Tercera Práctica
Procesamiento de Imagenes
Curso 2025-2026

Sergiy Khudoley

3 de febrero de 2026

Índice

1. Introducción **2**

2. Preparando el entorno **2**

2.1. Numpy 3

2.2. Pillow (PIL) 3

2.3. Matplot 3

3. Implementación del detector de bordes **4**

4. Aplicación del detector de bordes **5**

5. Discusión de resultados **5**

6. Conclusiones **6**

Figura 1: Imágen usada para la práctica

1. Introducción

En esta práctica se implementa un detector de bordes basado en gradiente para procesar imágenes digitales. El objetivo es identificar las zonas de cambio brusco de intensidad, que corresponden a los bordes de los objetos presentes en la imagen.

Se carga una imagen en escala de grises, se calculan los gradientes horizontal y vertical en cada píxel, y a partir de ellos se obtiene la magnitud del gradiente. Posteriormente, aplicando distintos valores de umbral, se generan imágenes binarias que muestran los bordes detectados.

2. Preparando el entorno

Para ejecutar el programa de forma correcta y evitar conflictos con otras librerías del sistema, se recomienda utilizar un entorno virtual de Python (venv).

Primero se crea el entorno virtual en el directorio del proyecto ejecutando el siguiente comando:

```
python3 -m venv venv
```

Una vez creado, se activa el entorno virtual:

```
source venv/bin/activate
```

Al activar el entorno virtual, todas las librerías que se instalen quedarán aisladas dentro del proyecto.

A continuación, se instalan las dependencias necesarias usando pip:

```
pip install numpy pillow matplotlib
```

Con esto, el entorno queda preparado para ejecutar el script de procesamiento de imágenes.

2.1. Numpy

NumPy proporciona estructuras de datos eficientes como los arrays multidimensionales y funciones matemáticas optimizadas.

En este proyecto se utiliza NumPy para convertir la imagen en una matriz numérica, almacenar los valores de intensidad de cada píxel y calcular los gradientes horizontal y vertical.

También se emplea para obtener la magnitud del gradiente mediante operaciones matemáticas sobre matrices.

2.2. Pillow (PIL)

Pillow permite abrir, convertir y manipular imágenes de forma sencilla.

En este trabajo se usa Pillow para cargar la imagen original desde un archivo y convertirla a escala de grises. Posteriormente, la imagen se transforma en una matriz NumPy para poder realizar los cálculos del gradiente.

2.3. Matplot

Matplotlib permite crear gráficos e imágenes de forma sencilla.

En este proyecto se utiliza Matplotlib para mostrar la imagen original, la magnitud del gradiente y los resultados del detector de bordes con distintos umbrales. También se emplea para guardar las figuras generadas en archivos de imagen.

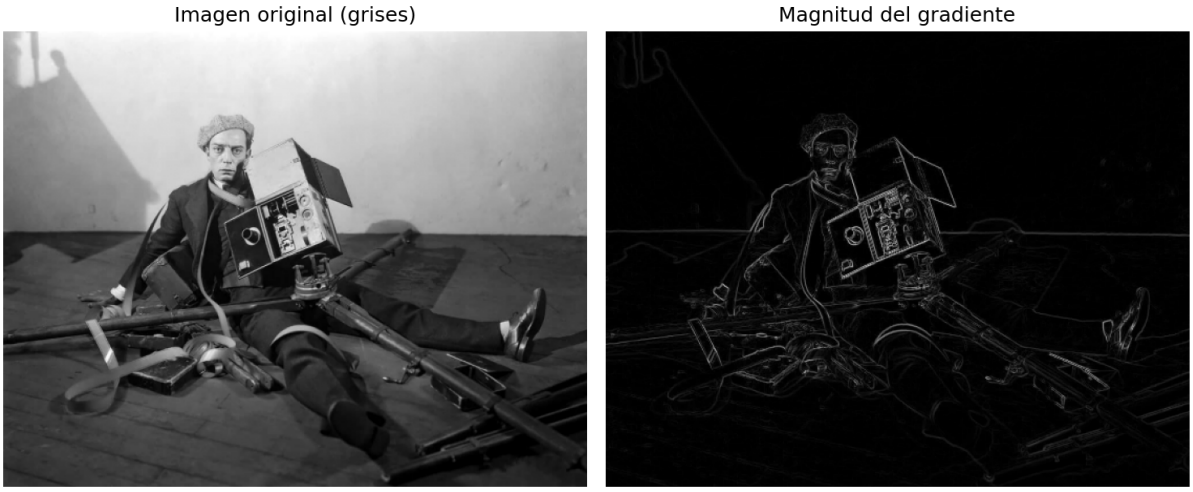

Figura 2: Gradiente generado con el algoritmo

3. Implementación del detector de bordes

En esta práctica, el detector de bordes se implementa mediante el cálculo del gradiente de intensidad de la imagen en las direcciones horizontal y vertical. La idea principal es que los bordes de la imagen corresponden a zonas donde la intensidad cambia bruscamente, es decir, donde el gradiente es grande.

Primero, se carga la imagen en escala de grises utilizando la librería Pillow y se convierte a un array de NumPy para poder trabajar con los valores de los píxeles de forma numérica:

```
im = Image.open('cameraman.png').convert('L')
I = np.array(im, dtype=np.float32)
```

Se definen matrices para almacenar los gradientes horizontal (Gx) y vertical (Gy), inicializadas a cero:

```
rows, cols = I.shape
Gx = np.zeros((rows, cols), dtype=np.float32)
Gy = np.zeros((rows, cols), dtype=np.float32)
```

El gradiente horizontal Gx se calcula como la diferencia entre un píxel y su vecino izquierdo, recorriendo cada fila de la imagen:

```
for i in range(rows):
    for j in range(1, cols):
        Gx[i, j] = I[i, j] - I[i, j-1]
```

El gradiente vertical Gy se calcula como la diferencia entre un píxel y su vecino inferior, recorriendo cada columna:

```
for i in range(rows-1):
    for j in range(cols):
        Gy[i, j] = I[i, j] - I[i+1, j]
```

Una vez obtenidos los gradientes horizontal y vertical, se calcula la magnitud del gradiente en cada píxel combinando ambos mediante la fórmula de la norma Euclidiana:

```
grad_mag = np.sqrt(Gx2 + Gy2)
```

La magnitud del gradiente indica la intensidad del borde en cada píxel. Para obtener una imagen binaria de bordes, se aplica un umbral sobre la magnitud, de forma que los píxeles cuya magnitud supere el umbral se consideran bordes.

Finalmente, se pueden generar y visualizar varias imágenes de bordes usando distintos valores de umbral para analizar subjetivamente cómo cambia la detección de bordes según este parámetro.

4. Aplicación del detector de bordes

Una vez implementado el detector de bordes, se aplicó sobre la imagen proporcionada para identificar las zonas de cambio brusco de intensidad. El proceso consiste en calcular la magnitud del gradiente para cada píxel, tal como se describió en la sección de implementación, y posteriormente aplicar distintos valores de umbral para generar imágenes binarias de bordes.

Para ilustrar el efecto del umbral, se seleccionaron varios valores representativos: 5, 10, 20, 30, 40, 50, 80 y 120. Para cada valor, los píxeles cuya magnitud del gradiente superaba el umbral se marcaron como bordes (valor 255), mientras que los demás se establecieron a 0.

El resultado es un conjunto de imágenes binarias que muestran cómo la elección del umbral afecta la detección de bordes:

1. Umbrales bajos (por ejemplo, 5 o 10) generan muchas líneas de borde, incluyendo ruido y detalles finos.
2. Umbrales intermedios (20–50) resaltan los bordes más significativos, manteniendo las estructuras principales de la imagen.
3. Umbrales altos (80–120) detectan únicamente los bordes más marcados, perdiendo detalles finos.

Este análisis permite observar de forma subjetiva cómo el valor del umbral influye en la detección de bordes y resalta la necesidad de elegir un valor adecuado según el nivel de detalle que se desee capturar.

Para cada umbral, las imágenes resultantes se guardaron y se mostraron usando Matplotlib, facilitando la comparación visual y la discusión de los efectos de distintos valores de umbral sobre la imagen final.

5. Discusión de resultados

La aplicación del detector de bordes mostró claramente cómo la magnitud del gradiente y la elección del umbral afectan la detección de los bordes en la imagen.

Al analizar las imágenes generadas con distintos valores de umbral, se observan los siguientes efectos:

1. **Umbrales bajos (5–10):** Se detectan prácticamente todos los cambios de intensidad, incluyendo detalles finos y ruido. La imagen de bordes resultante es muy densa y puede resultar confusa, ya que aparecen muchas líneas que no corresponden a bordes importantes.

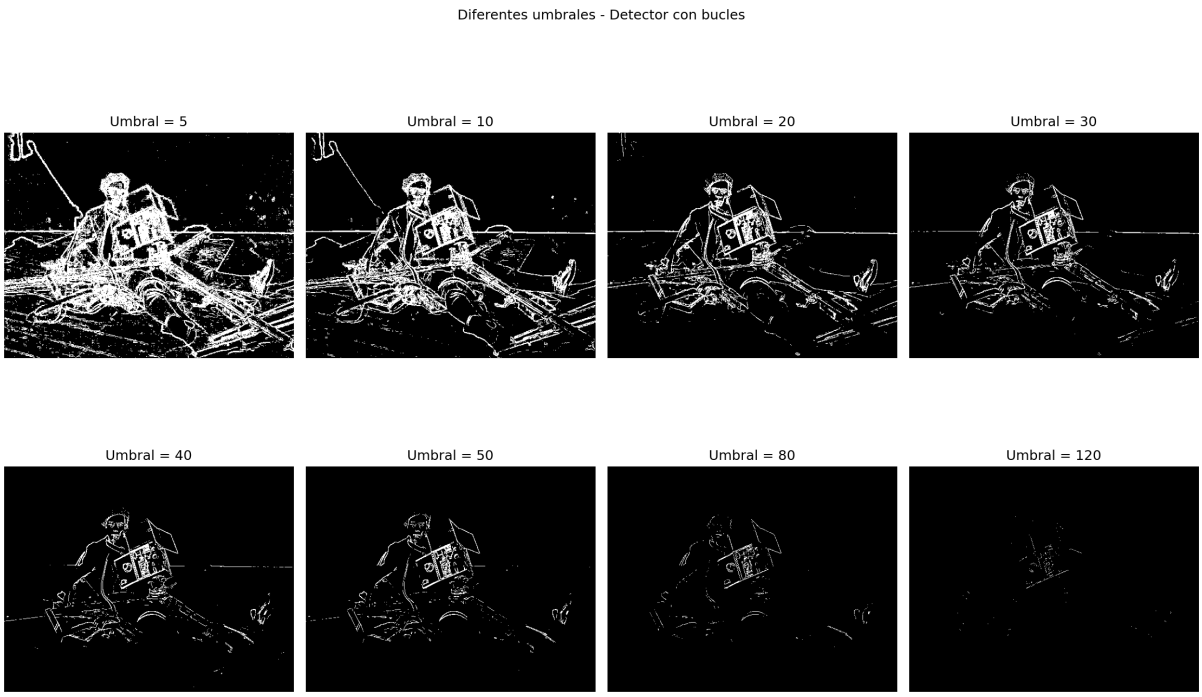

Figura 3: Aplicación del detector con distintos umbrales

2. **Umbrales intermedios (20–50):** Se destacan los bordes principales de la imagen, como los contornos de objetos y estructuras relevantes. Este rango proporciona un equilibrio adecuado entre detalle y claridad, permitiendo identificar los bordes significativos sin demasiada información de ruido.

3. **Umbrales altos (80–120):** Solo se detectan los bordes más marcados. Los detalles finos desaparecen y la imagen se simplifica considerablemente. Esto puede ser útil cuando se desea resaltar únicamente los contornos más evidentes de la escena.

En general, el resultado del detector de bordes depende de manera subjetiva del umbral elegido. No existe un valor "correcto" universal; la selección depende de la cantidad de detalle que se quiera resaltar y del nivel de ruido presente en la imagen.

La observación comparativa de las imágenes con distintos umbrales permite comprender cómo el gradiente de intensidad refleja la presencia de bordes y cómo pequeños cambios en el umbral pueden modificar significativamente la percepción de los contornos.

6. Conclusiones

En conclusión, este experimento demuestra la importancia de ajustar el umbral de manera adecuada para cada imagen y la utilidad del gradiente como herramienta básica para la detección de bordes en procesamiento de imágenes.