# Install required libraries if not already installed

!pip install opencv-python pillow matplotlib

# Import necessary libraries

import cv2

import numpy as np

from google.colab import files

from PIL import Image

import matplotlib.pyplot as plt

# Step 1: Upload an image

uploaded = files.upload()

# Open the image using PIL and convert it to a format OpenCV understands

image\_path = list(uploaded.keys())[0]

image = Image.open(image\_path)

image\_cv = np.array(image)

# Convert from RGB to BGR (since OpenCV uses BGR by default)

image\_cv = cv2.cvtColor(image\_cv, cv2.COLOR\_RGB2BGR)

# Step 2: Convert the image to grayscale (Sobel works on grayscale images)

gray\_image = cv2.cvtColor(image\_cv, cv2.COLOR\_BGR2GRAY)

# Step 3: Apply Sobel operator along the X-axis

# Sobel kernel along the X axis (dx=1, dy=0) computes the gradient in the X-direction.

sobel\_x = cv2.Sobel(gray\_image, cv2.CV\_64F, 1, 0, ksize=3)

# Step 4: Convert the result to a format suitable for display (uint8)

sobel\_x\_abs = np.absolute(sobel\_x) # Take absolute values

sobel\_x\_uint8 = np.uint8(sobel\_x\_abs) # Convert to uint8 for display

# Step 5: Display the original and Sobel edge-detected images along the X-axis

plt.figure(figsize=(10, 5))

# Display original image

plt.subplot(1, 2, 1)

plt.imshow(image)

plt.title("Original Image")

plt.axis('off')

# Display Sobel edge-detected image along the X-axis

plt.subplot(1, 2, 2)

plt.imshow(sobel\_x\_uint8, cmap='gray')

plt.title("Edge Detection (Sobel X-axis)")

plt.axis('off')

plt.show()

# Optional: Save and download the Sobel edge-detected image

cv2.imwrite("sobel\_x\_edges.jpg", sobel\_x\_uint8)

files.download("sobel\_x\_edges.jpg")

