Lab Assignment 3

Problem Statement

Write a program to implement the **Canny edge detection algorithm** using Python.

Description and Hints

Objective

The goal of this assignment is to understand and implement the **Canny edge detection algorithm** using Python. The Canny algorithm involves multiple steps, including noise reduction, gradient calculation, non-maximum suppression, and thresholding with hysteresis. You will apply this algorithm to an input image and visualize the output.

Guidelines for Implementation

1. **Libraries to Use**
   - Use **OpenCV** for image processing (loading, filtering, and edge detection).
   - Use **NumPy** for any required numerical computations.
   - Use **Matplotlib** or OpenCV's display utilities to visualize the original and edge-detected images.
2. **Steps to Solve**
   A. **Load an Image**
   Use OpenCV to read an image in grayscale. This simplifies edge detection as it operates on intensity values.

   B. **Reduce Noise**
   Apply a **Gaussian Blur** to the image to reduce noise. This helps prevent false edge detection.

   C. **Apply Canny Edge Detection**
   Use the OpenCV `Canny` function to apply the edge detection algorithm. Experiment with different threshold values to see their effect on the output.

   D. **Visualize Results**
   Plot the original and edge-detected images side by side using either Matplotlib (`plt.imshow`) or OpenCV's `cv2.imshow` / `cv2_imshow`.

Hints for Functions to Use

1. **Image Reading and Displaying**:
   - Use `cv2.imread` to load the image.

- Use `cv2.imshow` or `cv2_imshow` (if using Google Colab) or `plt.imshow` to display images.

2. **Noise Reduction**:
   - Use `cv2.GaussianBlur` for applying Gaussian blur. Provide appropriate kernel size (e.g., `(5, 5)`) and standard deviation (`sigmaX`).
3. **Edge Detection**:
   - Use `cv2.Canny(image, threshold1, threshold2)` for edge detection. Experiment with the thresholds to understand their effect.
4. **Plotting**:
   - If using Matplotlib, use `plt.subplot` to arrange images side by side and `plt.imshow` to display them.

Expected Output

- Display the original image.
- Display the edge-detected image showing prominent edges while suppressing noise.

In [1]:
```
# Download assignment files
!wget https://github.com/buntyke/vnr_dlcv2024_labs/releases/download/DLCVLab3/golde
```

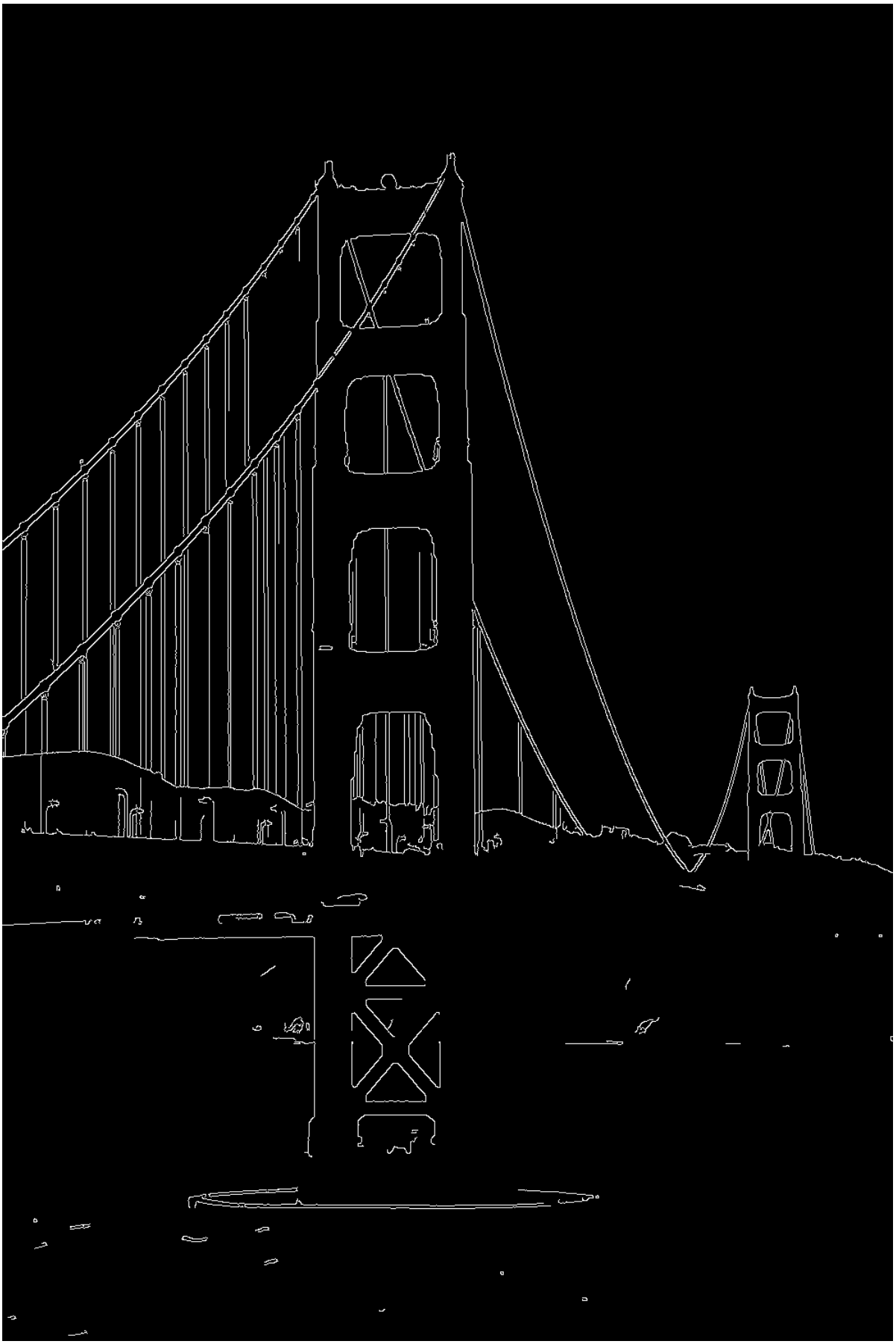
```
--2024-12-08 07:03:10--  https://github.com/buntyke/vnr_dlcv2024_labs/releases/dow
nload/DLCVLab3/golden-gate.jpeg
Resolving github.com (github.com)... 140.82.114.3
Connecting to github.com (github.com)|140.82.114.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e
65be/878811324/d168a62b-a7d7-43e2-a244-5cbcd93aaa4a?X-Amz-Algorithm=AWS4-HMAC-SHA2
56&X-Amz-Credential=releaseassetproduction%2F20241208%2Fus-east-1%2Fs3%2Faws4_requ
est&X-Amz-Date=20241208T070310Z&X-Amz-Expires=300&X-Amz-Signature=4aeda463f2b98b6b
0232d841d3d0a1c569c512cfcc00d21a0299593151e7ff94&X-Amz-SignedHeaders=host&response
-content-disposition=attachment%3B%20filename%3Dgolden-gate.jpeg&response-content-
type=application%2Foctet-stream [following]
--2024-12-08 07:03:10--  https://objects.githubusercontent.com/github-production-r
elease-asset-2e65be/878811324/d168a62b-a7d7-43e2-a244-5cbcd93aaa4a?X-Amz-Algorithm
=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetproduction%2F20241208%2Fus-east-1%2
Fs3%2Faws4_request&X-Amz-Date=20241208T070310Z&X-Amz-Expires=300&X-Amz-Signature=4
aeda463f2b98b6b0232d841d3d0a1c569c512cfcc00d21a0299593151e7ff94&X-Amz-SignedHeader
s=host&response-content-disposition=attachment%3B%20filename%3Dgolden-gate.jpeg&re
sponse-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.19
9.111.133, 185.199.110.133, 185.199.109.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.19
9.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 207679 (203K) [application/octet-stream]
Saving to: ‘golden-gate.jpeg’

golden-gate.jpeg    100%[===================>] 202.81K  --.-KB/s    in 0.04s

2024-12-08 07:03:11 (5.39 MB/s) - ‘golden-gate.jpeg’ saved [207679/207679]
```

In [2]:
```
### WRITE CODE HERE ###
import cv2
from google.colab.patches import cv2_imshow
```

```python
# Load an image
image = cv2.imread('./golden-gate.jpeg',cv2.IMREAD_GRAYSCALE)

# Apply Gaussian blur to reduce noise
blurred = cv2.GaussianBlur(image, (5, 5), 0)

# Apply Canny edge detection
edges = cv2.Canny(blurred, threshold1=100, threshold2=200) # You can adjust the thr

print("Original image")
# Original Image
cv2_imshow(image)

print("Edge image")
# Edge Image
cv2_imshow(edges)
```

```
Original image
```

Edge image



In [ ]: