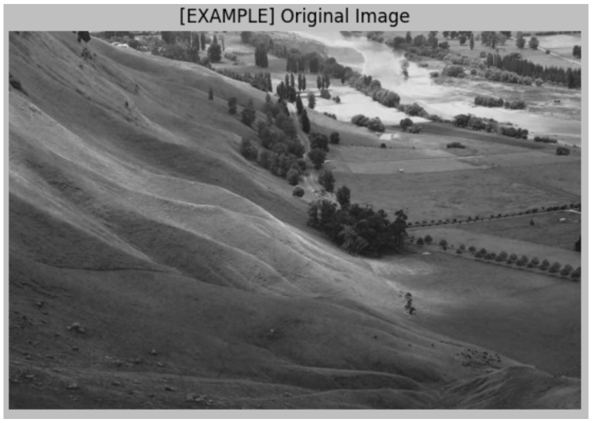
Lab 3: Image Processing

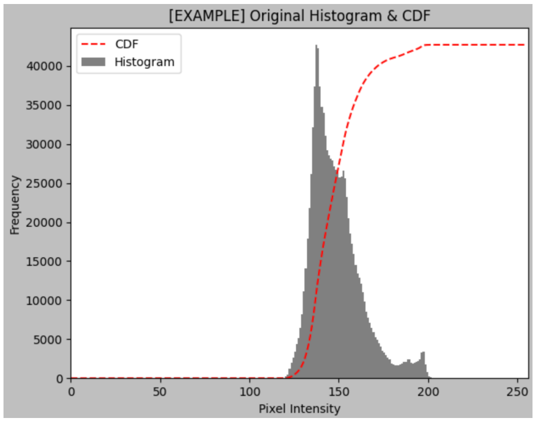
Name: AAAAAAAAAAAA   Student ID: 0000000000000

[Please delete these red sentences and my example answers for Question 1 from your submission. The examples are provided for reference only—you do not need to follow every detail. Just make sure to include all required answers.]

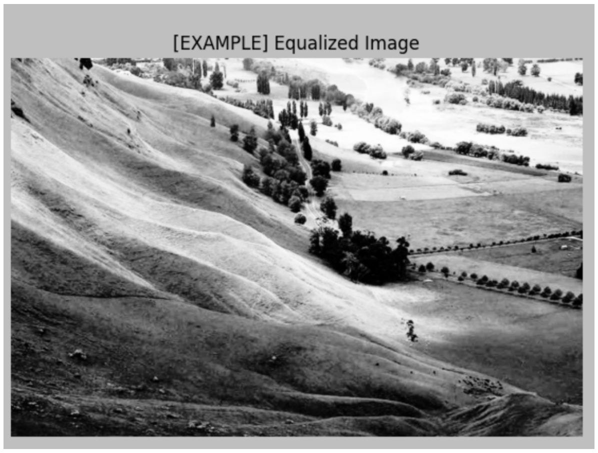
1. Histogram equalization

1. Compute and visualize histogram and cumulative distance function (CDF) of an input gray-scale image

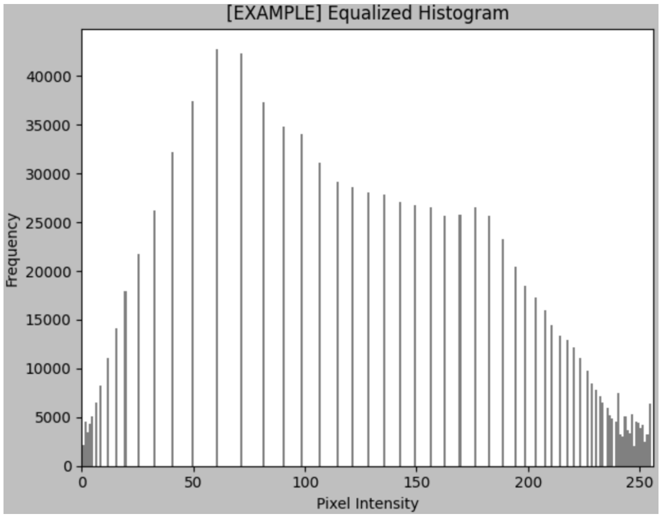

[EXAMPLE] Original Image


[EXAMPLE] Original Histogram & CDF

2. Apply histogram equalization using obtained CDF on the input image


[EXAMPLE] Equalized Image

3. Compute and visualize histogram of output image



2. Image denoising

1. Read the input image and convert to a grayscale image

2. Add two types of noise including Gaussian noise and Salt/Pepper noise (Implement your own functions to add noise to an image)

3. Implement mean and median filtering in 5x5 windows

4. Check if mean or median filtering is able to completely remove Gaussian noise or Salt/Pepper noise. Compare original image and denoised image.

3. Image gradient

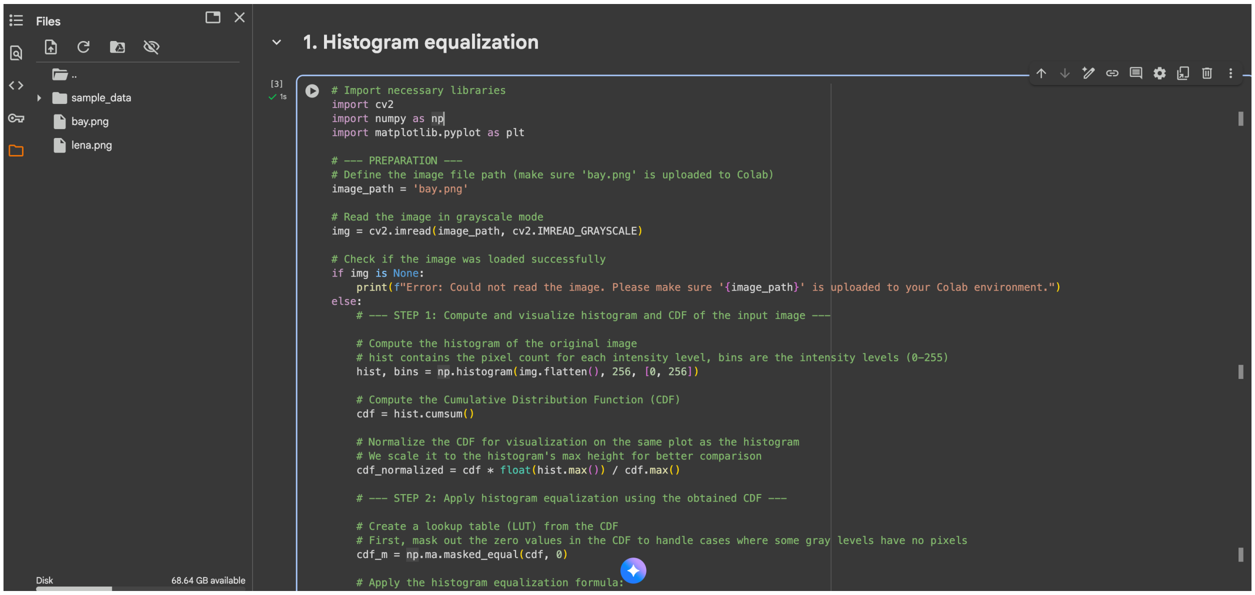
1. Compute image gradient in x and y direction respectively

2. Read the input image and convert to a grayscale image

3. Compute magnitude of image gradient for each pixel

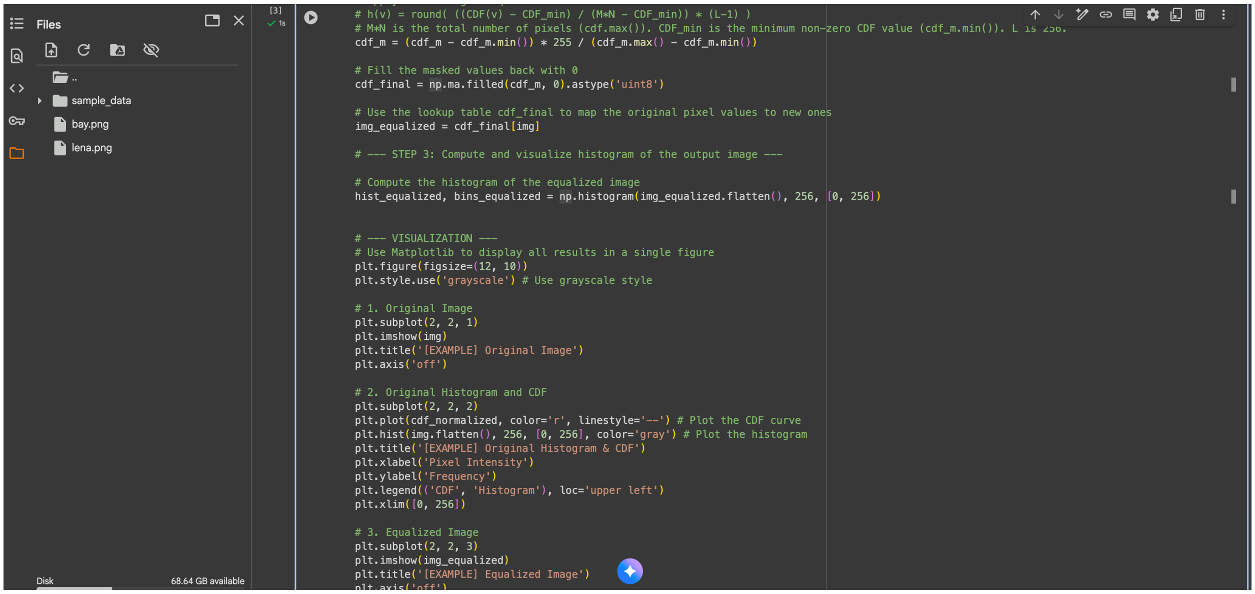
4. Thresholding on magnitude to determine image edges, try various thresholds.

4. Code screenshots

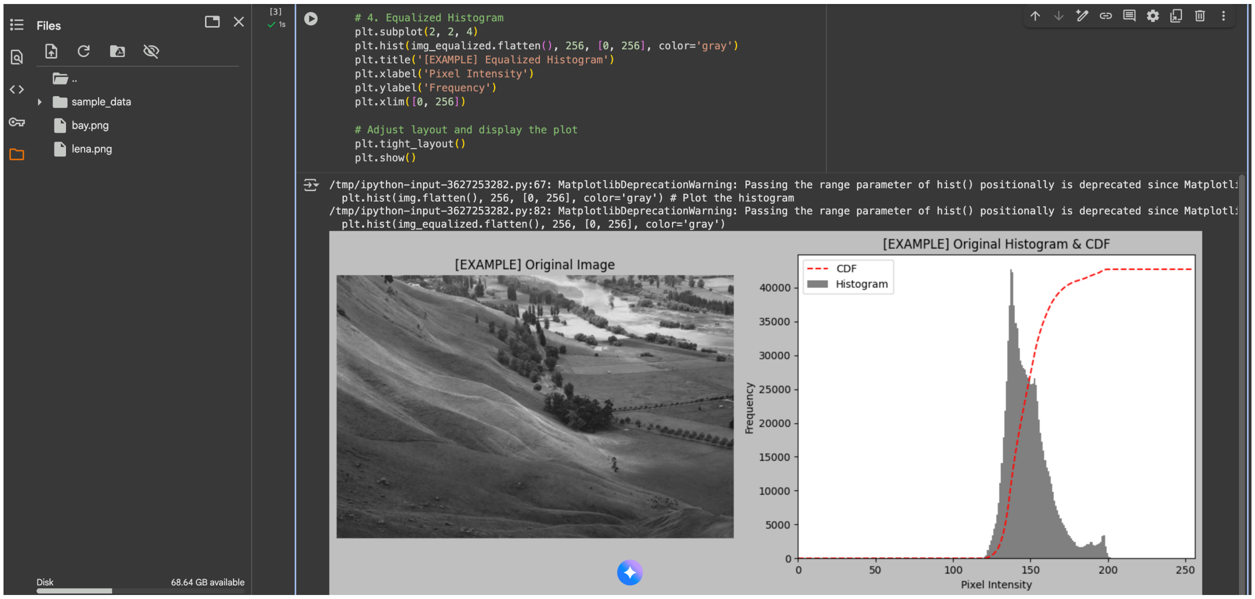


1. Histogram equalization

```
# Import necessary libraries
import cv2
import numpy as np
import matplotlib.pyplot as plt

# --- PREPARATION ---
# Define the image file path (make sure 'bay.png' is uploaded to Colab)
image_path = 'bay.png'

# Read the image in grayscale mode
img = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)

# Check if the image was loaded successfully
if img is None:
    print(f"Error: Could not read the image. Please make sure '{image_path}' is uploaded to your Colab environment.")
else:
    # --- STEP 1: Compute and visualize histogram and CDF of the input image ---

    # Compute the histogram of the original image
    # hist contains the pixel count for each intensity level, bins are the intensity levels (0-255)
    hist, bins = np.histogram(img.flatten(), 256, [0, 256])

    # Compute the Cumulative Distribution Function (CDF)
    cdf = hist.cumsum()

    # Normalize the CDF for visualization on the same plot as the histogram
    # We scale it to the histogram's max height for better comparison
    cdf_normalized = cdf * float(hist.max()) / cdf.max()

    # --- STEP 2: Apply histogram equalization using the obtained CDF ---

    # Create a lookup table (LUT) from the CDF
    # First, mask out the zero values in the CDF to handle cases where some gray levels have no pixels
    cdf_m = np.ma.masked_equal(cdf, 0)

    # Apply the histogram equalization formula:
```



```
# h(v) = round( ((CDF(v) - CDF_min) / (M*N - CDF_min)) * (L-1) )
# M*N is the total number of pixels (cdf.max()). CDF_min is the minimum non-zero CDF value (cdf_m.min()). L is 256.
cdf_m = (cdf_m - cdf_m.min()) * 255 / (cdf_m.max() - cdf_m.min())

# Fill the masked values back with 0
cdf_final = np.ma.filled(cdf_m, 0).astype('uint8')

# Use the lookup table cdf_final to map the original pixel values to new ones
img_equalized = cdf_final[img]

# --- STEP 3: Compute and visualize histogram of the output image ---

# Compute the histogram of the equalized image
hist_equalized, bins_equalized = np.histogram(img_equalized.flatten(), 256, [0, 256])


# --- VISUALIZATION ---
# Use Matplotlib to display all results in a single figure
plt.figure(figsize=(12, 10))
plt.style.use('grayscale') # Use grayscale style

# 1. Original Image
plt.subplot(2, 2, 1)
plt.imshow(img)
plt.title('[EXAMPLE] Original Image')
plt.axis('off')

# 2. Original Histogram and CDF
plt.subplot(2, 2, 2)
plt.plot(cdf_normalized, color='r', linestyle='--') # Plot the CDF curve
plt.hist(img.flatten(), 256, [0, 256], color='gray') # Plot the histogram
plt.title('[EXAMPLE] Original Histogram & CDF')
plt.xlabel('Pixel Intensity')
plt.ylabel('Frequency')
plt.legend(('CDF', 'Histogram'), loc='upper left')
plt.xlim([0, 256])

# 3. Equalized Image
plt.subplot(2, 2, 3)
plt.imshow(img_equalized)
plt.title('[EXAMPLE] Equalized Image')
plt.axis('off')
```



```
# 4. Equalized Histogram
plt.subplot(2, 2, 4)
plt.hist(img_equalized.flatten(), 256, [0, 256], color='gray')
plt.title('[EXAMPLE] Equalized Histogram')
plt.xlabel('Pixel Intensity')
plt.ylabel('Frequency')
plt.xlim([0, 256])

# Adjust layout and display the plot
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-3627253282.py:67: MatplotlibDeprecationWarning: Passing the range parameter of hist() positionally is deprecated since Matplotli
  plt.hist(img.flatten(), 256, [0, 256], color='gray') # Plot the histogram
/tmp/ipython-input-3627253282.py:82: MatplotlibDeprecationWarning: Passing the range parameter of hist() positionally is deprecated since Matplotli
  plt.hist(img_equalized.flatten(), 256, [0, 256], color='gray')
```