# Digitalización de libros y mejora de libros ya digitalizados con herramientas de software libre

Joaquín Ataz López

**Agosto del 2014**

***



[*]La *Free Software Foundation* exige que su licencia no se traduzca a un idioma distinto del inglés, y que se incluya, al final del documento, una versión completa de la misma, también en inglés Yo entiendo que no haciéndolo así —y no lo voy a hacer así— la licencia de mi documento ya no es exactamente la FDL (*Free Documentation License* de la *Free Software Foundation*). Pero eso no significa que no sea una licencia libre. De acuerdo con los términos de la Ley de Propiedad Intelectual española, es el autor el que decide los términos de la licencia, y yo decido que éstos sean exactamente los determinados por la *Free Software Foundation*. Estamos, por lo tanto, ante una licencia libre *similar* a la FDL pero no idéntica a ella.

Algún día escribirá un artículo sobre licencias de software.

# Índice



# Preliminar: Contenido de la presente guía

La presente guía expone todos los pasos necesarios para, a partir de un libro (o documento) en papel, obtener un fichero digital que lo reproduzca con la mayor fidelidad posible usando para ello, exclusivamente, software libre disponible en los repositorios habituales de las principales distribuciones de GNU Linux.

En grandes líneas los pasos necesarios para obtener una versión digital de un libro en papel son los siguientes:

1. Escaneo del libro original, a partir del cual se obtienen una serie de ficheros con la imagen de cada una de las páginas. A veces para cada página se genera un fichero distinto, otras veces un sólo fichero puede contener la imagen de varias páginas.

2. Preparación del libro a partir de las imágenes de sus páginas.

3. Generación del libro digitalizado.

4. Aditamentos adicionales del libro ya generado para facilitar su manejo.

Hay una diferencia entre el primer paso y los demás, y es que sólo el primero requiere un hardware específico y puede considerarse básicamente mecánico. El resto de los pasos se basan exclusivamente en el uso de software. Por lo tanto podemos aplicar las técnicas explicadas en dichos pasos para mejorar un fichero ya digitalizado que algún amigo nos haya hecho llegar, o que hayamos conseguido por Internet. Por ello he incluido en el título de este documento una mención a la mejora de libros ya digitalizados.

En fin: de los cuatro pasos indicados, es el segundo el que ocupará una mayor extensión en su explicación. Respecto de ese segundo paso indicaré, además, dos técnicas distintas: una basada principalmente en comandos de consola y otra basada en una herramienta gráfica pensada precisamente para el objetivo de mejorar las páginas digitalizadas de un libro: «`scantailor`». El segundo procedimiento es más sencillo y, en la mayor parte de los casos, más rápido. Pero el primer procedimiento nos enseña mucho más sobre las herramientas propias de GNU Linux y sobre el diseño de los libros. Por eso he decidido recogerlos los dos en este documento.

# 1. El escaneo físico del libro

Aunque este posiblemente sea el paso más importante, de él es del que menos se puede decir con carácter general ya que el escaneo de un libro es una tarea necesariamente manual que depende, fundamentalmente, de la habilidad mecánica con la que sea realizada, la cual, por otra parte, se consigue fundamentalmente con práctica. Mucha práctica. Por lo tanto aquí sólo puedo dar algunos consejos muy generales.

Este primer paso debe hacerse con mucho cuidado. Piénsese que de la calidad que tengan los ficheros obtenidos con el escaneo dependerá siempre la calidad final del libro digitalizado. Y aunque en las demás fases del proceso podremos mejorar, con ayuda de cierto software lo que hayamos obtenido en esta fase, un buen escaneo inicial puede ahorrarnos mucho trabajo posterior.

En relación con el escaneo los siguientes consejos pueden ser útiles:

**Resolución a la que escanear.** La resolución de una imagen es la cantidad de píxeles por unidad de longitud. Como unidad de longitud habitualmente se usa la pulgada y por

lo tanto para referirnos a la resolución usamos el término ppp (puntos por pulgada) o, en inglés, dpi (*dots per inch*). A mayor resolución más detalles captará el escáner y, en teoría, más calidad tendrá la imagen, pero ello también implicará que el escaneo de una página tardará bastante más y el fichero resultante para la página será más grande, lo cual a su vez implicará que los programas que posteriormente manipulen dicho fichero funcionarán con una mayor lentitud.

Los escáneres actuales admiten una gran variedad de resoluciones. Personalmente entiendo que para páginas que consistan básicamente en texto la resolución razonable es de 300 o 400 ppp. En ningún caso es admisible una resolución inferior a 200 ppp y rarísimamente necesitaremos una resolución superior a 600 ppp.

> Con una resolución muy baja se pierden muchos detalles. Y aunque el cerebro humano es capaz de "rellenarlos" de tal modo que da la impresión de que se ve "bien", lo cierto es que si, por ejemplo, queremos después hacer un OCR sobre la imagen para extraer el texto, los errores de reconocimiento serán mucho mayores de los que se producirían de haber escaneado con una resolución superior. Sobre el OCR véase el apartado 4.1.
>
> Pero, por otra parte, debe tenerse en cuenta que en ocasiones un exceso de resolución puede provocar que se vea la trama de la impresión. Hay, por lo tanto, que buscar un equilibrio.

**¿Color? ¿Blanco y negro? ¿Escala de grises?** Otro factor que afecta al tiempo del escaneo de cada página y al tamaño de los ficheros resultantes es el del color. Básicamente existen tres opciones: Escaneo bitonal (blanco y negro), escala de grises y color. Salvo en casos especiales la mejor opción es la escala de grises. Incluso aunque estemos escaneando un texto normal, escrito en negro sobre páginas blancas, la escala de grises es mejor opción que la del simple blanco y negro, porque en muchas ocasiones, los escáneres configurados para blanco y negro, producen líneas demasiado estrechas que dificultan el posterior reconocimiento óptico de caracteres.

Si estamos escaneando un libro que incluye partes en color, deberemos preguntarnos si dichas porciones de color son realmente necesarias y, en caso afirmativo, escanear dichas páginas —pero sólo esas— en color. Personalmente entiendo que en muchos libros que usan el color para, por ejemplo, los títulos... este sólo sirve para encarecer la edición.

**Formato del fichero de salida:** La mayor parte de los escáneres permiten elegir entre varios formatos posibles para el fichero de salida. Aquí la regla es rotunda: hay que elegir un formato de salida que no implique compresión con pérdida de información. Las mejores opciones son TIFF o PNG. Nunca —si se puede evitar— hay que generar la salida en formato JPEG, ya que ese formato, aunque comprime bastante los ficheros, ello lo hace descartando parte de la información de la imagen original. Por eso se dice que la compresión JPEG es "con pérdida"; de donde resulta que no se trata de un formato adecuado para imágenes que posteriormente se pretende manipular mediante software, ya que en cada manipulación se perderá algo de calidad.

Hay escáneres que ofrecen como formato de salida el PDF. En el escáner que yo suelo manejar, sin embargo, los ficheros PDF de salida suelen tener peor calidad que los ficheros TIFF[1]; razón por la que nunca elijo PDF. Pero incluso aunque no fuera así, seguiría sin optar por PDF como formato de salida ya que la mayor parte de los programas que se utilizan para el postprocesado de las imágenes del escáner no admiten, como formato de entrada, el PDF. De modo que, si pretendemos postprocesar la imagen, el formato PDF para la salida del escáner simplemente provocaría la necesidad de añadir un paso adicional al post procesado para extraer de él las imágenes de las páginas en un formato compatible con el software de postprocesado.

En el caso de escoger, como formato de salida, un formato con capacidad multipágina tal como TIFF o PDF, téngase en cuenta que almacenar todo un libro en un solo fichero de salida puede hacer que este sea excesivamente grande, lo que dificultará más tarde su postprocesado. Piénsese que la manipulación de imágenes consume mucha memoria. Una buena precaución para evitar el bloqueo del ordenador, o su excesivo ralentizamiento durante el postprocesado de las imágenes, puede ser la de fraccionar el fichero de salida, de tal modo que, dependiendo de la memoria de nuestro ordenador, y del postprocesado que vayamos a aplicar, ningún fichero de salida tenga más de cierto número de páginas.

Según mis pruebas, extraer 100 páginas de un fichero TIFF, convertirlas al formato PNG y grabarlas en disco, se hace a una velocidad relativamente rápida en mi sistema, pero en cuanto el número de páginas a extraer aumenta... el ordenador empieza a responder peor, de modo que, si, por ejemplo, se tienen dos ficheros TIFF cada uno de ellos con 100 páginas, es muchísimo más rápido extraerlas en dos operaciones (una para cada fichero) que en una sola operación que afecte a ambos ficheros.

**Tamaño de página:** Algunos escáneres exigen del usuario que indique, antes del escaneado, un tamaño estándar de página. Como es lógico habrá que escoger el tamaño más pequeño en el que quepan todas las páginas del libro. No importa que este tamaño no se ajuste al formato A4. En el postprocesado podemos ajustar el tamaño de salida, pero ahora es importante respetar el tamaño real de la imagen. No es, por lo tanto, tampoco una buena idea aplicar, en el momento del escaneado, ningún factor de compresión del tamaño de las imágenes.

**Aspectos físicos del escaneado:** Aunque los anteriores factores son importantes, este es el factor decisivo para la calidad del escaneado. Al respecto téngase en cuenta lo siguiente:

- Antes de colocar el libro sobre la pantalla del escáner, deberemos asegurarnos de que esta esté absolutamente limpia. En ocasiones los libros —sobre todo cuando son viejos— dejan caer pequeñas partículas de papel. Conviene deshacerse de ellas con un paño seco o, si no lo tenemos disponible, soplando

[1] Muy posiblemente ello sea debido a que cuando se opta por PDF el escáner codifique la imagen como JPEG formato este que, como se acaba de explicar, no es adecuado para imágenes que posteriormente se pretendan manipular o editar.

ligeramente sobre la pantalla del escáner, con cuidado para no echar sobre ella nada de saliva.

- El libro debe colocarse bien alineado, de tal modo que los renglones del texto formen, en la medida de lo posible, líneas paralelas con uno de los bordes de la pantalla.
- Si el libro entero, abierto, cabe sobre la pantalla, podemos escanear a doble hoja, sin que sea importante el que la orientación de las páginas sea o no la correcta para leerlas. La orientación de la imagen es bastante sencilla de corregir por software.
- Tanto si escaneamos a doble página, como si lo hacemos a simple página, debemos asegurarnos de que el libro esté todo lo abierto que sea posible sin dañar al propio libro. Conviene incluso presionar con fuerza sobre el lomo del libro para que se aplane lo más posible durante el escaneado. Con ello conseguiremos, de un lado, evitar que entre demasiado luz por la curvatura que forma el lomo sobre la pantalla ya que dicha luz afectaría a la calidad final de la imagen. Pero además presionando sobre el lomo reduciremos el efecto óptico del escaneado en virtud del cual las líneas de texto se ven curvadas, tal y como se muestra en la figura 1.

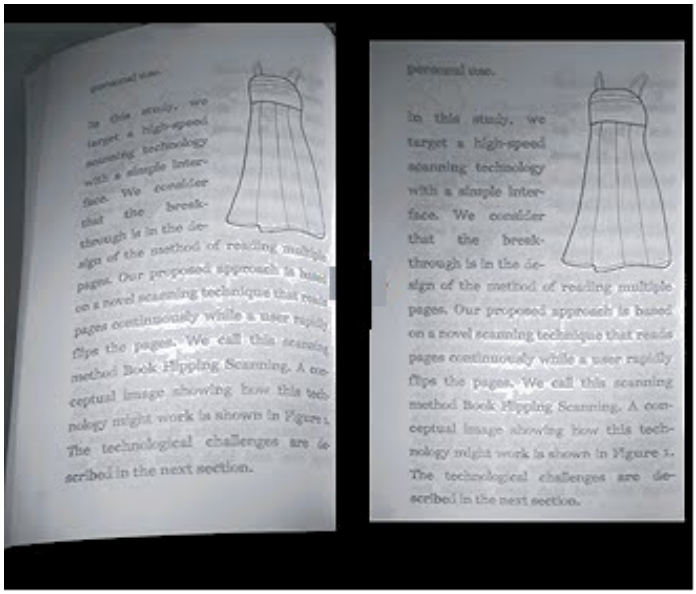


Figura 1: En la página izquierda puede verse la curvatura típica en las líneas de texto al escanear

Presionar con fuerza sobre el lomo del libro, y simultáneamente mantenerlo alineado con un borde, no siempre es sencillo. Una buena idea puede ser la de apoyarse ligeramente en el borde metálico (o de otro material) que suele delimitar la pantalla del escáner. Si así se hace, hay que tener mucho cuidado para evitar que los bordes de las páginas se arruguen como resultado de la presión.

Piénsese que al digitalizar un libro pretendemos preservarlo, y de ninguna manera es razonable, para ello, estropear el ejemplar físico a partir del cual se hace la digitalización.

Si el escáner está físicamente conectado a nuestro ordenador, necesitaremos algún programa para controlarlo. El más habitual, en sistemas GNU Linux, es XSane[2]. Pero existen muchas alternativas como «`simple-scan`»[3] o «`gscan2pdf`»[4], aplicación esta última que permite, además de escanear, realizar cierto post procesado simple de las imágenes obtenidas.

Finalmente hay que indicar que una alternativa al escaneado de las páginas con un escáner puede ser la de la fotografía digital de las mismas. Sin embargo en la práctica resulta muy difícil hacer fotografías de calidad. En los talleres profesionales de digitalización se dispone de una especie de bastidor en el que se introduce el libro, el cual es presionado con un cristal mate que mantiene las hojas abiertas y aplanadas. La fotografía se hace con un trípode que coloca a la cámara en una posición absolutamente perpendicular con respecto al libro, y la iluminación de la sala está pensada para evitar todo tipo de reflejos en el cristal y en las propias páginas blancas del libro... Si no disponemos de todo ese aparataje, posiblemente el escáner sea mejor opción que la fotografía digital.

## 2. Postprocesado de las páginas obtenidas en el paso anterior

Ya tenemos en nuestro ordenador uno o varios ficheros que contienen las páginas del libro que estamos digitalizando. Ahora hay que procesarlas para realizar una serie de tareas que *groso modo* consisten en:

1. Corregir, si fuera necesario, la orientación de las páginas.
2. Limpiar las páginas, eliminando las manchas que se producen a veces en el escaneado, y que se centran, sobre todo, en la zona próxima al lomo del libro, por donde a veces es inevitable que entre algo de luz, así como las partes de la pantalla no cubiertas por el libro, que se traducen en zonas absolutamente negras en el borde de la imagen obtenida con el escáner.
3. Si se ha escaneado a doble página, separar las dos páginas existentes en cada hoja escaneada.

[2] `www.xsane.or`
[3] `https://launchpad.net/simple-scan`
[4] `http://gscan2pdf.sourceforge.net/`

4. Rotar la imagen y rectificarla, si es necesario, para procurar que las líneas de texto estén plenamente horizontales y paralelas, eliminando la desviación que se haya podido producir por no haber alineado correctamente el libro, así como el efecto óptico de curvatura que se produce en la parte de las líneas de texto próxima al lomo del libro.

5. Diferenciar entre lo que es texto y lo que son márgenes, limpiando completamente estos últimos.

6. Corregir las pequeñas manchas que en el escaneo se hayan generado o, en general, los defectos en la imagen obtenida del escáner.

7. A veces, incluso, corregir fallos que ya estaban en el libro original. Por ejemplo: es posible, a veces, eliminar el subrayado de un texto.

Todo ello lo haremos con herramientas de software libre, disponibles en la mayor parte de las distribuciones de GNU Linux. Algunas son gráficas y otras son de línea de comandos. Básicamente se explicarán dos procedimientos alternativos: en el apartado 2.2 se expone un procedimiento totalmente "friki" basado en comandos de terminal, y en 2.3 se procurará hacer la misma tarea con una herramienta gráfica pensada para ello. No hay duda de que el segundo procedimiento es más fácil, y sus resultados no son peores... pero no nos precipitemos: si queremos realmente aprender el intríngulis del proceso, conviene conocer el primer procedimiento, pues sólo así comprenderemos la dificultad de lo que se está haciendo. Porque aunque está bien simplificar al usuario lo que es difícil, si se nos oculta la complejidad, realmente nunca terminaremos de aprender. Esa es, al menos, una de mis más firmes convicciones.

## 2.1. Convertir el fichero original en un formato que sea reconocido por el software que usaremos a continuación

Cualquiera que sea el camino que vayamos a seguir (el de línea de comandos o el gráfico), debemos asegurarnos de que la imagen obtenida con el escáner se encuentre en un formato que las herramientas de software que usaremos en el postprocesado puedan manejar. De no ser así el primer paso ineludible estriba en convertir nuestro fichero de partida a un formato distinto.

La herramienta universal en el mundo del software libre para convertir entre formas de imagen es «`convert`», incluido en el paquete «`imagemagick`», el cual se instala por defecto en prácticamente todas las distribuciones de GNU Linux que conozco.

### 2.1.1. Uso de «convert» (y «mogrify») para convertir entre distintos formatos de imagen

Convert es un comando de consola, por lo tanto hay que ejecutarlo desde una terminal. Posicionados en el directorio donde se encuentren nuestros ficheros originales, debemos ejecutar el siguiente comando:

```
convert fichorigen fichdestino
```

donde:

**fichorigen** Es el nombre del fichero origen. El fichero debe tener la extensión correspondiente a su formato interno.

**fichdestino** es el nombre del fichero a generar, indicando en él la extensión, que deberá ser la correspondiente al formato que deseamos obtener.

Así, por ejemplo, si hemos escaneado una página en un fichero llamado «`pag001.tif`» y queremos convertirlo a formato PNG, deberíamos escribir:

```
convert pag001.tif 001.png
```

También podemos usar «`convert`» para extraer todas las páginas de que pueden constar ciertos formatos gráficos como TIFF o PDF que tienen la capacidad de almacenar, en un sólo fichero, varias páginas distintas. Para ello debemos usar el siguiente formato de la orden:

```
convert fichero.tif %03d.png
```

Esta orden analizará el fichero TIF y, por cada página de que éste conste, generará una imagen de formato PNG que almacenará en un fichero cuyo nombre será un número de tres cifras. Mediante la instrucción «`%03d`» en el nombre del fichero de salida indicamos que queremos que el número a aplicar sea de tres cifras. Con «`%01d`» obtendríamos números de una sola cifra, y con «`%05d`» números de cinco cifras. También podemos, si lo deseamos, escribir un prefijo antes del número. Por ejemplo la instrucción «`pag%03d.png`» generará sucesivamente los ficheros “pag000.png”, “pag001.png”, etc.

> **NOTA:** Si el fichero original tiene demasiadas páginas, el sistema puede ralentizarse e incluso en ocasiones la ejecución del comando es suprimida por el propio sistema operativo. Cuántas páginas son “demasiadas” depende de la memoria instalada. Si el ordenador se nos bloquea hay que partir el fichero original en varios ficheros de tal modo que cada uno de ellos tenga un número inferior de páginas.

También podemos usar comodines para indicar conjuntamente varios ficheros de origen. Por ejemplo:

```
convert *.tif %03d.png
```

convertirá todos los ficheros TIF del directorio actual (sin importar si son o no multipágina).

> **OJO:** Al igual que en el caso anterior, una conversión masiva de ficheros puede ralentizar el sistema y dejarlo sin respuesta aparente durante bastante tiempo. Pero cuando se quieren convertir muchos ficheros con una sola orden, existe un procedimiento alternativo que evita este inconveniente: crear un bucle de *bash*. Para ello deberíamos escribir:
>
> ```
> f=-1
> for i in `ls *.tif`; do
> f=$((f+1))
> convert $i $f.png
> done
> ```
>
> Estas dos órdenes (pues las líneas 2 a 5 contienen una sola orden) convertirán a PNG todos los ficheros TIF del directorio, pero lo harán de uno en uno, por lo que no se bloqueará el sistema. La primera de las dos órdenes («`f=-1`») no es imprescindible; pero si en la misma terminal hemos ejecutado ya antes algún bucle usando la variable «`f`» como contador, esta línea se encarga de reiniciar su valor.
>
> Quienes entienden algo de *bash* comprenderán el anterior bucle y no necesitan más explicaciones. Los que no lo entiendan pueden simplemente ejecutarlo, y funcionará. Téngase en cuenta, no obstante, que mediante este bucle tan simple, no se garantiza que el número de los ficheros PNG tenga tres cifras. Es decir, si mediante el formato « `%03d`» el primer fichero generado se llamaría "000.png", mediante el bucle anterior el nombre sería "0.png". Esto también se puede arreglar mediante otros bucles. Por ejemplo:
>
> ```
> for i in `ls ?.png`; do
> mv $i 00$i
> done
> ```
>
> Este bucle añadirá dos ceros delante del nombre de todos los ficheros cuyo nombre conste de un sólo carácter.

Otra forma de evitar el colapso del ordenador como consecuencia de un procesado masivo de imágenes es usar, en lugar de «`convert`» la orden «`mogrify`» de la siguiente manera:

```
mogrify -format png *.tif
```

Esta orden, para cada fichero TIF de nuestro directorio, generará un fichero PNG con el mismo nombre (cambiando, claro está, la extensión).

«mogrify» también pertenece al paquete «imagemagick» y, en general, funciona igual que «convert» con la salvedad de que mientras «convert» deja intacto el fichero original, «mogrify» modifica el fichero original. Aunque esto tiene una excepción, y es que cuando lo que se pide a «mogrify» es que cambie el formato del fichero de imagen, «mogrify» genera un fichero nuevo, al igual que «convert», dejando intacto el fichero original.

Aparte de lo anterior, es importante tener en cuenta que, para el manejo masivo de ficheros, «mogrify» tiene una ventaja sobre «convert» pues nunca toma el control absoluto del sistema, por lo que no es probable que este se bloquee usando «mogrify».

### 2.1.2. Extraer imágenes de un fichero PDF (o PS) mediante «gs»

Si nuestro fichero original es un fichero PDF (o, poco probable, un fichero PS), también podemos usar «convert» para extraer las páginas del mismo y generar un fichero de imagen, en el formato deseado, para cada una de ellas. Pero en ciertos casos «convert» no consigue conservar toda la calidad que tenían las páginas originales del fichero PDF. Por suerte, tratándose de PDFs, hay otra posibilidad bastante versátil: «gs», el programa intérprete del lenguaje PostScript usado por los formatos PS y PDF.

Mediante la siguiente orden generaríamos un fichero PNG para cada página del fichero «origen.pdf»:

```
gs -q -dBATCH -dNOPAUSE -sDEVICE=png16m -r300x300 -sOutputFile=%03d.png origen.pdf
```

donde:

**-q, -dBATCH y -dNOPAUSE** son tres parámetros generales de «gs» cuyo efecto básico es el de evitar que se escriban mensajes en la salida estándar y que «gs» se detenga tras procesar cada página (lo cual es su funcionamiento por defecto).

**-sDEVICE** indica el formato de salida. En nuestro ejemplo PNG con 16 millones de colores. Hay muchísimos otros formatos de salida posibles. Podemos obtener un listado de los mismos, y del nombre con el que los reconoce «gs», simplemente escribiendo el comando «gs» sin ningún otro parámetro, para entrar en su modo interactivo, y, a continuación, escribiendo «devicenames ==». Entre los formatos de salida interesantes, además de «pnggray» para generar ficheros PNG en escala de grises, disponemos de varios tipos de TIFF (mi preferido es «tiffg4»), así como de los formatos «pnm», «pgm», «ppm» o «pbm», que son los formatos que admite el programa «unpaper» del que se hablará en la sección 2.2.1[5].

**-r** permite indicar la resolución de los ficheros de salida. A mayor resolución, mayor tamaño en los ficheros de salida pero también mayor calidad de la imagen. Con el

[5]Para más información sobre los formatos de salida de «gs» véase http://ghostscript.com/doc/current/Devices.htm

límite de la resolución que tuviera la imagen original. Si indicamos como salida una resolución mayor que la del fichero original, no habrá ganancia de calidad, pero sí aumentará el tamaño del fichero.

**-sOutputFile** Recoge el nombre del o de los ficheros de salida. Al igual que «convert» (y que muchas aplicaciones para GNU Linux) admite la indicación «%0xd» para indicar números de «x» cifras.

Aparte de con «gs», otro procedimiento para extraer las imágenes que conforman las páginas de un fichero PDF es el de la aplicación «pdfimages» que en las distribuciones basadas en Debian forma parte del paquete «poopler-utils», que habitualmente se instala por defecto. Hay ciertos PDFs, no obstante, en donde «pdfimages» no devuelve una imagen real de las páginas de que consta el fichero, sino una especie de negativo invertido de las mismas. La verdad es que no he indagado demasiado la razón de ello.

«pdfimages» extrae las imágenes en formato PNM.

## 2.2. Mejora de los originales exclusivamente (casi) con herramientas de línea de comando

Las posibilidades de «convert», «mogrify» y otras herramientas de líneas de comando son muchísimas. Con ellas podemos mejorar sensiblemente las imágenes procedentes del escáner de una forma casi totalmente automática. Sólo necesitaremos mirar las imágenes para asegurarnos de que todo funciona correctamente.

Como, por esta vía, usaremos «unpaper» que sólo admite ciertos formatos gráficos, el primer paso es convertir nuestros ficheros originales al formato PBM en el que, a partir de ahora, trabajaremos. Para ello, suponiendo que la salida del escáner fueran ficheros TIF, escribiremos:

```
convert *.tif %03d.pbm
```

teniendo en cuenta todo lo que se indicó en el apartado 2.1. Si nuestro fichero original es un PDF, podemos extraer los ficheros PBM mediante «convert», o mediante «gs», tal y como ya se ha explicado.

Si la orientación de las imágenes no es la natural de lectura, podemos cambiarles ya la orientación. Para girar todos los ficheros 90 grados a la derecha usaremos:

```
mogrify -rotate 90 *.pbm
```

**NOTA:** Obsérvese que aquí usamos «mogrify» en lugar de «convert». Eso es porque no tenemos especial interés en conservar los ficheros PBM originales que tienen una orientación inadecuada. En general, decidirnos entre «convert» y «mogrify» exige que nos preguntemos si volveremos a necesitar los ficheros originales. Aparte del caso en que vayamos a necesitar en el futuro los ficheros originales, también conviene usar «convert» cuando no estemos totalmente seguros del efecto que producirá cierta orden.

También podemos, si lo deseamos, ahorrarnos ahora el cambio de orientación y encomendárselo al propio «unpaper», tal y como en seguida veremos.

### 2.2.1. Preparación de las páginas usando «unpaper»

#### A) Funcionamiento general de «unpaper»

Unpaper es una herramienta, disponible en los repositorios de Debian, para el postprocesamiento de páginas de papel escaneadas a partir de libros. Básicamente "limpia" las hojas escaneadas para hacerlas más legibles en la pantalla, de cara a su posible conversión a PDF o a un proceso de OCR. Para hacer esa limpieza de las imágenes, «unpaper»:

- Detecta qué parte de la imagen se corresponde con el contenido real de la página, eliminando los bordes oscuros que, en torno a dicho contenido real, es corriente que aparezcan en las imágenes escaneadas.
- Rota la página para alinear correctamente los renglones del texto, de tal forma que se vean paralelos.
- Corrige las curvaturas que se suelen producir en la parte de las líneas de texto más próximas al lomo del libro.
- Limpia la imagen, eliminando los píxeles aleatorios que son corrientes en las imágenes escaneadas.
- Detecta el contenido real de texto de la página, y lo centra en la imagen. En hojas que recogen dos páginas, divide la hoja en dos partes y en cada una de ellas centra la página correspondiente.

El formato general de una llamada a «unpaper» es el siguiente

```
unpaper [opciones] entrada salida
```

donde

**opciones** Alguna de las opciones de «unpaper» que permiten ajustar su funcionamiento. Respecto de estas opciones hay que decir que, en general, si las páginas a procesar con «unpaper» fueron correctamente escaneadas, y el fondo de las mismas no es demasiado oscuro (cosa que ocurre, por ejemplo, en libros muy viejos), «unpaper» funciona bastante bien con sus opciones por defecto. Si los resultados del primer intento no son satisfactorios, podemos hacer un segundo intento ajustando alguno de los parámetros.

En la próxima sección se explican algunas de las principales opciones de «unpaper». De momento sólo diré que yo, salvo casos especiales, utilizo exclusivamente las opciones «-v» y «--layout».

**entrada** El fichero de entrada. Sólo se admiten ficheros en los siguientes formatos: PBM, PGM, PPM ó PNM. Para hojas de libro escaneadas sin color el mejor formato es PBM.

> PBM es un formato bitonal: sólo admite blanco y negro. Antes se dijo que era preferible escanear en escala de grises a hacerlo en blanco y negro. No hay, sin embargo, contradicción entre aquella afirmación y la actual, que aconseja usar PBM en lugar de PNM (que es un formato en escala de grises). El blanco y negro da malos resultados en el momento de escanear, pero una vez que hemos escaneado, conviene pasarse a un formato bitonal porque éstos ocupan bastante menos espacio en el disco y dan lugar a ficheros cuyo manejo tiene muchos menos requerimientos de memoria.

Se pueden indicar varios ficheros de entrada, pero no con los comodines habituales de *bash*, sino exclusivamente mediante la convención «`%0xd`». Y así «`pag%02d.pbm`», por ejemplo, procesará todos los ficheros cuyo nombre sea «`pagxx.pbm`», siendo xx un número de dos cifras. Además, por defecto, cuando recibe este parámetro, se asume que la primera página será la nº 1, y no la número 0. Para indicar un número distinto por el que empezar hay usar la opción «`-start npag`», donde npag es el número por el que empezar.

Cuando se usa el formato «`%0xd`» «`unpaper`» empieza por la primera página y sigue procesando páginas consecutivas, salvo que se haya indicado la opción «`-end npag`» para indicar el número de la última página a procesar.

**salida** Indica el nombre que asumirá el fichero de salida. En modo multipágina, hay que usar también la convención «`%0xd`». Es imprescindible indicar la extensión que, normalmente, será la misma que la del fichero de entrada; en nuestro ejemplo: PBM.

Por ejemplo: el siguiente comando hará que «`unpaper`» procese todos los ficheros PBM del directorio cuyo nombre consista en «`pag`» deguido de tres cifras, generando, para cada uno de ellos, un fichero llamado "unpap" seguido también de tres cifras. El primer fichero a procesar será «`pag001.pbm`» y el fichero generado tras su procesamiento se denominará «`unpap001.pbm`»:

```
unpaper pag%03d.pbm unpap%03d.pbm
```

Y el próximo comando hará lo mismo que el anterior, pero empezando en «`pag105.pbm`» y terminando en «`pag345.pbm`»:

```
unpaper -start 105 -end 345 pag%03d.pbm unpap%03d.pbm
```

**B) Opciones de ajuste de «unpaper»**

Ya se han mencionado las opciones «-start» y «-end» de «unpaper». Yo suelo usar, también las opciones «-v», que provoca que «unpaper» emita por pantalla información sobre lo que se le hace a cada imagen, y «--layout» (o «-l») que indica a «unpaper» cuál es el diseño de las páginas que se van a procesar. El diseño puede ser «double» si se ha escaneado a doble página, «single» si se ha escaneado a razón de una página por hoja, o «none» si deseamos desactivar la detección automática de diseño de página que realiza «unpaper».

Pero «unpaper» dispone de muchas más opciones. No las voy a explicar todas en este documento, por no hacerlo demasiado extenso, y porque, además, no estoy muy seguro de cómo funcionan varias de ellas: la información que al respecto ofrece la ayuda del programa no es muy completa y presupone un conocimiento del diseño interno de los ficheros de imagen del que yo carezco. Explicaré, por lo tanto, sólo, las opciones que me parecen más importantes en la mayor parte de los casos, agrupadas en dos categorías: opciones que permiten realizar cierta manipulación de la imagen antes o después de haberla procesado, y opciones que afectan al modo en que «unpaper» procesará la imagen:

**a) Manipulación de la imagen antes y después de procesarla:** Podemos pedirle a «unpaper» que, antes de analizar la imagen y procesarla, o después de haberlo hecho, realice en ella alguna transformación como, por ejemplo, corregir la orientación de la imagen, guillotinarla, producir un reflejo horizontal o vertical de la misma, etc.

Por ejemplo: si la orientación de nuestras imágenes no es la correcta, podemos pedirle a «unpaper» que, antes de procesarla, rote la imagen 90 grados a la derecha o a la izquierda y que después del procesado la restaure a su posición original, si así lo deseamos.

Las transformaciones posibles son:

**Rotación de la imagen 90 grados.** Para rotar la imagen antes del procesado se usa «--pre-rotate» y para rotarla después del procesado, «--post-rotate», ambas opciones deben ir seguidas de un valor que puede ser «90», para rotar la imagen a la derecha, o «-90» para rotarla a la izquierda.

**Reflejo,** que puede ser horizontal («h»), vertical («v»), o también horizontal y vertical simultáneamente («h,v»). El reflejo previo al procesado se obtiene con las opciones «-M» o «--pre-mirror», seguidas de «h», «v» o «h,v» según el reflejo deseado. El reflejo posterior al procesado se obtiene con «--post-mirror», seguido también del valor representativo de la dirección del reflejo.

**Guillotinar la imagen,** recortando una porción de su ancho y de su largo. Las opciones son «--pre-shift» o «--post-shift» seguidas de dos números separados por una coma, que representan el número de píxeles horizontales y verticales que se deben recortar de la imagen.

**Limpiar los bordes de la imagen** dejándolos totalmente blancos. Ello se consigue con «`--pre-border`» y «`--post-border`», seguido de cuatro coordenadas que indican la parte de la imagen que no se limpiará.

**Limpiar un área de la imagen** dejándola totalmente blanca. Ello se consigue con «`--pre-wipe`» o «`--post-wipe`», seguidos de cuatro valores que indican las coordenadas izquierda, superior, derecha e inferior del rectángulo a limpiar. La diferencia con la opción anterior está en que en aquella las coordenadas definen la zona que no se limpiará, y en esta las coordenadas definen la zona que se limpiará. Como «`--pre-border`» limpia toda la imagen, salvo una zona, no se puede usar más de una vez; por el contrario podemos usar varias veces «`--pre-wipe`» (o «`--post-wipe`») para limpiar varias zonas de la imagen.

**Cambiar el tamaño de la imagen.** Para hacerlo antes del procesado se usa la opción «`-s`» (o «`--size`») seguido, bien de los valores de ancho y alto, bien de un nombre de tamaño estándar como a4, a3, letter, etc. «`--post-size`» realiza este mismo cambio, después de haber procesado la imagen. Obsérvese que este cambio afecta exclusivamente al tamaño del lienzo sobre el que se encuentra la imagen, pero no afecta al contenido de la imagen propiamente dicho. Esto último se obtiene escalando la imagen.

**Escalar la imagen.** «`--strecth`» y «`--post-stretch`» seguidos de un valor para ancho y alto, o del nombre de una unidad de medida de páginas estándar provocan un escalamiento de la imagen hasta dicha medida. Esta opción, a diferencia de la anterior, no afecta sólo al tamaño de la página, sino que también altera su contenido para agrandarlo o reducirlo proporcionalmente. También se puede escalar la imagen indicando, no la medida final, sino el factor de zoom que hay que aplicar. Para ello se usan las opciones «`-z`» (o «`--zoom`») y «`--post-zoom`» seguidas del valor del factor de zoom a aplicar.

**b) Opciones que afectan al procesado propiamente dicho.** Unpaper busca ciertos defectos en la imagen e intenta corregirlos. A continuación indicaré los defectos que se buscan y las principales opciones que controlan el cómo «`unpaper`» los corregirá. Téngase en cuenta que todas estas opciones admiten dos nombres, uno corto y otro largo. El nombre corto va precedido de un solo guión y el largo de dos. Con carácter general indicaré, para las opciones que explico, el nombre largo —que es más significativo de lo que la opción hace— y para las demás el nombre corto.

**Zonas oscuras.** Para detectar las zonas oscuras y las manchas de la imagen, «`unpaper`» utiliza un filtro de negro que podemos configurar mediante varias opciones («`-bn`», «`-bs`», «`-bd`», «`-bp`», «`-bt`», «`-bx`»), de las que la que en principio hay que utilizar más veces es «`--blackfilter-intensity`» (o «`-bi`») seguido de un número que refleja la intensidad en la búsqueda y detección de zonas oscuras en la imagen.

Por defecto este parámetro vale 20, aumentándolo se aclarará más la imagen, y reduciéndolo se oscurecerá.

**Ruido.** En tratamiento de imágenes se llama "ruido" de la imagen a cierta variación aleatoria de su brillo o de su color que afectan a la nitidez de la misma, produciendo, a veces, una sensación de granulado. La opción «`--noise-intensity`» (también llamada «`-ni`») controla la intensidad en que «`unpaper`» intenta reducir dicho ruido. Por defecto su valor es de 4.

**Desenfoque.** La detección de zonas desenfocadas se controla también con varias opciones («`-ls`», «`-lp`») entre las que hay que destacar «`--blurfilter-intensity`» (o «`-li`») que afecta a la intensidad en la detección de zonas desenfocadas o borrosas. Por defecto esta opción vale 0.01.

**Curvatura en las líneas de texto.** Este peculiar efecto de las páginas escaneadas, al que en inglés se denomina "deskew", es también altamente configurable. Podemos indicar la dirección de búsqueda para este defecto («`-dn`»), el tamaño de la línea virtual para su detección («`-ds`»), la cantidad de píxeles que se deben acumular para considerar que el defecto se produce («`-dd`»), el grado de rotación («`-dr`»), la desviación máxima («`-dv`»), etc.

**C) Aspectos en los que el funcionamiento por defecto de «`unpaper`» produce resultados inadecuados** En general, como se ha dicho antes «`unpaper`», con sus opciones por defecto, tiende a funcionar bastante bien siempre que, claro está, el escaneo de la imagen haya sido correcto. Los resultados son fantásticos, aunque conviene revisarlos página a página antes de dar el siguiente paso.

Hay dos aspectos, no obstante, en los que, incluso con un buen escaneo, a veces el funcionamiento por defecto de «`unpaper`» plantea problemas:

1. Las páginas que no están totalmente llenas de texto. Por ejemplo: las páginas de terminación de capítulo, que no están totalmente llenas de texto. En un libro bien diseñado el texto de estas páginas estará alineado con el borde superior de la misma. Unpaper, sin embargo, por defecto, centra también estas páginas. Esto lo podemos arreglar individualmente con el «`gimp`» o confiar en que el paso explicado en la sección 2.2.3 nos permita detectar y solucionar de forma casi automática dichas páginas.

2. Un efecto bastante habitual en «`unpaper`» es el de que en las páginas que contienen índices, muchas veces una parte del índice se descarta como si fueran manchas de la imagen. En particular, en los índices de contenido, es bastante corriente que desaparezcan los números de página correspondientes a los epígrafes; y en los índices de abreviaturas, muchas veces desaparece la columna de las abreviaturas.

Este último defecto se tiene necesariamente que arreglar a mano. Para ello podemos abrir con el «gimp» simultáneamente la página original y la que es fruto del procesado de «unpaper», copiar de la primera la parte de información que «unpaper» descartó y pegarla en la segunda.

Por último debe tenerse en cuenta que en libros viejos, cuando el fondo de la página amarillea, o en libros muy voluminosos en los que no ha sido posible aplastar totalmente el lomo contra la pantalla, el funcionamiento de «unpaper» es más problemático. A veces se tarda tanto en ajustar las opciones del programa, que es casi más rápido limpiar las páginas individualmente con el «gimp»; sobre todo si el libro no es muy voluminoso.

### 2.2.2. Guillotinado de las páginas

Si hemos escaneado a doble página, tras ejecutar «unpaper» tendremos, por cada fichero (hoja escaneada) dos páginas. Ha llegado el momento de separarlas. Podemos confiar en que «unpaper» habrá centrado correctamente cada página en la parte de la hoja que le corresponda, por lo que para separar las páginas basta con guillotinar las hojas originales justo por el centro. La herramienta para hacer esto es, de nuevo «convert». Pero antes debemos medir las páginas, para lo cual usaremos «identify», que es otra de las aplicaciones incluidas en el paquete «imagemagick». Si en nuestro sistema está instalado convert, también lo estará «identify».

Basta con ejecutar «identify» sobre cualquiera de las imágenes generadas por «unpaper» ya que podemos asumir que todas las hojas tendrán el mismo tamaño puesto que se han escaneado en una misma operación y con una misma configuración del escáner. Por lo tanto, si, por ejemplo, la primera hoja escaneada (con las dos primeras páginas del libro) la tenemos almacenada en el fichero «unpap001.pbm» simplemente deberemos ejecutar la instrucción:

```
identify unpap001.pbm
```

y obtendremos la siguiente información:

```
unpap001.pbm PBM 2184x1436 2184x1436+0+0 1-bit Bilevel
DirectClass 393KB 0.000u 0:00.010
```

En primer lugar se nos informa del nombre del fichero, en segundo lugar del formato de imagen que contiene y en tercer lugar de las dimensiones de la imagen. Esta última es la información que nos interesa, en nuestro ejemplo 2184x1436, es decir: la imagen mide 2184 píxeles de ancho y 1436 de alto, lo que significa que, si queremos guillotinar justo por el centro, deberíamos obtener, de cada hoja dos imágenes con la misma altura que la original, pero con la mitad de la anchura, o, lo que es lo mismo, la dimensión de cada imagen sería de 1082x1436.

Con la opción «-crop» de «convert» podemos partir las imágenes originales. La orden que necesitamos para ello es:

```
convert -crop 1082x1436! +repage unpap*.pbm part%03d.pbm
```

Esta orden partirá todos los ficheros PBM cuyo nombre empiece por «unpap» (obtenidos en el paso anterior, de «unpaper») y generará los nuevos ficheros, que se denominarán «partxxx.pbm», donde xxx representa un número de tres cifras. Obsérvese que he indicado tres cifras, asumiendo que el libro que estábamos convirtiendo no tiene más de 999 páginas. Si el número de páginas llega a las 1000 habría que cambiar «03d» por «04d».

**NOTA:** Cuando se aplica la función «-crop» a «convert» y se están procesando masivamente varias imágenes, si alguna de las dimensiones indicadas a «-crop» coincide con la dimensión original de la imagen, como en nuestro caso, en donde la altura de la imagen original se mantiene en la nueva imagen a generar, «convert» se comporta como si la imagen original fuera una salchica que hubiera que partir a rodajas. En nuestro ejemplo se hacen sólo dos rodajas (dos imágenes de salida por cada imagen de entrada) debido a que la anchura de las rodajas equivale, exactamente, a la mitad de la imagen original. Pero si se hubiera indicado en «-crop» una anchura inferior a la mitad, saldrían más de dos rodajas. Por ejemplo: si la imagen original tuviera unas dimensiones de 1500x1000 e indicáramos «-crop 500x1000!», por cada imagen original se generarían no dos imágenes de salida sino tres, cada una de ellas de 500 píxeles de ancho.

Esta forma de funcionar «-crop» implica que si la imagen original tiene una anchura que, medida en píxeles, es impar, por ejemplo 3001 e indicamos como valor de «-crop» 1500, por cada imagen original se generarán tres imágenes: dos de ellas tendrán una anchura de 1500 píxeles y la tercera una anchura de un sólo píxel, que es lo que queda tras la última rodaja. Esta tercera imagen de un píxel de anchura, debería luego ser descartada, lo que es un follón, pues o bien vamos borrando a mano una de cada tres imágenes de salida, o las ordenamos por tamaño y eliminamos las más pequeñas que se corresponderán con las imágenes de sólo un píxel de anchura.

Si, por el contrario, siguiendo con el mismo ejemplo, indicamos como anchura para las nuevas imágenes no 1500 sino 1501, «convert» generará una primera imagen con la anchura indicada, y una segunda imagen con el resto, es decir: la primera imagen tendría 1501 píxeles de anchura y la segunda sólo 1500. Pero como un píxel de diferencia en la anchura no es perceptible a la vista, esto parece mejor solución.

Conclusión: si la anchura de la imagen original es par, como parámetro de «-crop» hay que indicar exactamente la mitad. Pero si es impar, hay que indicar la mitad redondeada hasta el entero inmediatamente superior.

**OTRA NOTA:** Si el número de hojas a procesar es muy alto, es fácil que el ordenador quede sin memoria para ejecutar esta orden, lo que se traduciría en un bloqueo temporal del mismo hasta que, tras un rato más o menos largo, el sistema nos informaría de que ha decidido matar el proceso. También es posible que aunque el proceso no llegue a ser "matado", tarde mucho tiempo en ejecutarse, dejando el ordenador muy ralentizado durante todo ese tiempo. Para evitar esto hay varios procedimientos:

1. En primer lugar podemos procesar los ficheros «`unpap`» por lotes. Por ejemplo, podríamos indicar, en lugar de «`unpap*.pbm`», «`unpap0*.pbm`», lo que haría que sólo se procesaran los 100 primeros ficheros, luego reejecutaríamos la orden con «`unpap1*.pbm`». etc. En este caso debe tenerse en cuenta por cada cien ficheros de entrada se generarán 200 ficheros de salida (o 300 si estamos en el caso de la nota anterior), por lo que puede ser una buena idea aumentar una cifra al fichero de destino, indicando «`part0%03d.pbm`». Si nuestro ordenador ni siquiera tiene suficiente memoria para partir de golpe 100 ficheros, podemos crear un directorio temporal, e ir llevando a él los ficheros originales para procesarlos. Podemos hacerlo por lotes de 10, de 20, de 30... según cómo vaya respondiendo nuestro ordenador.
2. Otra solución sería escribir un bucle de *bash* que fuera recortando los originales de uno en uno. Si bien en este caso debe tenerse en cuenta una peculiaridad de «`convert`», y es que cuando se ejecuta la opción -crop sobre un solo fichero, en lugar de generar dos ficheros, cada uno con una parte de la hoja, generará un único fichero con la parte de la hoja original que se le haya indicado. Por lo tanto el bucle debería primero extraer la página del lado izquierdo de la hoja y después extraer la del lado derecho. El siguiente bucle funcionaría correctamente:

   ```
   f=-1
   for i in `ls unpap*.pbm`; do
   f=$((f+1))
   convert -crop 1082x1436+0+0 +repage $i $f.pbm
   f=$((f+1))
   convert -crop 1082x1436+1082+0 +repage $i $f.pbm
   done
   ```

   Obsérvese que en el bucle se ejecuta «`convert`» dos veces y en ambas para recortar una parte de la página original de 1082x1436. Pero la primera vez el recorte empieza en el punto 0,0 de la imagen (que se corresponde que la esquina superior izquierda) y la segunda vez el recorte empieza en el punto 1082, que es donde empieza la mitad derecha de la imagen.

### 2.2.3. Detección del contenido de las páginas

El siguiente paso consiste en la detección del contenido de las páginas. Esto lo hacemos porque nos permitirá encontrar algunos pequeños fallos de «`unpaper`»: manchas de la imagen que han sido tomadas por parte del texto, pero, sobre todo, porque nos permitirá formar correctamente las páginas de salida del libro digitalizado.

#### Cuestión previa: algunas ideas en torno al diseño "corriente" de las páginas de un libro.

Si analizamos un libro "normal", cuyo contenido sea, básicamente, texto, veremos que en las páginas se puede distinguir el texto, propiamente dicho, y los márgenes. En terminología tipográfica a la zona impresa se la llama «*mancha*», y al rectángulo ideal en el que se encierra la "mancha" se le denomina «*caja*». Si el libro está bien diseñado, la caja tendrá aproximadamente las mismas dimensiones en todas las páginas "normales". Puede haber

pequeñas diferencias; pero muy pequeñas. No es normal que la anchura de la caja difiera en más de 10 o 20 píxeles, lo cual es apenas imperceptible a la vista. Y si bien en cuanto a la altura puede haber diferencias algo mayores, éstas raramente alcanzan los 100 píxeles. Esto, claro es, en cuanto a las páginas "normales". Junto a ellas hay páginas que tienen menos texto, o en las que el texto está distribuido de una forma especial, y por lo tanto tienen una caja distinta. Por ejemplo: una página en la que aparezca un poema, o una página que contenga la dedicatoria, o una cita, o la página inicial o final de un capítulo o sección: En las páginas finales de los capítulos raramente la "mancha" ocupa toda la página, y en las páginas iniciales es costumbre desplazar la "mancha" hacia abajo dejando en blanco aproximadamente el primer tercio de la página. A veces incluso algo más de un tercio.

La caja, por otra parte, se enmarca en la página, pero habitualmente no se encuentra "anclada" exactamente en el centro de la página, sino que en la mayor parte de los casos la caja queda anclada en la parte superior, de tal modo que la distancia desde el borde superior de la página hasta el inicio de la mancha suele ser siempre la misma; pero la distancia entre el final de la mancha y el borde inferior, admite una mayor variación. El anclar la mayor parte de las páginas al borde superior del margen ayuda a que visualmente las pequeñas diferencias de tamaño vertical de la caja, queden más disimuladas; porque como el sentido de la lectura es de arriba hacia abajo, normalmente los ojos se detienen en la parte inferior de la página menos tiempo que en la parte superior.

Pero, aunque en la mayor parte de las páginas la caja quede anclada en el borde superior, en todo libro hay páginas con un anclaje distinto. En algunos libros incluso es posible que todas las páginas se anclen al centro. Esto ocurre, por ejemplo, en gran parte de los libros de poesía; sobre todo cuando los poemas ocupan una sola página.

De lo anterior se deduce que para que nuestro libro quede bien debemos:

1. Medir el tamaño de la caja que precisaremos para almacenar en ella todas nuestras páginas, para lo cual hay que obtener una versión de ellas en la que se hayan suprimido los márgenes, y luego medir las dimensiones de cada imagen para detectar cuál es la más ancha y cuál la más alta.

2. Catalogar las páginas de nuestro libro atendiendo al anclaje de su caja. La mayor parte de las páginas irán ancladas en su parte superior. Pero algunas se anclarán en el centro de la página y otras en la parte inferior.

**Procedimientos.**

Estos dos objetivos se pueden obtener de una forma relativamente sencilla. Para ello procederemos a eliminar totalmente de nuestras páginas la parte en blanco que enmarca el texto. Eso nos permitirá con facilidad medir la caja de texto que necesitamos y catalogar las imágenes atendiendo al anclaje que necesitan. Para ello necesitaremos usar de nuevo «`convert`» o «`mogrify`», «`identify`» y, para el ajuste fino, el «`gimp`». También ne-

cesitaremos un buen editor de texto que trabaje con expresiones regulares, por ejemplo, «vim».

El primer paso consiste en recortar las páginas, descartando totalmente los margenes blancos de las mismas. Esto se consigue con la opción «-trim» de «convert» y de «mogrify». Por ejemplo:

```
convert -trim part*.pbm rec%03d.pbm
```

> **NOTA:** Una vez más hay que recordar los problemas de memoria y de posible bloqueo del ordenador que el procesado masivo de imágenes puede ocasionar. Como ya se ha explicado, usando «mogrify» se evita este problema, aunque, a cambio, los ficheros originales se pierden. Si no estamos seguros de quererlos descartar totalmente, y tenemos poca memoria disponible en el sistema, es preferible usar «convert», tal vez con algún bucle de *bash* para que los ficheros sean procesados de uno en uno.

La función *trim* provoca el autorrecorte de las imágenes: el programa recorre la imagen de borde a borde comprobando si todos los píxeles de la primera fila (o columna) tienen el mismo color que el fondo de la imagen (por defecto blanco) en cuyo caso los borra y pasa a analizar la siguiente fila (o columna)... así hasta que encuentre una fila o columna en donde haya algún píxel en un color distinto. Esto significa que si «unpaper» hizo bien su trabajo, cada uno de los bordes de la imagen se irán desplazando hasta llegar al punto en el que empieza el texto, y si la página original estaba en blanco, la imagen entera debería desaparecer, caso este en el que «convert» informará que ha encontrado un fichero sin imagen. Aunque no siempre que la página original estaba en blanco «-trim» consigue eliminarla totalmente, pues por muy bien que haya hecho su trabajo «unpaper» es muy fácil que en la imagen obtenida del escáner haya varios píxeles aislados, casi imperceptibles a simple vista, pero suficientes para que «-trim» los identifique y sólo recorte hasta ellos.

> Respecto a las páginas del libro original que estan totalmente en blanco, debemos decidir qué hacer en nuestro libro digital. Borrarlas tiene el inconveniente de que introduce una diferencia entre el libro físico y el digitalizado, en virtud de la cual la numeración que consta en las páginas del libro digitalizado no coincide con el número de página del correspondiente fichero. Pero, por otro lado, mantener páginas totalmente en blanco es un engorro, y una incomodidad para la lectura.
>
> Yo soy partidario de borrarlas porque considero que la función de una página en blanco dentro de un libro estriba exclusivamente en asegurarse de que ciertas secciones del libro empiecen siempre en una página impar; pero eso sólo tiene sentido en un libro "impreso". Y la incomodidad de que la numeración real de las páginas en el fichero no coincida con la que aparece impresa en ellas, se puede suavizar generando un *outline* para el fichero. Véase, al respecto la sección 4.2 de este documento.

Una vez que tenemos los ficheros recortados conviene analizarlos. Para ello necesitaríamos información sobre la anchura y altura de cada uno de ellos, para lo cual generaremos un fichero de texto en el que almacenaremos la información que, de cada fichero, nos proporcione «identify». Esto se obtiene con el siguiente bucle de *bash*:

```
for i in `ls rec*.pbm`; do
identify $i >> dimen.txt
done
```

Tras esta orden se habrá generado en nuestro directorio de trabajo un nuevo fichero llamado «`dimen.txt`», y en él habrá una línea con información sobre cada uno de nuestros ficheros. Si recordamos nuestro anterior uso de «`identify`» la información que nos interesa (las dimensiones de la imagen) se encuentra en tercer lugar, tras el nombre del fichero y su formato interno (en nuestro ejemplo PBM). Con un editor de texto que admita expresiones regulares es fácil desplazar esa información al principio de la línea, de tal modo que si luego ordenamos alfabéticamente las líneas del fichero tendremos un listado de nuestros ficheros, ordenados por la anchura de la imagen que almacenan.

Con el editor «`vim`», por ejemplo, deberíamos ejecutar los siguientes comandos, asumiendo que nuestros ficheros son PBM:

```
:%s/^\([^ ]\+\) PBM \([^ ]\+\)/\2 \1/
:%s/^...x/0&/
:%sort
```

La primera orden es una expresión regular que traslada al principio de la línea la información relativa a las dimensiones de la imagen[6]. La segunda orden es una nueva expresión regular que detecta si el número por el que empiezan todas las líneas, en algún caso es de sólo tres cifras, en cuyo caso añade un 0 al principio de dicha línea, con lo que se consigue que todas las anchuras tengan el mismo número de cifras, de tal modo que la ordenación alfabética coincida con la numérica. Tras ello la tercera instrucción ordena alfabéticamente (y, como hemos visto, también numéricamente) todas las líneas del fichero.

Resultado de todo lo anterior: en nuestro fichero «`dimen.txt`» tenemos un listado de los ficheros de que consta nuestro proyecto, ordenado por la anchura de la caja de texto en cada uno de ellos. A partir de aquí conviene ir examinando las primeras imágenes del listado y las últimas:

[6]Para aquellos a quienes interesen el manejo de «`vim`» y el lenguaje de las expresiones regulares, podemos destriparla de la siguiente manera: El primer carácter («`:`») es el comando de «`vim`» que indica que a continuación viene una orden de línea de comado. El segundo («`%`») indica que esa orden se debe aplicar a todas las líneas del documento y el tercero («`s`») contiene el nombre del comando a aplicar: "s", que es un diminutivo de "*substitute*". El carácter que sigue «`/`» indica que a partir de ahí empieza la expresión regular que hay que buscar, y que consta de tres partes: la primera recoge en un grupo el texto que va desde el principio de la línea hasta el primer espacio en blanco, la segunda recoge el texto «` PBM `», el cual no se almacena en ningún grupo porque es una parte de la línea que vamos a descartar, y la tercera recoge el texto que va, desde el espacio en blanco posterior a PBM hasta el siguiente espacio en blanco. Ese fragmento de la línea, que es el que contiene la información relativa a las dimensiones de la imagen, se almacena en un segundo grupo. El siguiente carácter «`/`» indica que ahí termina la expresión a buscar y empieza la expresión de sustitución, la cual es muy simple, escribe en primer lugar el texto almacenado en el segundo grupo («`\2`») —o sea, las dimensiones de la imagen— y, a continuación, el texto almacenado en el primer grupo («`\1`») —o, lo que es lo mismo, el nombre del fichero—. Entre ambos textos se escribe un espacio en blanco.

- Las primeras líneas del listado representan páginas que, dada su anchura inferior a la normal, muy posiblemente contengan exclusivamente un título o una dedicatoria, o, tal vez, sea una página en blanco en la que quedó algún píxel perdido y que por ello «`-trim`» no pudo identificarla correctamente como página en blanco. En este último caso podemos directamente borrar el fichero (si eso es lo que hemos decidido hacer con las páginas en blanco). En los demás casos mi consejo es mover el fichero en cuestión a un subdirectorio en el que almacenaremos los ficheros cuyo anclaje no vaya a ser el normal. De hecho en esta fase yo suelo generar dos subdirectorios: Uno para las páginas cuyo anclaje deba ser al centro, y otro para las páginas cuyo anclaje deba ser a la parte inferior.

- Las últimas líneas del listado representan páginas en las que, dada su anchura superior a la normal, probablemente una mancha de la imagen, o un "píxel loco", inevitable en el escaneado, se ha salvado de la limpieza hecha por «`unpaper`». Con estas imágenes podemos hacer dos cosas: corregirlas a mano (con el «`gimp`») o dejarlas como están. Hacer una cosa u otra depende de lo meticulosos que pretendamos ser con nuestra digitalización y, por supuesto, del número de páginas que se desvían de la anchura normal, y de cuál sea la cantidad de ese desvío. Pues una anchura que sólo sea "ligeramente" superior a la normal, apenas será perceptible, y afectará en muy poco al aspecto final de nuestro libro.

  Si optamos por corregirlas con el «`gimp`», tras limpiar la mancha hay que, en el mismo «`gimp`», volver a efectuar el autorrecorte de la imagen y guardarla.

  > Las últimas versiones del «`gimp`» sólo permiten guardar imágenes en el formato nativo de esta aplicación (XFC), lo que me parece una regresión del programa que, para arreglar ligeramente una imagen, obliga a mucho más "tecleado". Para grabar los cambios en la imagen PBM hay que usar la función: exportar (CTRL-MAY-e) [1]: el cuadro de diálogo nos ofrecerá como nombre por defecto el del fichero que estamos editando; confirmamos [2]; entonces el «`gimp`» ¡verá que ese fichero ya existe! y nos pedirá confirmación para sobreescribirlo; se la damos [3]; a continuación nos volverá a preguntar respecto del tipo de exportación (el cual cambia según a qué formato estemos exportando); volvemos a confirmar lo que nos sugiere por defecto [4], y cerramos la imagen pulsanco CTRL-W [5]: como no la hemos grabado en formato XFC nos preguntará si queremos descartar los cambios, indicamos que sí, [6 y 7][7] y salimos del programa: ¡Lo que antes se hacía simplemente pulsando CTRL-S (grabar los cambios hechos en la imagen editada) seguido de CTRL-W para cerrar la imagen, ahora requiere al menos siete golpes de teclado, o varios movimientos de ratón! No veo dónde está la mejora. A mi, personalmente, me parece un estorbo esta forma de trabajar.

Asumiendo que es normal una ligera variación de la anchura de la caja entre distintas páginas, no es preciso que examinemos individualmente la anchura de todas las páginas

[7] Entre corchetes he ido contando los golpes de teclado. En este último paso no cuento un sólo golpe sino dos, porque cuando se nos pregunta si queremos descartar los cambios, la opción que se nos muestra por defecto es la de no descartarlos, de modo que hay que seleccionar el botón de cerrar sin grabar y luego pulsarlo: dos golpes de teclado.

del libro. Bastará con que centremos la atención en las más estrechas y en las más anchas. Deberemos también anotar, para su posterior uso, la mayor de las anchuras que, tras nuestro proceso de ajuste, finalmente se mantenga. Esa anchura nos servirá para diseñar la caja final de nuestras páginas.

Cuando hayamos terminado de analizar los ficheros por su anchura, procederemos al análisis por su altura. Para ello, en «vim», las siguientes instrucciones

```
:%s/^[^x]\+x//
:%sort
```

borrarán de todas las líneas del fichero de texto la información referente a la anchura de las imágenes y las reordenará atendiendo ahora a su altura. Al igual que hicimos con la anchura conviene examinar las imágenes que tengan menos altura y las que tengan más altura. Las que tienen menos altura son candidatas a requerir algún tipo de anclaje especial, y las que tienen más altura posiblemente sean imágenes en las que una mancha ha sido tomada como parte del texto, que podemos libremente corregir (con el «gimp») o ignorar.

Al terminar este proceso, que tardará más o menos dependiendo, en primer lugar, de lo correcto que haya sido el escaneado, y, en segundo lugar, de lo meticulosos que queramos ser con respecto a los bordes, tendremos la siguiente información:

1. Las dimensiones de la caja de texto que necesitamos para almacenar en ella todas nuestras páginas que se corresponderán con la máxima anchura y con la máxima altura de las concretas páginas.

2. Una correcta catalogación y separación de nuestras páginas, según la cual sabremos cada una de ellas cómo debe anclarse. En el directorio principal tendremos las páginas que hay que anclar en la parte superior, y en dos subdirectorios adicionales tendremos, respectivamente, las páginas que hay que anclar en el centro o en la parte inferior.

Tras las operaciones anteriores, ya estamos en condiciones de dibujar un lienzo del mismo tamaño para todas las imágenes, y ubicar cada una de ellas en el punto correcto de dicho lienzo. Las siguientes instrucciones producirían ese efecto, suponiendo que el ancho máximo de imagen detectado en el paso anterior fuera de 1400, el largo máximo de 2000 y que tengamos en el directorio actual las imágenes que deban anclarse en la parte superior, en un directorio llamado “cent” las imágenes que se deben anclar al centro y en un directorio llamado “sur” las que se deben anclar en el borde inferior de la página:

```
mogrify -gravity North -background white -extent 1400x2000 rec*.pbm
mogrify -gravity Center -background white -extent 1400x2000 cent/*.pbm
mogrify -gravity South -background white -extent 1400x2000 sur/*.pbm
mv cent/*.pbm ./
mv sur/*.pbm ./
```

Las tres primeras órdenes generan un lienzo del mismo tamaño y sobre él superponen una imagen. La diferencia entre cada una de ellas está en a qué imágenes se aplica y cómo se alinea la imagen sobre el lienzo, en el norte, en el centro o en el sur. La opción «`-gravity`» de «`mogrify`» o de («`convert`») usa los puntos cardinales para indicar cómo superponer una imagen sobre otra: podemos indicar, norte, sur, este, oeste o centro; pero también noroeste o sureste si así lo deseamos.

> La clave de estas tres instrucciones está en la opción «`-extent`» de «`mogrify`» (o de «`convert`»). Este operador dibuja un lienzo del tamaño indicado sobre el que superpone la imagen original. Siempre que se usa «`-extent`» conviene usar también «`-gravity`» para indicar en qué punto cardinal hay que anclar la imagen original. La opción «`-background`» es necesaria si el nuevo tamaño es mayor que el original, pues indica a «`mogrify`» con qué color rellenar el fondo.

Las dos últimas instrucciones vuelven a traer a nuestro directorio de trabajo los ficheros que en el paso anterior sacamos de él, puesto que, como ya están correctamente anclados, y ahora, de nuevo todas las imágenes tienen las mismas dimensiones, ya no necesitamos diferenciarlos.

Al llegar aquí tenemos todas las imágenes del mismo tamaño, y cada una de ellas alineada en un lugar distinto de la página. Ya sólo falta añadir un margen alrededor de nuestra caja de texto. El tamaño del margen va en gustos. A mí me gusta calcularlo como un porcentaje de la caja. Por ejemplo, si queremos que los márgenes ocupen un 10 %, en el ejemplo con el que venimos trabajando tendríamos que añadir 140 píxeles a la anchura y 200 a la altura. Esto lo obtendríamos con la siguiente instrucción:

```
mogrify -gravity Center -background white -extent 1540x2200 rec*.pbm
```

Y ya está. Ya tenemos todas nuestras páginas preparadas para componer el libro. Podemos saltar a la sección 2.4.

## 2.3. Mejora de los originales con scantailor

Aunque el procedimiento que acabamos de describir funciona razonablemente bien, tiene el inconveniente de que implica el uso de muchas herramientas las cuales, además, son poco intuitivas. Creo que conocerlo ilustra bastante sobre el manejo masivo de imágenes con herramientas de software libre, pero eso no lo convierte en el procedimiento óptimo.

Existe una herramienta gráfica, también de software libre, que realiza ella sola todos los pasos que hasta ahora hemos visto, y lo hace razonablemente bien con su configuración por defecto. Pero incluso cuando dicha configuración falla, la corrección del fallo es bastante sencilla y puramente visual. La herramienta en cuestión es «`scantailor`» y se encuentra en los repositorios normales de Debian. En `http://scantailor.org/` es posible conseguir información sobre ella.

### 2.3.1. Crear un proyecto nuevo

Para trabajar con «scantailor» hay que empezar por crear un proyecto nuevo. Al iniciar el programa este ya nos pregunta si deseamos crear un proyecto nuevo o abrir un proyecto anterior. Cuando se crea un proyecto nuevo se muestra el cuadro de diálogo de la figura 2 en el cuál, como se ve, hay que indicar el directorio donde se encuentran los ficheros de imagen sobre los que construir nuestro libro (directorio de entrada) y el directorio donde se generarán las páginas ya tratadas por «scantailor» (directorio de salida). El programa lee los ficheros que hay en el directorio de entrada y los muestra en dos listados: a la izquierda se muestran los que no tienen un formato apto para «scantailor» y a la derecha los que sí lo tienen. El proyecto constará de las imágenes mostradas en el listado de la derecha, aunque podemos traspasar alguna de esas imágenes al listado de la izquierda para evitar que forme parte del proyecto.

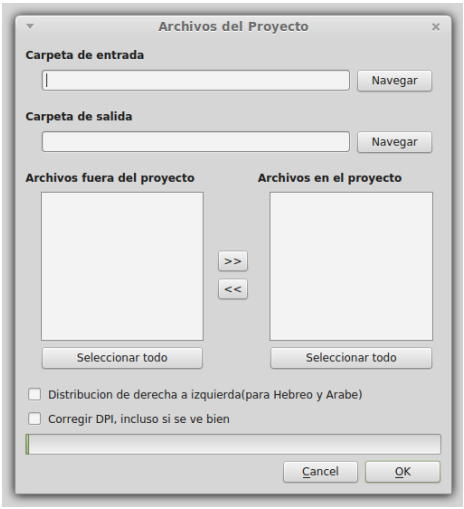


Figura 2: Cuadro de diálogo para crear un nuevo proyecto

Los formatos de imagen que «scantailor» admite son TIF, TIFF, PNG, JPG y JPEG.

Al hacer click sobre el botón OK, tras haber indicado un directorio de entrada en el que exista alguna imagen válida, «scantailor» lee los ficheros que componen el proyecto y se pone en marcha. Hay casos, no obstante, en que alguna de las imágenes no informa sobre su resolución, o a «scantailor» le parece que la información contenida en la imagen es errónea, caso este en el que se muestra un cuadro de diálogo que permite ajustar individualmente la resolución de cada una de las imágenes sobre las que se plantean dudas.

### 2.3.2. Interfaz de scantailor

La interfaz de «scantailor» es muy sencilla y se puede observar en la figura 3. La pantalla se divide en tres zonas. En la zona izquierda hay una especie de menú que recoge

las acciones principales del programa, así como las opciones de configuración para cada una de ellas. En la parte central se ve una imagen de la página en la que se está trabajando en ese momento, y en la parte derecha aparece un listado de imágenes en miniatura de las hojas o páginas de que consta el proyecto.

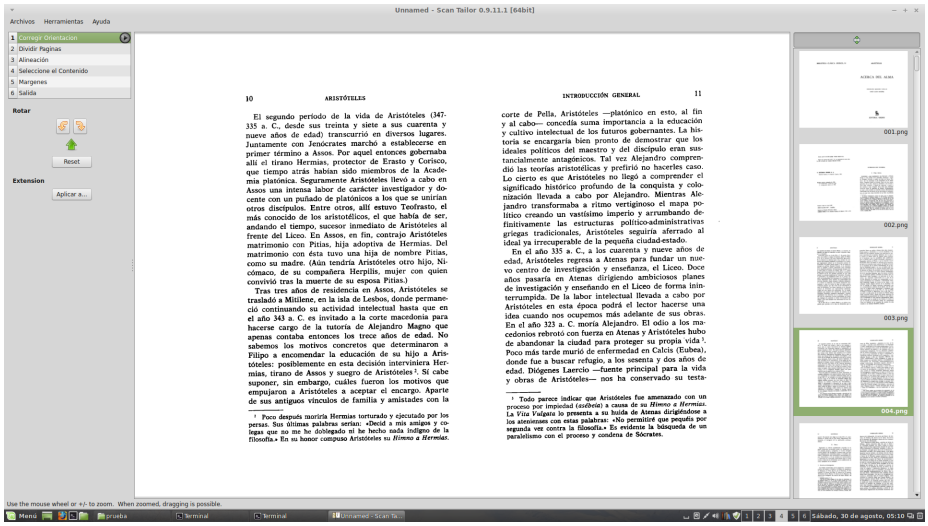

Figura 3: Interfaz gráfica de scantailor

En cada momento hay seleccionada al menos una página, que es la que se muestra en la zona central. Y RePág y AvPág seleccionarán, respectivamente, a la página anterior o posterior a la actual. En el listado de imágenes de la derecha podemos movernos entre ellas con más rapidez y sin necesidad de recorrerlas de un modo secuencial. El botón derecho del ratón sobre ese listado nos permite añadir una página al proyecto (delante o detrás de la página del listado sobre la que se haya hecho click) o eliminar la página del proyecto. Pulsando la tecla mayúsculas al tiempo que se hace click sobre una página del listado seleccionaremos todas las páginas comprendidas entre aquella sobre la que se ha hecho click y la anteriormente seleccionada, y mediante la combinación CTRL-click podemos seleccionar varias páginas no consecutivas. En la parte inferior del listado hay un menú que nos permite indicar cómo queremos ordenar las páginas: por su orden natural (el orden alfabético en el que se encontraban en el directorio de entrada) por su anchura o por su altura. La ordenación por anchura y por altura nos será bastante útil más adelante.

### 2.3.3. Tratamiento de las imágenes con «scantailor»

Las tareas que «scantailor» realiza sobre nuestras imágenes originales, y que se muestran en el listado de tareas de la parte izquierda del interfaz del programa son las que muestra la figura 4:

Haciendo click sobre cada una de ellas se muestran, en la parte inferior de ese mismo menú, las opciones de configuración de dicha tarea. Para cada opción de configuración

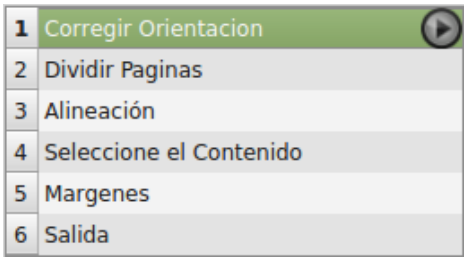


Figura 4: Listado de tareas que realiza «scantailor»

podemos indicar si queremos aplicar tal configuración sólo a la página actual, a todas las páginas, a la página actual y a todas las que la siguen, pero no a las anteriores, o a todas las páginas salvo la actual. Si hay más de una página seleccionada también podemos indicar que la configuración se aplique a todas las seleccionadas.

Cuando indicamos la configuración de alguna de las tareas, ésta se aplica inmediatamente a la página actual (la que se muestra en la parte central de la interfaz). Para aplicar el cambio al resto de páginas a las que deba aplicarse, hay que pulsar la pequeña flecha que se muestra en el lado derecho de la línea del menú correspondiente a la tarea actualmente activa.

Por otra parte, aunque nos podemos mover libremente entre las distintas tareas, seleccionando cualquiera de ellas, hay que tener en cuenta que «scantailor» siempre las realiza todas, y lo hace en el orden en el que se muestran en el menú. De manera que si, por ejemplo, nos colocamos directamente en la quinta tarea (fijar los márgenes), para la imagen seleccionada «scantailor» realizará todas las tareas previas antes de mostrar la imagen preparada para fijarle los márgenes.

> La libertad para movernos entre las distintas tareas es absoluta respecto de las cinco primeras, pero, en cuanto a la sexta, tiene un límite: La sexta y última de las tareas no es mostrada por «scantailor» hasta que se hayan realizado todas las tareas anteriores sobre todas las páginas del proyecto. Eso es porque sólo en ese momento está «scantailor» en condiciones de conocer cuál debe ser el tamaño real de las páginas de salida.

En cuanto a las concretas tareas que «scantailor» realiza, su nombre es bastante claro, y también lo son, en general las opciones de configuración de cada una de ellas. No creo que nadie que haya podido seguirme en los apartados anteriores a través de los intríngulis de «convert», tenga problemas para manejar «scantailor». Por ello, sin entrar en demasiados detalles, haré simplemente unas indicaciones generales de cada una de ellas:

**Corrección de la orientación:** Implica que «scantailor» analizará la imagen para asegurarse de que está orientada correctamente, es decir: si se trata de una página de texto, que su orientación coincide con el sentido de la lectura. De no ser así, la página será rotada hasta hacerla coincidir con tal sentido.

Podemos cambiar la orientación fijada por «scantailor» en cada página, rotándola a la derecha o a la izquierda.

**División de páginas:** En este punto «scantailor» detecta si el escaneo se hizo a doble página y, en el caso de que así fuera, el punto de la hoja por el que las páginas deben dividirse. Esa detección en general se hace bastante bien, pero en páginas con poco texto a veces falla. Para corregir la detección automática hecha por «scantailor» basta con arrastrar la línea de separación entre las páginas que se muestra en la zona central de la interfaz.

Si no tenemos imágenes a doble página, aunque no podemos saltarnos esta tarea, si podemos establecer para todas las páginas un diseño concreto, de tal modo que «scantailor» se ahorrará el tiempo necesario para intentar detectar si la hoja encierra o no dos páginas. Para ello en el menú de configuración de distribución de página, que se muestra bajo el menú de tareas cuando se selecciona la división de páginas, debemos hacer click sobre el icono que representa una sola página y en el cuadro de diálogo que aparece al pulsar el botón «cambiar» seleccionar "modo manual" y "extensión todas las páginas".

**Alineación:** La alineación de las páginas pretende asegurarse de que los renglones están totalmente horizontales. Para ello se muestra una cuadrícula sobre la que se superpone la imagen de la página y una especie de volante azul que podemos girar a derecha o izquierda, tal y como se muestra en la figura 5.

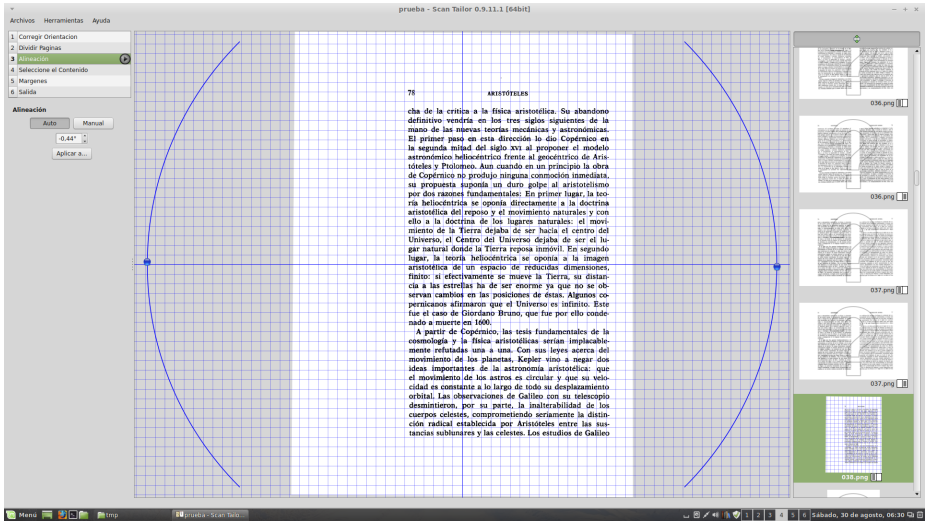


Figura 5: Alineación de la página

Aunque podemos cambiar individualmente la alineación de cada página, en general la alineación automática hecha por «scantailor» es bastante correcta. Hay páginas, no obstante, en las que resulta imposible una alineación perfecta de la imagen debido al efecto de curvatura de los renglones que el escáner produce en ocasiones.

Eso tiene mal arreglo en esta fase, en la que debemos conformarnos con la alineación que en general permita una mejor lectura de la página.

**Selección de contenido:** Esta es posiblemente la tarea que más tiempo nos ocupará, pues en ella el funcionamiento automático de «`scantailor`» es menos preciso. En teoría, para cada página, el programa detectará lo que es contenido de la página y lo introducirá en un rectángulo, el cual debe encerrar todo el texto y sólo el texto. Hay que procurar que no queden márgenes.

En la práctica, no obstante, en muchas ocasiones «`scantailor`» se deja engañar por manchas en la imagen que son consideradas parte del texto, o, por el contrario, se descartan algunas partes de la página por ser consideradas manchas del texto. Esto último ocurre, según mi experiencia, sobre todo en los encabezados y pies de página cuando están demasiado separados del cuerpo del texto.

> Si se observa, esta «tarea» es equivalente a la que, en el procedimiento realizado mediante órdenes de línea de comando, se obtenía haciendo «`-trim`» sobre la imagen, explicado en la sección 2.2.3. Se trata de dejar exclusivamente el texto de la página, para poder medir la caja de contenido que necesitaremos. Y, al igual que entonces pasaba, los defectos en el escaneo se traducen en errores en el autorrecortado de la imagen.

Aunque aquí es donde más errores se producen, a cambio repararlos es muy sencillo: basta con arrastrar los bordes del rectángulo que encierra el cuerpo de la página, para ajustarlos al contenido real de la misma. Si queremos ser absolutamente precisos y alinear totalmente el rectángulo con el borde de la imagen, sin dejar márgenes, podemos hacer un zoom sobre cualquier zona de la misma, simplemente colocando en ella el cursor del razón y girando hacia arriba la rueda del mismo.

Lo ideal es revisar una a una las páginas de nuestro proyecto para asegurarnos de que la detección ha funcionado bien en todo caso. No obstante, si tenemos prisa, hay un atajo que funcionará bien la mayor parte de las veces y que consiste en ordenar las imágenes del proyecto, no por su orden natural, sino, en primer lugar por su anchura y, en segundo lugar, por su altura. Si asumimos que en un libro bien diseñado la caja de texto de la mayor parte de las páginas tendrá el mismo tamaño, resulta que es en las imágenes con la menor anchura o altura donde resulta más probable que se haya descartado una parte real de la imagen, confundiéndola con una mancha, y es en las imágenes de mayor anchura o altura donde es más probable que una mancha haya sido confundida con contenido real de la página.

> Si se observa, en realidad este atajo es el mismo que hemos aplicado en la sección 2.2.3, donde se explica cómo hacer esta misma tarea con herramientas de línea de comando: es en las páginas cuya anchura o altura difiere de la "normal" en donde es más probable que haya habido algún error, o que se trate de una página que requiere un anclaje especial.
>
> Esta revisión de páginas atendiendo a su anchura y a su altura, tiene, sin embargo, un problema en «`scantailor`» y es que, cuando se alteran las dimensiones de la caja de contenido de una página concreta, el programa inmediatamente la reordena... con lo que, para seguir con el examen, hay que buscar el punto en el que estaba antes la imagen, o

volver a empezar desde el principio. Sería bueno que en una futura revisión del programa, una alteración de las dimensiones de la caja no provocara una inmediata reubicación de la imagen, pues hay que suponer que si alguien decide examinar las imágenes atendiendo a su anchura o a su altura, querrá hacerlo con todas. Pero si la imagen seleccionada cambia su lugar dentro de la lista, eso es mucho más díficil. En mi experiencia, como más tiempo se pierde en «`scantailor`» es reajustando márgenes... no tanto porque aquí es donde más errores puede haber, sino porque cada reajuste exige buscar el punto de la lista en donde estábamos.

**Márgenes:** Una vez que se ha detectado el contenido real de la página, el próximo paso consiste en extraerlo del original, añadirle los márgenes que se desee, y alinear el contenido con respecto a algún punto de la página. Todo ello lo podemos hacer en este paso. En primer lugar se puede indicar la medida de los márgenes superior, inferior, izquierdo y derecho y, en segundo lugar, se puede indicar la alineación del contenido, con respecto a la página. Para ello basta con pulsar sobre alguno de los iconos que, para cada página, se muestran en la figura 5.


Figura 6: Alineación del contenido dentro de la página

Por defecto se asume que la la alineación del contenido se hará con respecto a la parte superior de la página real, como suele ocurrir en los libros, y se deja que ciertas páginas individuales se alineen de otro modo. Respecto a esta alineación, me remito a lo que ya expliqué en el apartado 2.2.3 de este documento.

Por otro lado debe tenerse en cuenta que el tamaño real que tendrá cada página no consistirá exclusivamente en la suma del tamaño de su contenido y de los márgenes indicados en este apartado, sino que, si queremos que todas las páginas de nuestro libro tengan el mismo tamaño, dado que no todas las cajas de contenido tienen la misma anchura y altura, los márgenes en realidad se aplicarán a una caja de contenido ideal compuesta por la anchura de la caja más ancha, y la altura de la más alta.

Hay una opción en la configuración de esta tarea, denominada "*Igualar en tamaño a las demás páginas*" que si se desmarca, provoca que las dimensiones de cada página sean exclusivamente la suma de su contenido y los márgenes indicados. Pero hacer

esto no creo que sea buena idea: todos los libros que conozco constan de páginas del mismo tamaño.

**Salida:** El último paso consiste en generar un fichero con la imagen correspondiente a las páginas ya procesadas. Esta opción, a diferencia de las anteriores, sólo está disponible cuando se ha generado la caja de contenido de todas las páginas del proyecto, ya que, hasta ese instante, «`scantailor`» no puede saber qué tamaño deben tener las páginas de salida.

Una vez disponible esta opción, conforme vayamos seleccionando páginas de nuestro proyecto, se irá generando un fichero de formato TIF en el directorio de salida indicado al crear el proyecto. Este directorio, por defecto, se denomina “out”.

Aparte del tamaño de las páginas (que no podemos controlar directamente en esta fase, pues depende de la caja de contenido ideal generada en las fases anteriores), las opciones de configuración de esta tarea son las que se muestran en la figura 7:

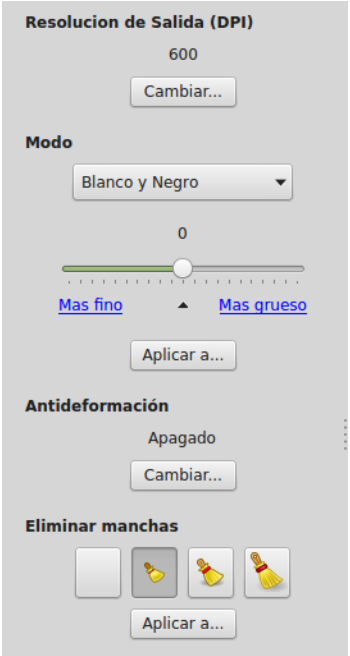


Figura 7: Opciones de configuración de los ficheros a generar

- **Resolución:** De ella depende la calidad de la imagen. Por defecto se propone una resolución de 600 DPI (PPP, dicho en español), pero se puede elegir cualquier otra. Personalmente creo que, si el escáner era correcto, en la mayor parte de los casos basta con una resolución de 300 PPP. Piénsese que a mayor resolución, mayor tamaño tendrán los ficheros generados y, por lo tanto, mayor tamaño tendrá el libro final. Hay que buscar un equilibrio entre la calidad y la manejabilidad.

- **Modo:** Podemos elegir entre imágenes en blanco y negro (por defecto), imágenes en color/escala de grises o mezcla de ambas: unas páginas en blanco y negro y otras en color. Si elegimos imágenes en blanco y negro, podemos también seleccionar el grosor del trazo y si elegimos color/escala de grises podemos indicar si queremos que los márgenes sean o no blancos, o dejarlos con el color de la imagen, y, si los márgenes se deciden blancos, si queremos normalizar el color del fondo de las imágenes.
- **Antideformación:** El filtro antideformación en «`scantailor`» es experimental y, por defecto, está desactivado. En teoría corrige el efecto de curvatura de algunos renglones, pero en ciertas páginas produce efectos inesperados y sorprendentes. Mi consejo es que nunca se aplique de forma automática, sino exclusivamente a las páginas que lo necesiten, controlando los resultados del mismo. Si se decide aplicarlo masivamente es muy importante que nos aseguremos de examinar individualmente todas las páginas para evitar sorpresas.
- **Eliminar manchas:** El filtro antimanchas intenta detectar y eliminar las manchas de la imagen producidas por el escáner. Puede desactivarse, o activarse con hasta tres niveles de intensidad. Por defecto se encuentra activado con el nivel de intensidad más bajo. La ventaja de que «`scantailor`» sea un programa gráfico es que podemos ver inmediatamente, en la imagen central de la pantalla, los resultados de aumentar, reducir o eliminar totalmente este filtro.

Cuando hayamos acometido todas las tareas, antes de salir de «`scantailor`» debemos asegurarnos de que se ha generado una imagen nueva para cada una de las páginas de nuestro libro. El fichero de salida de cada página no se genera hasta que, en la última tarea, se selecciona dicha página. Para asegurarnos de que se han generado todos los ficheros, estando seleccionada la última tarea («`Salida`»), debemos pulsar la flecha que en el menú de tareas hay a la derecha del nombre de la tarea activa, para forzar al programa a procesar todas las páginas.

## 2.4. Ajuste final con el *gimp*

Tanto si hemos usado «`scantailor`» como si hemos trabajado con herramientas de línea de comando, el último paso —si queremos que nuestro libro digital tenga la máxima calidad— consiste en examinar una a una las imágenes correspondientes a las páginas del libro y, si vemos algún defecto en ellas, arreglarlo individualmente con el «`gimp`» que es la mejor herramienta para la edición de ficheros gráficos que existe en el mundo del software libre y que creo que no tiene nada que envidiar a «`photoshop`» (aunque como no soy un experto en temas gráficos, tampoco estoy seguro).

En este ajuste final podemos:

- Quitar ciertas manchas que sigan estando en el interior de las páginas y que no hayan desaparecido todavía.

- Restaurar algún texto borroso, para lo cual habría que determinar de qué letras consta, irlas buscando en otra parte de la página (o en otra página), copiarlas y pegarlas en el lugar correspondiente.
- Eliminar subrayados que hubiera en el original. Eso es fácil de hacer si el subrayado realmente lo es: basta con ampliar lo suficiente la imagen para seleccionar los trazos del mismo y suprimirlos. Resulta bastante más difícil de conseguir cuando la raya que subrayaba en algún punto de su recorrido se superpone sobre el texto, pues entonces más que un subrayado es un tachado.
- Etc.

El único límite que tenemos aquí es nuestra paciencia y el tiempo de que dispongamos. Pensemos, no obstante, que en general, basta con que el libro sea cómodamente legible. Si intentamos eliminar todos los píxeles locos que como consecuencia del escaneado estén diseminados por las páginas no acabaremos nunca.

Personalmente yo en este punto, cuando lo acometo (cosa que no siempre hago), me limito a eliminar los fallos más ostensibles. Y respecto de los restantes, sólo me molesto en arreglar aquellos que me da la impresión de que dificultarán el posterior OCR, sobre el cual hablaré en la sección 4.1.

# 3. Generación del libro digitalizado

Al llegar aquí tenemos una serie de ficheros, cada uno de los cuales almacena una de las páginas de nuestro futuro libro. Ahora hay que agruparlos en un formato adecuado para un libro. Los dos candidatos principales son PDF y DJVU.

> Cuando digo que los formatos principales para un libro digitalizado son PDF y DJVU, excluyendo otros formatos como EPUB, estoy asumiendo que un libro digitalizado no es exactamente lo mismo que un libro digital, sino más bien un subtipo de estos. Podemos llamar libro digital a cualquier libro en formato electrónico. Pero libro digitalizado sólo es aquel libro digital en cuyo interior se contienen imágenes que reproducen las páginas del libro original. El libro digitalizado es siempre reproducción más o menos exacta de un libro físico y material.

## 3.1. Generación de un fichero PDF que contenga todo nuestro libro

Para crear un fichero PDF a partir de nuestras imágenes, de nuevo «`convert`» nos puede ayudar. Así si, hemos seguido el procedimiento descrito en el apartado 2.2, de tal modo que las páginas de nuestro libro se almacenan en ficheros cuyo nombre empieza por «`rec`» y cuya extensión es PBM, la orden

```
convert rec*.pbm libro.pdf
```

agrupará todas las páginas en un sólo fichero PDF llamado «libro.pdf».

Pero ese fichero puede llegar a ser gigantesco. Conviene por ello primero convertir nuestras páginas a un formato más comprimido. En mi experiencia la mayor compresión sin pérdida de calidad la obtiene el formato PNG, por tanto, asumiendo que nuestros ficheros finales se denominan «recxxx.pbm», donde «xxx» representa un número de tres cifras, las órdenes a ejecutar deberían ser:

```
mogrify -format png rec*.pbm convert *.png libro.pdf
```

> Si hemos trabajado con «scantailor» y nuestros ficheros finales tienen el formato TIF, no es probable que PNG los comprima más de lo que ya están, pues «scantailor» usa, para generar sus ficheros, tecnología LZH que es una de las que mejor compresión alcanza. Podremos por lo tanto ejecutar simplemente un «convert *.tif libro.pdf».

Aún así el fichero PDF puede ser bastante grande y, al igual que ocurre siempre que manejamos «convert», el proceso puede tardar mucho tiempo, quedando el ordenador bloqueado hasta que termine. Una alternativa puede ser la de convertir cada página a un fichero PDF individual (tal vez con un bucle de *bash*, o mediante «mogrify») y, a continuación, usar «pdftk» para juntar todos los PDF en un único fichero. Ello se conseguiría con la siguiente secuencia de comandos:

```
mogrify -format pdf *.pbm
pdftk *.pdf cat output libro.pdf
```

La primera orden genera un fichero pdf por cada fichero PBM de nuestro directorio (asumiendo que en él sólo tenemos ya las páginas finales del libro y que el formato de nuestras páginas en PBM) y la segunda utiliza «pdftk» para unir todos los ficheros PDF en un único fichero llamado «libro.pdf». Pdftk es un fantástico editor de línea de comandos para ficheros PDF, disponible también en los repositorios de prácticamente todas las distribuciones de GNU Linux.

## 3.2. Generación de un fichero DJVU

Aunque es un excelente formato, usar PDF para nuestro libro tiene algunos inconvenientes. El primero es, ya lo hemos señalado, el tamaño que tendrá el fichero: por mucho que comprimamos las páginas, es difícil que un libro digitalizado de varios cientos de páginas tenga un tamaño inferior a 10 Megabytes, lo que hace que esos ficheros sean difíciles de transmitir e inmanejables en un ordenador con pocos recursos de memoria.

Pero además, PDF tiene un grave problema adicional si estamos trabajando con software libre: son pocas las herramientas que permiten manipular los ficheros PDF. La mejor, para mi gusto, es «pdftk», pero con esta herramienta no podemos, por ejemplo, hacer un

OCR del fichero, o generar un índice virtual del mismo (*outline*). No hay, que yo sepa, ninguna herramienta libre que permita hacer eso con ficheros PDF. Por ello entiendo que la opción DJVU, aunque sea menos conocida, es preferible a la opción PDF.

DJVU (acrónimo de *Deja Vu*) es —según dice la wikipedia— un formato de archivo informático diseñado principalmente para almacenar imágenes escaneadas que se caracteriza por incorporar avanzadas tecnologías aritméticas y compresión con pérdida para imágenes bitonales[8], permitiendo que imágenes de alta calidad se almacenen en un mínimo de espacio. Dependiendo de la distribución de GNU Linux que usemos está disponible en distintos paquetes. En Debian estos paquetes son, principalmente:

- djvulibre-bin: Es el paquete básico. Incluye todas las librerías necesarias y las herramientas fundamentales para manejar este tipo de ficheros mediante órdenes de línea de comando.
- djview4: Es un visor para ficheros DJVU, aunque *evince*, el programa visor de PDF por defecto de muchas distribuciones, también trabaja con el formato DJVU (si se han instalado las bibliotecas básicas, las cuales se instalan como dependencia de cualquier aplicación que maneje DJVU. Pero aunque «`evince`» también lea estos ficheros, a mí me gusta más el visor de «`djview`», Tal vez porque tiene atajos de teclado para casi todo, cosa de la que «`evince`» no puede presumir. A mí no me gusta demasiado manejar el ratón. Creo que con él se pierde mucho tiempo.
- minidjvu: Es un codificador DJVU para imágenes bitonales alternativo al "oficial" (que es «`cjb2`», el cual se instala con el paquete «`djvulibre-bin`»). Ofrece varias ventajas frente a «`cjb2`»: tiene capacidad para trabajar simultáneamente con muchos ficheros, puede generar directamente ficheros DJVU multipágina y, sobre todo, alcanza unas mayores ratios de compresión. Muchas veces la diferencia en el tamaño del fichero es bastante significativa.
- ocrodjvu: Este programa permite hacer un OCR a un fichero DJVU de tal modo que, sin dejar de almacenar las imágenes de las páginas, podamos hacer búsquedas de texto. El proceso para lograr esto lo explicaré en 4.1.
- djvusmooth: Es un editor gráfico para ficheros DJVU. Se puede usar para generar el *outline* del libro, tal y como se explica en la sección 4.2.
- pdf2djvu: Como su propio nombre indica, convierte un fichero PDF en un fichero DJVU.
- djview-plugin: un plugin para que firefox sepa entenderse con ficheros DJVU.
- didjvu: otro compresor DJVU alternativo al oficial. No lo he probado.

[8]Transcribo aquí lo que dice la wikipedia. Lo cierto es, no obstante, que aunque el compresor de imágenes bitonales admite la pérdida, por defecto la compresión se hace sin pérdida.

Con estas herramientas instaladas, asumiendo que nuestras páginas estarán en formato PBM o TIF bitonal, podemos proceder de dos maneras para crear el fichero DJVU que almacene nuestro libro:

1º Con «`minidjvu`». Tiene la ventaja de que permite el procesado simultáneo de varios ficheros, genera en un sólo paso un fichero DJVU que almacene todas las páginas y obtiene mejores ratios de compresión. Para lograr todo esto bastará con escribir:

```
minidjvu *.pbm libro.djvu
```

suponiendo que nuestras páginas estén en formato PBM. Lo mismo puede hacerse si están en formato TIF.

> **NOTA:** Aunque es cierto que «`minidjvu`» obtiene una mayor compresión que «`cjb2`», la diferencia es mucho más perceptible si «`minidjvu`» funciona en modo multifichero. Comprimiendo los ficheros de uno en uno, apenas en algunos casos se ve que uno de los dos compresores tenga ventaja sobre el otro.

2º Con el compresor oficial «`cjb2`». Tiene el inconveniente de que sólo puede comprimir las páginas de una en una de tal modo que, tras haberlas comprimido todas, necesitaremos una nueva instrucción para agruparlas en un sólo fichero. Además alcanza peores ratios de compresión, quizás porque, por defecto, la compresión de «`cjb2`» es sin pérdida, aunque se puede indicar, mediante la opción «`-losslevel`» el nivel de pérdida que se está dispuesto a asumir.

> La «pérdida» en la compresión se traduce en una disminución de calidad. Esto es un problema grave en imágenes que posteriormente se vayan a editar de nuevo, pues cada nueva edición mermará más la calidad. Pero para imágenes que no se piensa volver a editar, puede ser asumible cierta dosis de pérdida si a cambio se obtiene una gran reducción en el tamaño del fichero.
>
> No obstante la experiencia me dice que en muchas ocasiones en las que pensaba que cierta imagen no debería volver a ser manipulada, hubo después que volverla a editar. Por lo que yo tiendo a ser en este punto muy precavido y rarísimamente realizo una compresión con pérdida de calidad en la imagen.

Si optamos por esta vía, suponiendo que nuestros ficheros estén en formato TIF, deberemos ejecutar las siguientes órdenes:

```
for i in `ls *.tif`; do
f=${i%.tif}.djvu
cjb2 $i $f
done
djvm -c libro.djvu *.djvu
```

Las líneas 1 a 4 contienen un bucle que irá comprimiendo con «`cjb2`» todos los ficheros TIF del directorio, y la última línea crea un nuevo fichero, llamado «`libro.djvu`» en el que se agrupan todos los ficheros individuales creados en el paso anterior.

**NOTA:** Aunque tanto «`minidjvu`» como «`cjb2`» pueden trabajar indistintamente con ficheros PBM y con ficheros TIF, téngase en cuenta que, en el caso de que algún fichero TIF no almacene una imagen bitonal, tanto uno como otro generarán un error.

# 4. Aditamentos adicionales al libro para facilitar su manejo

En el paso anterior hemos generado un fichero llamado libro.djvu que es ya nuestro libro digitalizado. Podemos, no obstante, añadirle un par de mejoras:

1. Podemos hacer un OCR del fichero que extraiga el texto de las imágenes de las páginas, lo que nos permitirá hacer búsquedas de texto en nuestro flamante libro digitalizado.

2. Podemos diseñarle una especie de índice que se muestre en el visor de DJVU y nos permita, en cualquier momento, saltar a ciertas páginas del libro. A este índice (o conjunto de marcadores) se le denomina, a veces, "*outline*".

## 4.1. Hacer el OCR del libro DJVU

OCR son las siglas de la expresión en inglés *Optical Character Recognition* y consiste en el proceso en virtud del cual, se buscan e identifican, dentro de una imagen, los caracteres de texto que contenga. Su necesidad nace del hecho de que para un ordenador no es lo mismo un fichero de texto que un fichero de imagen. En el fichero de texto, internamente, se almacenan códigos correspondientes a cada uno de los caracteres del mismo. Por eso en un fichero de texto podemos añadir fragmentos de texto en un punto concreto, o cambiar la fuente; el texto propiamente dicho no se ve afectado, sino sólo su apariencia. El fichero de imagen, por el contrario, recoge una sucesión de puntos que conforman una imagen (simplificando muchísimo). El ordenador es capaz de reconstruir la imagen, pero, en condiciones normales, no puede "ver" lo que hay dentro de la imagen. El OCR intenta "mirar" dentro de una imagen para detectar caracteres propios de un texto.

Los sistemas de OCR son, en mayor o menor medida, imprecisos. Para estar seguros de que el OCR de una imagen es totalmente correcto, siempre es preciso contrastar detenidamente el texto localizado y la imagen. Piénsese que, aunque el cerebro humano con mucha facilidad identifica formas dentro de una imagen, para los ordenadores sólo hay agrupaciones de bits y no hay programa que pueda estar seguro de con cuanta perspectiva

hay que mirar la agrupación de bits para identificar la forma de un carácter. Además, algo que nosotros hacemos casi inconscientemente, como es distinguir una letra de una mancha en un papel, al ordenador le cuesta mucho más trabajo. De hecho uno de los test más seguros que se utilizan en Internet para diferenciar a los seres humanos de los robots consiste en identificar el texto que hay en una imagen. Eso es algo que a los humanos apenas nos cuesta esfuerzo, pero las máquinas no hacen demasido bien.

Aún así, aunque el OCR nunca es totalmente seguro, si hay hoy día programas con la suficiente precisión. Aunque durante mucho tiempo este ha sido uno de los aspectos en los que el mundo Windows ha tenido mejores herramientas que el software libre, hoy día, por suerte, eso está empezando a cambiar: existen dos aplicaciones libres que ofrecen unos resultados bastante aceptables: «`tesseract`» y «`cuneiform`». Los dos están disponibles en la mayor parte de las distribuciones de GNU Linux.

Someter un texto digitalizado a un proceso de OCR se puede hacer básicamente de dos maneras:

1. Para simplemente extraer el texto de las imágenes y luego, descartar las imágenes, generando un fichero sólo con el texto. Esto tiene la ventaja de que ese fichero ocuparía muchísimo menos espacio en disco que las imágenes y, además, nos permitiría formatear el texto a nuestro gusto. Pero también tiene inconvenientes:

   - En primer lugar, el proceso de OCR siempre es impreciso. Antes de descartar las imágenes hay que estar seguro de que no hay errores y ello exige un examen tan minucioso que, en ocasiones, se tardaría casi el mismo tiempo tecleando el texto (si somos buenos mecanógrafos).
   - Pero además si descartamos las imágenes, perdemos ciertas características del libro tales como la distribución del texto en una página concreta. Lo que obtendríamos ya no sería "el mismo libro". No podríamos, por ejemplo, hacer una cita de ese libro, porque no sabríamos nunca en qué página del libro original se contiene el texto que estamos citando, salvo que, claro está, en el fichero de texto hayamos anotado toda esa información, lo que implicaría una inversión de tiempo mayor aún de la que ya de por sí es necesaria para hacer un buen OCR que excluya las imágenes.

2. También podemos hacer un OCR en el que, junto con la imagen de cada página, se almacene su transcripción. Eso lleva a ficheros más grandes que los ficheros con sólo imágenes, pero tiene la ventaja de que podremos hacer búsquedas de texto dentro de las imágenes. Además, como las imágenes se conservan, no es tan grave si en algún punto el OCR ha errado: tal vez nuestras búsquedas de texto no sean totalmente precisas, pero, a cambio, podremos ahorrarnos el minucioso y tedioso trabajo de revisar palabra a palabra el resultado del OCR.

Tanto los ficheros PDF como los ficheros DJVU permiten almacenar conjuntamente la imagen de una página y su transcripción en texto. Pero, hasta donde yo se, no hay ninguna

herramienta de software libre que permita hacer un OCR dentro de un fichero PDF. Por el contrario, en ficheros DJVU existe «ocrodjvu»: una aplicación que permite hacer un OCR de un fichero DJVU.

El formato general de «ocrodjvu» es el siguiente:

```
ocrodjvu [opciones] fichero-a-procesar
```

Las principales opciones que «ocrodjvu» admite son las siguientes:

- «-o, --save-bundled»: Esta opción permite indicar el nombre del fichero que se generará. Se tratará de un fichero DJVU que contendrá las mismas páginas que el fichero a procesar pero, además, para cada página, la transcripción del texto que «ocrodjvu» haya localizado en ella.
- «--in-place»: En lugar de generar un nuevo fichero que incorpore el OCR, esta opción hace que se modifique el fichero original, para incorporar dentro de él el OCR.
- «-e, --engine»: Indica el programa que se usará para hacer el OCR. Esto es porque «ocrodjvu» no hace el OCR él mismo, sino que va enviando las páginas al programa de OCR que se le indique. Con la opción «--list-engines» «ocrodjvu» nos informará de los programas disponibles para hacer el OCR con los que él puede comunicarse. Entre estas opciones se encuentra tanto «tesseract» como «cuneiform» (si están instalados en nuestro sistema, claro está), que son, como antes dije, dos de los sistemas de OCR que mejores resultados dan. He leído en ciertas páginas de Internet que «cuneiform» da mejores resultados, pero mis pruebas indican lo contrario: yo me siento más a gusto con «tesseract».

  > En mi última instalación de «ocrodjvu», éste no reconocía la existencia en mi sistema de «tesseract». Para arreglar este fallo tuve que descartar la versión de ocrodjvu que hay en los repositorios de mi distribución (LinuxMint), que coincide —creo— con la de los repositorios Debian, e instalar la última versión, descargada de la página matriz del propio «ocrodjvu».

- «-l, --lang»: Sirve para indicar el idioma en el que está escrito el texto. Esta opción es fundamental para que el OCR tenga una mínima calidad, pues una de las técnicas que usan los programas de OCR cuando tienen dudas respecto de cierto fragmento de la imagen, es consultar un diccionario, de tal modo que si entre los posibles significados del fragmento de texto dudoso hay uno que se encuentre en el diccionario, ese será el elegido. El diccionario a usar, claro está, es distinto para cada idioma. Normalmente cuando instalamos un programa de OCR debemos instalar también un módulo adicional para cada idioma distinto del inglés con el que pretendamos trabajar.

La lista de idiomas disponibles, la obtendremos con la opción "--list-languages". El español se identifica como «spa». Si el texto está en más de un idioma, y como sistema de OCR hemos elegido «tesseract», podemos indicar en la opción «--lang» varios idiomas separados por el carácter "+". Por ejemplo: «--lang spa+eng» indicará que nuestro texto tiene fragmentos en español y otros en inglés.

- «-p, --pages»: Si no queremos procesar todo el fichero, podemos indicar mediante esta opción el rango de páginas a procesar. Para indicar varios rangos hay que separarlos por una coma. Esto es útil si, por ejemplo, en nuestro libro de 300 páginas, sabemos que de la página 15 a la 26 se contienen sólo gráficos o imágenes: Ahorraríamos tiempo de procesado si indicáramos la opción «-p 1-14,27-300».

Por tanto, la siguiente orden

```
ocrodjvu -o salida.djvu --engine tesseract --lang spa libro.djvu
```

procesará nuestro fichero «libro.djvu» en busca de texto en él, asumiendo que dicho texto, de encontrarse, será en español, y generará un nuevo fichero llamado «salida.djvu» con los resultados de tal procesado.

## 4.2. Generación de un *outline* para el fichero DJVU

Los libros digitales, a diferencia de los libros físicos, no pueden simplemente "hojearse", por lo que buscar en ellos un punto concreto es a veces tedioso: hay que recorrer el libro, página a página, hasta llegar al punto buscado; o hacer varias "catas" saltando a ciegas de una página a otra, buscando la página deseada. Si hemos hecho un OCR del libro podemos intentar una búsqueda de texto, pero esto tampoco es seguro: el OCR puede haber fallado, es posible que no recordemos exactamente el texto que buscamos, o este puede usarse en distintos lugares del libro.

Lo ideal para moverse con soltura por un libro digital es tener a la vista siempre, en el visor del libro, un marco en el que tengamos el índice del libro de tal modo que haciendo doble click sobre cualquier apartado, el visor salte directamente a esa página. A este índice "virtual", que realmente no forma parte del libro, pues no constituye ninguna de sus páginas, se le denomina, a veces, «outline» y en el fondo no es sino una colección de marcadores, cada uno de los cuales apunta a cierta página del libro.

En un fichero DJVU hay dos herramientas que permiten crear un outline: «djvsmooth» y «djvused». La primera es una herramienta gráfica y la segunda una instrucción de línea de comandos. Como suele ocurrir, la primera es más sencilla de usar, pero mucho menos potente.

> **NOTA:** Tanto «djvusmooth» como «djvused» son herramientas de uso general que no sirven sólo para incluir *outlines* en un fichero DJVU. El primero es un editor gráfico de

ficheros DJVU que permite examinar y editar los metadatos del documento y sus capas ocultas, además del posible *outline*. Y «djvused» es un muy potente editor de línea de comandos para documentos DJVU.

### 4.2.1. Con djvusmooth

Generar un *outline* con «djvusmooth» es bastante sencillo, simplemente hay que abrir el documento e ir pasando por sus páginas. Si en una página pulsamos CTRL-B, en el marco izquierdo de la ventana del programa se generará un marcador que apunta a dicha página. Haciendo click sobre él podemos cambiar su título.

Los marcadores, por otra parte, son jerárquicos: Los hay de primer nivel, de segundo nivel, etc. En principio cada marcador nuevo será de primer nivel, pero podemos hacer que sea de segundo o ulterior nivel, simplemente arrastrándolo hacia el marcador de nivel superior del que queremos que dependa.

### 4.2.2. Con djvused

Como antes se dijo «djvused» es un potente editor de línea de comandos para documentos DJVU. Su formato general es el siguiente:

```
djvused [opciones] ficherodjvu
```

Las opciones principales son «-e» y «-s». La primera va seguida de un parámetro entrecomillado donde se indica la acción a realizar. La segunda hace que, si la orden previa ha producido algún cambio en el fichero a editar, este cambio sea grabado en el disco.

Para incluir un *outline* en nuestro fichero DJVU con «djvused» necesitamos, en primer lugar, generar un fichero de texto que contenga el outline. Si, por ejemplo, este fichero se denomina «bm.txt» la siguiente orden incluirá dicho *outline* en nuestro «libro.djvu».

```
djvused -e 'set-outline bm.txt' -s libro.djvu
```

El fichero que contenga el outline usa lo que podríamos llamar "sintaxis de paréntesis anidados". El fichero empieza con la línea

```
(bookmarks
```

y termina cerrando el paréntesis que se abrió en la primera línea, de tal manera que todo su contenido estará encerrado en ese primer paréntesis. Cualquier otro contenido se indica también entre paréntesis. Cada marcador tiene la siguiente sintaxis:

```
("Nombre de marcador" "#Nº de página")
```

Y si un marcador, a su vez, tiene submarcadores, estos se incluyen dentro del paréntesis del marcador al que pertenecen del siguiente modo:

```
(bookmarks
  ("Primer marcador" "#Nº de página")
  ("Segundo marcador" "#Nº de página"
    ("Submarcador del primer marcador" "#Nº de página")
  )
)
```

Mediante el sangrado del texto puede verse la jerarquía de los distintos marcadores. Pero, desde el punto de vista de la sintaxis, el que el texto esté o no sangrado es indiferente: el programa mide la jerarquía atendiendo exclusivamente al nivel de profundidad en los paréntesis en que se encuentre cada marcador. No obstante escribir un texto correctamente sangrado hace que, a los humanos, nos sea más sencillo ver los distintos niveles y relaciones entre las entradas del *outline*.

> Una vez más un buen editor de texto, como «vim» es una gran ayuda para escribir este tipo de ficheros. Porque se trata de documentos en los que es importante que los paréntesis estén totalmente balanceados, es decir: que todo paréntesis abierto, sea cerrado en algún punto posterior del documento. Y en «vim» el comando « %» nos ayuda a comprobar el balanceo de paréntesis: Puesto el cursor sobre un signo de paréntesis (de apertura o de cierre) si se pulsa « %» se saltará al punto del documento donde se encuentra la pareja exacta de ese paréntesis: desde el signo de apertura se salta al de cierre y viceversa, dando igual cuantos paréntesis anidados haya por el camino. Por lo tanto si en «vim» colocamos el cursor sobre el primer carácter del documento (que como hemos visto es un signo de apertura de paréntesis) y pulsamos « %» deberá saltarse, si los paréntesis están correctamente balanceados, al último carácter del documento. Puesto el cursor sobre este último carácter, una nueva pulsación de « %» deberá saltar al principio. Si así ocurre es que no hay ningún error en el balanceado de los paréntesis.

La ventaja que ofrece «djvused» sobre «djvusmooth» está en que si nuestro libro tiene un índice, y hemos hecho OCR del libro, podemos rescatar las páginas del índice y ajustarlas al formato de los ficheros de bookmarks. Para un índice complejo, si se dispone de un buen editor de texto como «vim» y se manejan bien las expresiones regulares, esto sería relativamente sencillo, y mucho más cómodo que escribir a mano todas las entradas del índice.

Para extraer el texto correspondiente al OCR de una página de nuestro fichero DJVU se puede usar el comando «djvutxt» o el mismo «djvused» con el comando «print-pure-txt». Por ejemplo, si el índice de nuestro libro estuviera en la página 913 del mismo, el siguiente comando generaría un fichero llamado indice.txt con el resultado del OCR de dicha página:

```
djvused -e 'select 913; print-pure-txt' libro.djvu > indice.txt
```

A partir de ahí tendríamos que editar el índice para darle la sintaxis adecuada y para ajustar la numeración de páginas a la que sea correcta en el fichero DJVU. Tras ello con «`djvused`» podríamos incorporar dicho outline.

***

And... That's All Folks.

Have a nice day