PRÁCTICA 10

Procesamiento de imágenes

Parte 2

Descripción de la práctica

El objetivo de esta práctica es realizar una aplicación que amplíe la realizada en las prácticas 8 y 9, introduciendo nuevas funcionalidades relativas al procesamiento imágenes. Concretamente, deberá incluir las siguientes nuevas funcionalidades:

- Modificación del contraste
- Rotación y escalado de imágenes

El aspecto visual de la aplicación será el mostrado en la Figura 1. El menú incorporará en su opción "Imagen", además de lo incluido en la practica 9, los ítems "AffineTransformOp", "LookupOp", "BandCombineOp" y "ColorConvertOp". En la parte inferior, además de los ya incluido en la práctica 9, se incorporará un área con tres botones asociados a tres tipos de contraste (normal, iluminado y oscurecido), otra área asociada a la rotación, con un deslizador para girar la imagen y tres botones para rotaciones fijas, y un área de escalado con dos botones (incrementar y reducir).

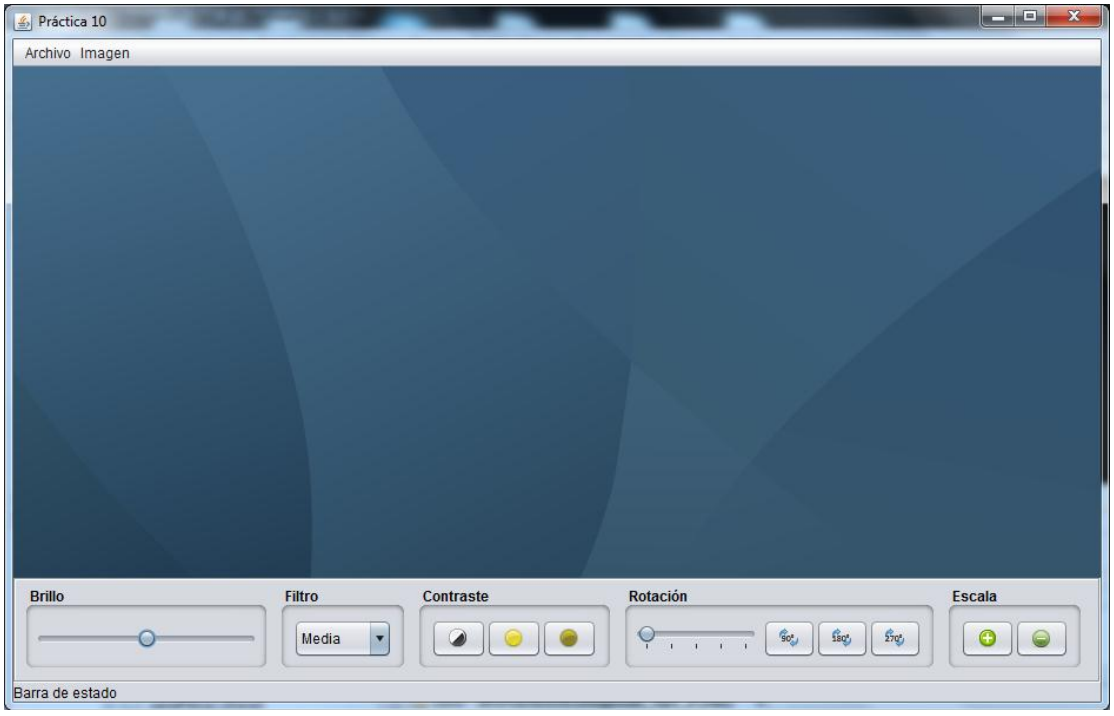

Figura 1: Aspecto de la aplicación

Pruebas iniciales

En una primera parte, probaremos los operadores "`AffineTransformOp`", "`LookupOp`", "`BandCombineOp`" y "`ColorConvertOp`" usando parámetros fijos (i.e., sin interacción del usuario para definir sus valores). Para ello, incluiremos las correspondientes opciones en el menú "Imagen" cuya selección implicará la aplicación de la correspondiente operación en la imagen seleccionada.

En estos casos, se probará el código mostrado en las transparencias de teoría (que incluiremos en el manejador del evento "acción" asociado al menú), teniendo en cuenta que dicha operación se aplicará sobre la imagen mostrada en la ventana activa (véase ejemplo de práctica 9).

En la clase `LookupTableProducer`, que se adjunta a esta práctica, se incluyen varios ejemplos de funciones para usar con el operador `LookupOp`. Por un lado, contiene métodos estáticos asociados a funciones clásicas (negativo, función-S, potencia, raíz, corrección gamma, etc.); por ejemplo, para crear la función S pasándole los correspondientes parámetros usaríamos el siguiente código:

```
LookupTable lt = LookupTableProducer.sFuction(128.0,3.0);
```

Por otro lado, define un método `createLookupTable` que devuelve objetos `LookupTable` correspondientes a las funciones clásicas anteriores pero usando parámetros predefinidos. Por ejemplo, el siguiente código crearía un objeto `LookupTable` por defecto asociado a la función-S:

```
LookupTable lt;
lt=LookupTableProducer.createLookupTable(LookupTableProducer.TYPE_SFUNCION);
```

Variación del contraste

En este apartado modificaremos el contraste de la imagen aplicando el operador `LookupOp`. Para ello incluiremos tres botones correspondientes a tres posibles situaciones:

- Contraste "normal", para imágenes en las que la luminosidad esté equilibrada. En este caso, se usan funciones tipo S
- Contraste con iluminación, para imágenes oscuras. En este caso, se usan funciones tipo logaritmo (si es muy oscura), funciones raíz (con cuyo parámetro podemos determinar el grado de iluminación) o correcciones gamma (con gamma mayor que 1)
- Contraste con oscurecimiento, para imágenes sobre-iluminadas. En este caso, se usan funciones potencia (con cuyo parámetro podemos determinar el grado de oscurecimiento) o correcciones gamma (con gamma entre 0 y 1).

Todas las funciones anteriores se encuentran implementadas en la clase `LookupTableProducer` que se adjunta a esta práctica. Para estos ejemplos, bastaría usar los parámetros por defecto que ofrece el método `createLookupTable`.[1]

Rotación

En este apartado rotaremos la imagen ofreciendo dos posibilidades:

- Giro libre, donde el usuario podrá girar 360° la imagen usando un deslizador (con rango [0,360]).
- Rotaciones predeterminadas de 90°, 180° y 270°.

Para ello haremos uso del operador "`AffineTransformOp`"; en el primer caso, se usará como grado el definido por el usuario en el deslizador, en el segundo serán valores fijos. En ambos casos, la rotación tendrá que hacerse poniendo como eje de rotación el centro de la imagen[2]:

[1] En caso de que se quisiera ofrecer al usuario la posibilidad de modificar los parámetros del contraste (p.e., mediante un deslizador similar al caso del brillo), habría que usar las funciones directamente (sin llamar al método `createLookupTable`) pasándole como parámetros los definidos por el usuario.

```
double r = Math.toRadians(180);
Point c = new Point(imgSource.getWidth()/2, imgSource.getHeight()/2);
AffineTransform at = AffineTransform.getRotateInstance(r,p.x,p.y);
AffineTransformOp atop;
atop = new AffineTransformOp(at,AffineTransformOp.TYPE_BILINEAR);
BufferedImage imgdest = atop.filter(imgSource, null);
```

Escalado

En este apartado escalaremos la imagen mediante el operador "*AffineTransformOp*". Para ello se usarán dos botones: un para aumentar el tamaño de la imagen y otro para reducirlo[3].

[2] Tras aplicar el *AffineTransformOp*, los operadores *RescaleOp* y *LookupOp* pueden no funcionar por incompatibilidad con el tipo de imagen devuelto por *AffineTransformOp* (que usa transparencia), si bien no genera excepción que permita detectarlo. En cualquier caso, si antes de aplicar los operadores *RescaleOp* y *LookupOp* la imagen se convierte a *TYPE_INT_RGB*, sí funciona correctamente (aunque inicializa a negro las zonas "transparentes" generadas tras una rotación). Este problema no se da con el resto de operadores, que sí funcionan con la salida generada tras la rotación (si bien puede generar excepciones).

[3] En este caso fijaremos el factor de escala (por ejemplo, 1,25 para aumentar y 0,75 para reducir).