



INTERNET DE LAS COSAS

PRACTICA # 3

MANIPULACIÓN DE DATOS EN

DISPOSITIVOS DE IOT

**BY DR. FRANCISCO JAVIER
ALVARADO RODRIGUEZ**

18/11/24

ACTIVIDAD 10

**ISAAC
MENCHACA**

3320488

MCC

INTRODUCCION



En el contexto del Internet de las Cosas (IoT), la capacidad de capturar y procesar imágenes en tiempo real ofrece grandes posibilidades para aplicaciones como la vigilancia, el monitoreo ambiental, y el análisis visual. En esta práctica, se trabajará con una Raspberry Pi Zero W y un módulo de cámara para capturar imágenes y aplicar técnicas de procesamiento mediante filtros pasa altas y pasa bajas, los cuales son útiles para resaltar o suavizar los detalles en una imagen.

El filtro pasa bajas atenúa las frecuencias altas de la imagen, lo que resulta en una imagen suavizada que reduce el ruido y elimina detalles finos. En contraste, el filtro pasa altas resalta los bordes y los detalles de alta frecuencia, lo que permite detectar contornos y elementos prominentes en la imagen.

El propósito de esta práctica es familiarizarse con los conceptos básicos de procesamiento de imágenes en sistemas IoT, permitiendo explorar aplicaciones más complejas en sistemas de visión por computadora.

DESARROLLO

1. Preparación del entorno

Antes de iniciar la práctica, fue necesario preparar el entorno de trabajo. Se realizaron las siguientes actividades:

- **Instalación del sistema operativo:** Se descargó e instaló Raspberry Pi OS Lite en una tarjeta microSD utilizando una herramienta como Raspberry Pi Imager. Este sistema operativo fue elegido por ser ligero y adecuado para proyectos de IoT.
- **Configuración de SSH:** Para facilitar la comunicación con la Raspberry Pi Zero W, se habilitó el acceso remoto mediante SSH. Esto se realizó creando un archivo vacío llamado ssh en la partición de arranque de la tarjeta microSD.
- **Conexión a la red Wi-Fi:** Se configuró la conexión Wi-Fi de la Raspberry Pi creando un archivo wpa_supplicant.conf con los datos de la red, lo cual permitió que el dispositivo se conectara automáticamente al encenderse.
- **Conexión de la cámara:** El módulo de cámara se conectó al puerto CSI de la Raspberry Pi, asegurándose de que el cable de cinta estuviera orientado correctamente.
- **Habilitación de la cámara:** Desde la terminal SSH, se usó el comando sudo raspi-config para habilitar el módulo de la cámara en la sección Interfacing Options.

2. Instalación de las bibliotecas necesarias

Se instalaron las bibliotecas de Python necesarias para capturar y procesar imágenes:

```
sudo apt-get update  
sudo apt-get install python3-opencv
```

OpenCV es una biblioteca de procesamiento de imágenes que permite aplicar filtros y realizar análisis avanzado de las imágenes capturadas.

3. Aplicación de Filtros Pasa Altas y Pasa Bajas

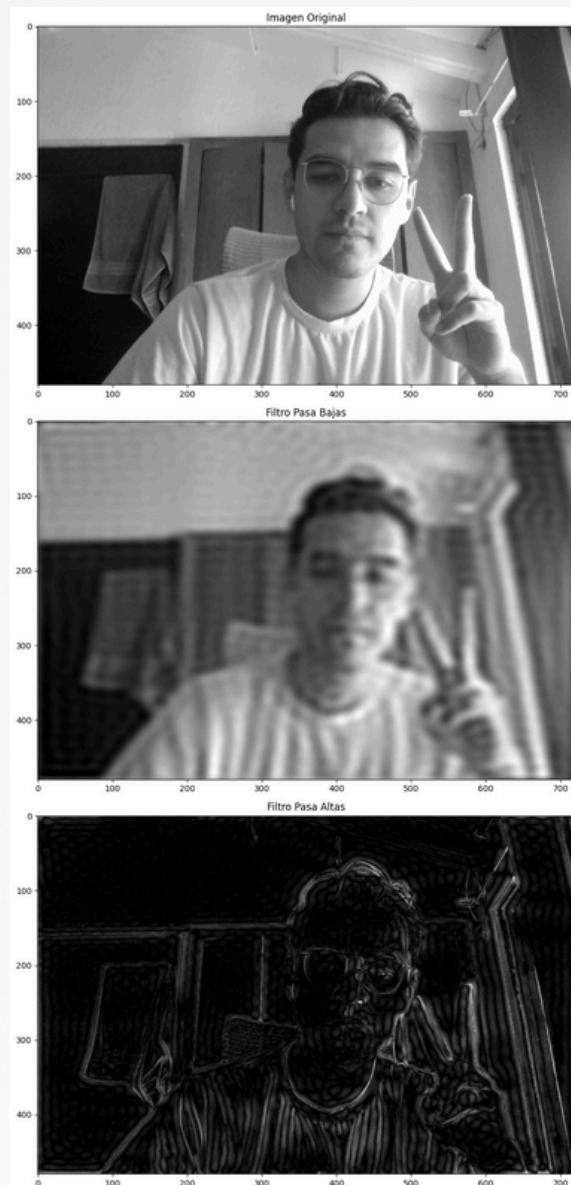
Se creó un script en Python para cargar la imagen y aplicar los filtros utilizando OpenCV.

Anexos

4. Resultados

Después de aplicar los filtros pasa bajas y pasa altas, se obtuvo lo siguiente:

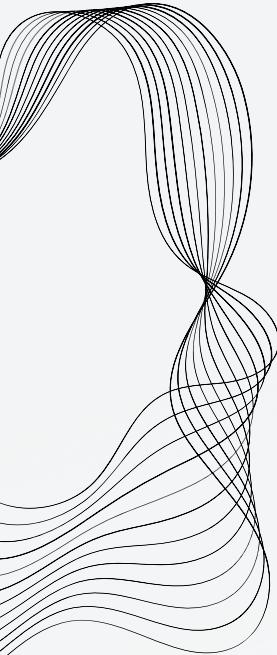
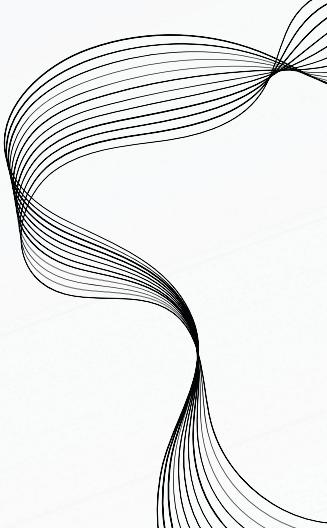
- La imagen con el filtro pasa bajas presentaba una apariencia suavizada, con los detalles finos y el ruido notablemente reducidos. Esto es útil en aplicaciones donde se requiere una imagen más limpia, sin el impacto de ruido o texturas complejas.
- La imagen con el filtro pasa altas resaltó los bordes y contornos, lo que ayuda a identificar elementos destacados en la imagen. Esta técnica es común en la detección de objetos y análisis de bordes, especialmente útil en sistemas de vigilancia y monitoreo.



CONCLUSIONES

En esta práctica se logró capturar una imagen con la cámara de la Raspberry Pi Zero W y aplicar filtros pasa bajas y pasa altas para su procesamiento. Estos filtros permitieron suavizar la imagen y resaltar detalles específicos, respectivamente, lo que ilustra las posibilidades del procesamiento de imágenes en aplicaciones IoT. La implementación de estos filtros constituye un paso inicial hacia el análisis de imágenes en tiempo real, un campo con aplicaciones prácticas como la detección de objetos, el reconocimiento de patrones y la vigilancia.

REFERENCIAS

- 
- 
- [1] A. Monk, Programming the Raspberry Pi: Getting Started with Python, 2nd ed. New York, NY: McGraw-Hill, 2015.
 - [2] S. Monk, The Internet of Things: Do-It-Yourself Projects with Arduino, Raspberry Pi, and BeagleBone Black, New York, NY: McGraw-Hill, 2015.
 - [3] Upton, E., and Halcree, G., Raspberry Pi User Guide, 4th ed. New York, NY: Wiley, 2016.
 - [4] G. Bradski and A. Kaehler, Learning OpenCV: Computer Vision with the OpenCV Library, 1st ed. Sebastopol, CA: O'Reilly Media, 2008.

ANEXOS

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Cargar la imagen en escala de grises
imagen = cv2.imread('ruta_de_tu_imagen.jpg',
cv2.IMREAD_GRAYSCALE)

# Aplicar Transformada de Fourier a la imagen
dft = cv2.dft(np.float32(imagen),
flags=cv2.DFT_COMPLEX_OUTPUT)
dft_shift = np.fft.fftshift(dft)

# Tamaño de la imagen
filas, columnas = imagen.shape
centro_filas, centro_columnas = filas // 2, columnas
// 2

# Crear máscara para filtro pasa bajas
radio_pasa_bajas = 30 # Cambia este valor según tus
necesidades
mascara_pasa_bajas = np.zeros((filas, columnas, 2),
np.uint8)
cv2.circle(mascara_pasa_bajas, (centro_columnas,
centro_filas), radio_pasa_bajas, (1, 1), -1)

# Aplicar filtro pasa bajas
dft_shift_pasa_bajas = dft_shift * mascara_pasa_bajas
dft_inv_shift_pasa_bajas =
np.fft.ifftshift(dft_shift_pasa_bajas)
imagen_filtro_pasa_bajas =
cv2.idft(dft_inv_shift_pasa_bajas)
imagen_filtro_pasa_bajas =
cv2.magnitude(imagen_filtro_pasa_bajas[:, :, 0],
imagen_filtro_pasa_bajas[:, :, 1])
```

```
# Crear máscara para filtro pasa altas
mascara_pasa_altas = np.ones((filas, columnas, 2),
np.uint8)
cv2.circle(mascara_pasa_altas, (centro_columnas,
centro_filas), radio_pasa_bajas, (0, 0), -1)

# Aplicar filtro pasa altas
dft_shift_pasa_altas = dft_shift * mascara_pasa_altas
dft_inv_shift_pasa_altas =
np.fft.ifftshift(dft_shift_pasa_altas)
imagen_filtro_pasa_altas =
cv2.idft(dft_inv_shift_pasa_altas)
imagen_filtro_pasa_altas =
cv2.magnitude(imagen_filtro_pasa_altas[:, :, 0],
imagen_filtro_pasa_altas[:, :, 1])

# Mostrar resultados
plt.figure(figsize=(12, 6))

plt.subplot(1, 3, 1)
plt.title('Imagen Original')
plt.imshow(imagen, cmap='gray')

plt.subplot(1, 3, 2)
plt.title('Filtro Pasa Bajas')
plt.imshow(imagen_filtro_pasa_bajas, cmap='gray')

plt.subplot(1, 3, 3)
plt.title('Filtro Pasa Altas')
plt.imshow(imagen_filtro_pasa_altas, cmap='gray')

plt.tight_layout()
plt.show()
```