto que el histograma es una línea para el blanco y para el negro.

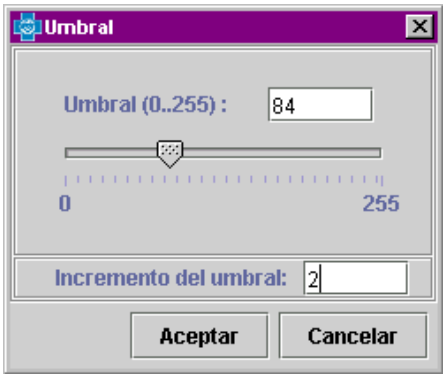
Menu de operaciones especiales: “Aplics.”



Este menú contiene operaciones de binarización sobre imágenes con varios parámetros cuyos valores selecciona el usuario así como el cálculo de momentos invariantes de regiones en imágenes.

Para hacer cualquier operación de binarización sobre una imagen es necesario seleccionar un píxel de una determinada imagen pulsando el botón izquierdo del ratón sobre el píxel deseado (cuyos valores RGB constituyen un parámetro de la operación). En la barra de estado aparecerán las coordenadas del píxel elegido en color rojo hasta que se vuelva a mover el ratón, momento en el que se empezarán a mostrar en azul las coordenadas actuales.

Si se trata de una operación especial de tipo 1, habrá que especificar un umbral para la binarización pero si es de tipo 2 se deberá indicar además el incremento o decremento del umbral. A continuación aparecerá la imagen transformada en el marco activo.



Si se quiere calcular momentos invariantes el primer requisito es que la imagen del marco activo sea binaria, en la cuál se diferenciarán las distintas regiones. La aplicación informará de esta situación si se trata de llevar a cabo esta operación sobre imágenes de grises o color RGB.

Al pulsar la opción del menú se entra en modo de selección de regiones. Sólo se podrán seleccionar regiones blancas para el cálculo de sus momentos. La selección de una región (que queda coloreada en gris) se hace pulsando sobre uno de sus píxeles con el botón izquierdo del ratón. A continuación se seleccionarán los valores “p” y “q” y se mostrará un formulario con los momentos invariantes calculados para la región. Se pueden seleccionar cuantas regiones se quiera. Para salir de este modo hay que pulsar el botón derecho del ratón sobre cualquier píxel de la imagen.

El menú Operaciones

El siguiente menú en la barra de menús es el de operaciones. En este menú se recogen todas las operaciones que SDTI permite realizar. Estas operaciones se realizan entre dos imágenes. Las operaciones pueden ser de dos tipos lógicas (And, Or, Xor) o aritméticas (Suma, Resta, Multiplica, Divide). Para aplicarle a una imagen una determinada operación con respecto a otra imagen se pincha con el puntero del ratón en la operación deseada. En este momento aparece un cuadro de diálogo. Se elige el archivo con el que vamos a utilizar la operación y se pulsa aceptar. La aplicación mostrará el resultado de la operación en la ventana activa donde estaba la imagen principal.