Informe de Laboratorio #2: Procesamiento de Imágenes BMP en C

Santiago Alarcón V. y Richard Gómez P.
Universidad Sergio Arboleda
Ingeniería Electrónica

Agosto 19 de 2025

Resumen

Este informe presenta los resultados del Laboratorio #2 de Sistemas Embebidos, donde se desarrolló un programa en C para procesar imágenes BMP, cumpliendo con los requerimientos de cargar un archivo BMP, almacenar sus píxeles en una matriz, implementar un menú con opciones de conversión a escala de grises y convolución con un kernel 3x3, e investigar kernels de detección de bordes. Se incluye un diagrama de flujo que describe el funcionamiento del programa, junto con los resultados de las pruebas y un análisis de los kernels investigados.

1. Introducción

El procesamiento de imágenes es fundamental en sistemas embebidos, con aplicaciones en visión por computadora, reconocimiento de patrones y análisis de datos visuales. El Laboratorio #2, descrito en [1], requirió el desarrollo de un programa en C para manipular imágenes BMP, cumpliendo con los siguientes objetivos:

1. Cargar un archivo BMP en memoria.
2. Copiar los datos de los píxeles a una matriz en memoria.
3. Implementar un menú interactivo con dos opciones: conversión a escala de grises y convolución con un kernel 3x3.
4. Investigar kernels de detección de bordes y probar su funcionalidad en las distintas combinaciones.

El formato BMP, descrito en [2], es un estándar ampliamente utilizado por su estructura simple y falta de compresión con pérdida. Este informe detalla la metodología empleada, el funcionamiento del programa, los resultados obtenidos y un análisis de los kernels investigados.

2. Metodología

El programa se desarrolló en lenguaje C, siguiendo las especificaciones del laboratorio. A continuación, se describe la metodología para cada objetivo partiendo del diagrama qque se puede ver en la Figura 4.

2.1. Carga de BMP y Matriz de Píxeles

El programa lee un archivo BMP en modo binario, extrayendo el encabezado del archivo (14 bytes) y el encabezado de información (40 bytes). Verifica que el archivo sea un BMP válido (`bfType = 0x4D42`) y que cumpla con los requisitos de profundidad de color (inicialmente 24 bits). Los píxeles se almacenan en una matriz dinámica bidimensional de estructuras que representan los valores RGB de cada píxel, respetando el padding por fila requerido por el formato BMP.

2.2. Menú y Procesamiento

Se implementó un menú interactivo que permite al usuario seleccionar entre tres opciones: conversión a escala de grises, convolución con un kernel 3x3, y salir. La conversión a escala de grises utiliza la fórmula estándar ITU-R 601-2 para calcular la luminancia. La convolución solicita al usuario los valores de un kernel 3x3 y un divisor, procesando la imagen píxel por píxel. Los resultados se guardan en nuevos archivos BMP, manteniendo el formato original.

2.3. Kernels de Detección de Bordes

Se investigaron tres kernels de detección de bordes, como se indica en el punto 4 del laboratorio y se investigo en [3]:

- **Sobel Horizontal**:
$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \text{Divisor: 1 o 8}$$
Detecta bordes horizontales al enfatizar gradientes verticales.

- **Sobel Vertical**:
$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}, \text{Divisor: 1 o 8}$$
Detecta bordes verticales al enfatizar gradientes horizontales.

- **Laplaciano**:
$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \text{Divisor: 1}$$
Detecta bordes en todas las direcciones, resaltando cambios bruscos de intensidad.

3. Descripción del Programa

El programa principal cumple con los requerimientos del laboratorio. A continuación, se detalla su estructura y funcionamiento.

3.1. Estructuras de Datos

Se definieron estructuras para manejar el formato BMP y los píxeles:

- **Encabezado del archivo**: Contiene el identificador (`BM`), tamaño del archivo y offset a los datos de píxeles.
- **Encabezado de información**: Incluye dimensiones (ancho y alto), profundidad de color, y otros parámetros.
- **Píxel**: Estructura para almacenar valores RGB de cada píxel durante el procesamiento.
- **Paleta**: Usada para BMPs de 8 bits, mapeando índices a colores RGB.

3.2. Funcionalidad Principal

El programa sigue estos pasos:

1. **Carga del archivo**: Lee el archivo BMP, verifica su validez y profundidad de color, y carga los píxeles en una matriz.
2. **Almacenamiento de píxeles**: Crea una matriz bidimensional para almacenar los valores RGB, manejando el padding y las caracteristicas del archivo.
3. **Menú interactivo**: Permite seleccionar conversión a escala de grises (usando la fórmula de luminancia), convolución (con un kernel 3x3 ingresado por el usuario), o salir.
4. **Procesamiento**: Aplica las operaciones seleccionadas y guarda los resultados en un nuevo archivo BMP que se le pone nombre por el usuario.

3.3. Gestión de Memoria

Se utiliza memoria dinámica para la matriz de píxeles. Toda la memoria se libera al finalizar para evitar fugas.

4. Resultados

Se generaron imágenes BMP de prueba (100x100 píxeles) con un fondo blanco y una franja de color para evaluar el programa. El archivo BMP que se implemento como el original es el que se puede ver en la Figura 1. Las pruebas incluyeron:

4.1. Conversión a Escala de Grises

La conversión transformó las imágenes a tonos de gris, preservando la profundidad de color original. En este punto del diagrama se implemento con la imagen original siendo como el resultado la Figura 2 mostrando que la imagen original cambíe sus tonalidades a escala de grises.

4.2. Convolución

Se aplicaron los kernels investigados a las imágenes de prueba. Por ejemplo, usando el kernel Sobel horizontal:

```
Ingresa 3x3 valores kernel (fila por fila):
kernel[0][0]: -1
kernel[0][1]: 0
kernel[0][2]: 1
kernel[1][0]: -2
kernel[1][1]: 0
kernel[1][2]: 2
kernel[2][0]: -1
kernel[2][1]: 0
kernel[2][2]: 1
Ingrese divisor para kernel: 1
```

En este parte del diagrama de flujo; que se muestra en la Figura 4; se implemneta esta convinación de kernels para resaltar los bordes horizontales como se muestra en la Figura 3 resaltó los bordes horizontales de la franja de color, sin embargo también resalto de forma minima los bordes verticales.

5. Discusión

El programa cumple con los requerimientos del laboratorio, procesando imágenes BMP de manera efectiva. La conversión a escala de grises es precisa, y la convolución resalta bordes según el kernel utilizado. Sin embargo, no se llegaron a suplir ciertos requisitos que ayudaban al desarrollo del laboratorio, empezando por el lado del menu donde aquí es limitado a que se siga trabajando con el mismo archivo BMP y en caso de cambiarlo se requeria cerrar el programa, además de eso, también se requería mayor conocimiento del codigo en C y los tipos y formas de la convolución.

6. Conclusiones

El laboratorio permitió desarrollar habilidades en procesamiento de imágenes en C, demostrando la capacidad de manipular archivos BMP y aplicar técnicas como convolución y conversión a escala de grises. Para mejora del programa hacer los ajustes necesarios para que sea más sencillo de usar.

Referencias

[1] Universidad Sergio Arboleda, Documento del Laboratorio #2: Procesamiento de Imágenes BMP, Sistemas Embebidos, 2025.

[2] Equipo de contenido de Movavi, https://www.movavi.com/es/learning-portal/bmp-file.html, consultado el 19 de agosto de 2025, creado el 3 de agosto del 2025.

[3] GeekesForGeeks team, https://www.geeksforgeeks.org/deep-learning/types-of-convolution-kernels, consultado el 17 de agosto de 2025, modificado el 23 de julio de 2025.

7. Anexos

Figura 1: Archivo bmp de prueba o de ejemplo.

Figura 2: Archivo de prueba convertido a escala de grises.

Figura 3: Archivo convertido a escala de grises convolucionado de manera Sobel horizontal.

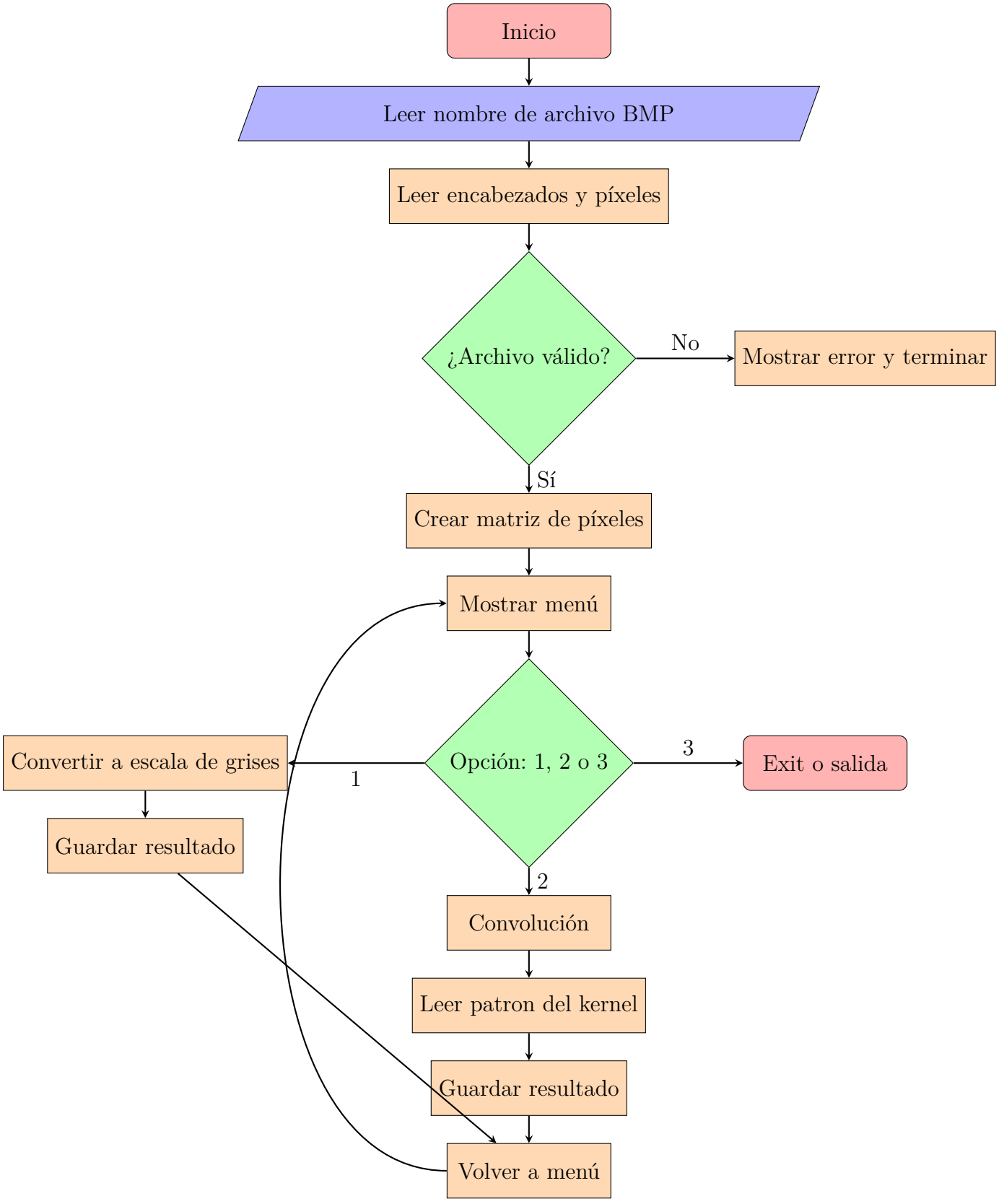

Figura 4: Diagrama de flujo del programa del laboratorio 2.