# 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 (Canny works on grayscale images)

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

# Step 3: Apply Gaussian Blur to reduce noise and improve edge detection

blurred\_image = cv2.GaussianBlur(gray\_image, (5, 5), 0)

# Step 4: Perform Canny Edge Detection

# The two threshold values can be adjusted to fine-tune the edge detection.

low\_threshold = 100

high\_threshold = 200

edges = cv2.Canny(blurred\_image, low\_threshold, high\_threshold)

# Step 5: Display the original and edge-detected images

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

# Display original image

plt.subplot(1, 2, 1)

plt.imshow(image)

plt.title("Original Image")

plt.axis('off')

# Display edge-detected image

plt.subplot(1, 2, 2)

plt.imshow(edges, cmap='gray')

plt.title("Edge Detection (Canny)")

plt.axis('off')

plt.show()

# Optional: Save and download the edge-detected image

cv2.imwrite("edges.jpg", edges)

files.download("edges.jpg")

