DSA0210 Computer Vision with Open CV LAB Experiments

EXPERIMENT-1: Perform basic Image Handling and processing operations on the image.

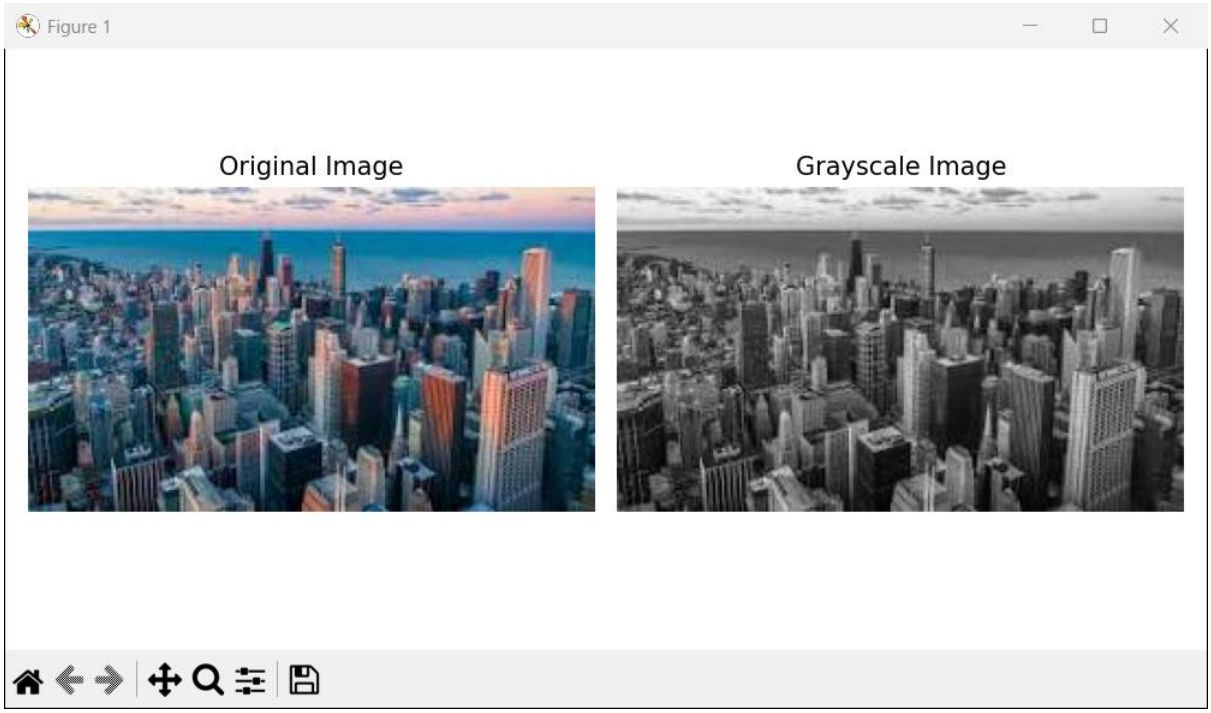
a. Read an image in python and convert the given Image into Grayscale

PROGRAM:

```
import cv2
import matplotlib.pyplot as plt

# Read the color image
img = cv2.imread(r"D:\New Folder\input.jpeg")

# Check if image is loaded
if img is None:
    raise FileNotFoundError("Image not found. Check the file path.")

# Convert the image to grayscale
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Display original and grayscale images
plt.figure(figsize=(8, 4))

plt.subplot(1, 2, 1)
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.title("Original Image")
plt.axis("off")

plt.subplot(1, 2, 2)
plt.imshow(gray, cmap="gray")
```

```
plt.title("Grayscale Image")
plt.axis("off")

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

OUTPUT:



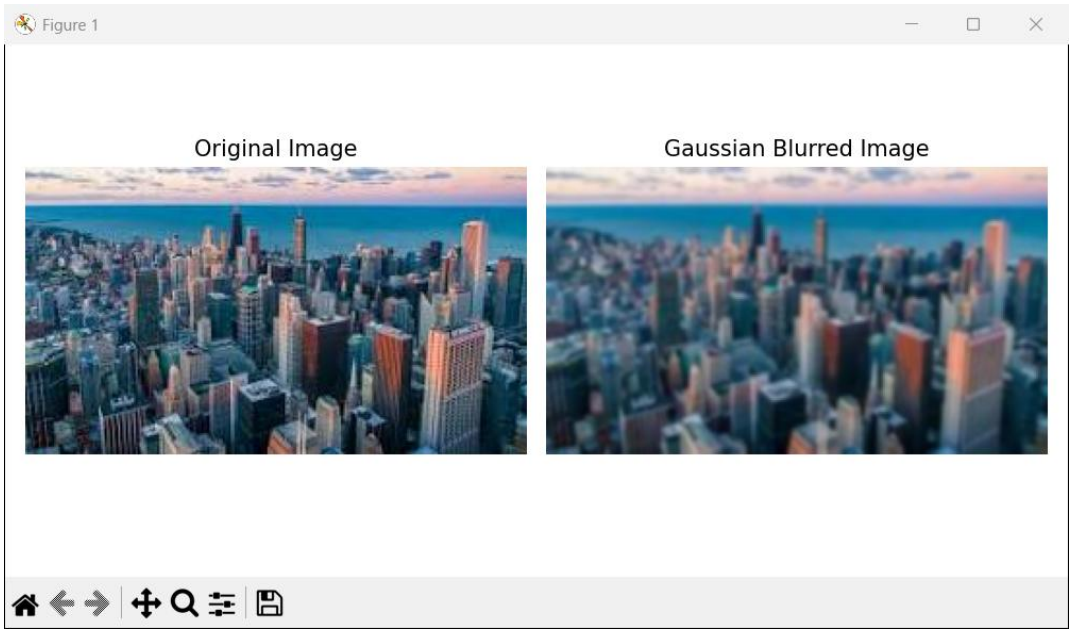
b. Read an image in python and Convert an Image to Blur using GaussianBlur.

PROGRAM:

```
import cv2
import matplotlib.pyplot as plt

# Read the input image
img = cv2.imread(r"D:\New Folder\input.jpeg")

# Check if image is loaded
if img is None:
    raise FileNotFoundError("Image not found. Check the file path.")
```

```
# Apply Gaussian Blur
blur = cv2.GaussianBlur(img, (5, 5), 0)

# Display original and blurred images
plt.figure(figsize=(8, 4))

plt.subplot(1, 2, 1)
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.title("Original Image")
plt.axis("off")

plt.subplot(1, 2, 2)
plt.imshow(cv2.cvtColor(blur, cv2.COLOR_BGR2RGB))
plt.title("Gaussian Blurred Image")
plt.axis("off")

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

OUTPUT:



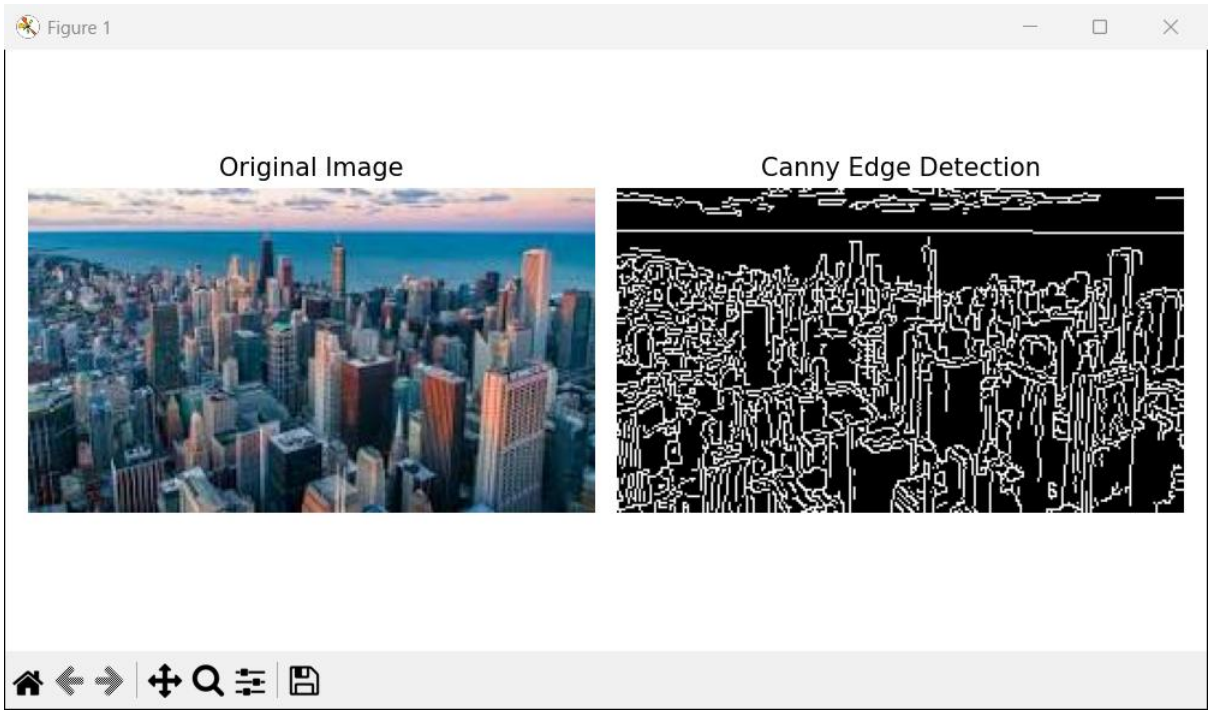
c. Read an image in python and Convert the given Image to show outline using Canny function.

PROGRAM:

```
import cv2
import matplotlib.pyplot as plt


# Read the input image
img = cv2.imread(r"D:\New Folder\input.jpeg")


# Check if image is loaded
if img is None:
    raise FileNotFoundError("Image not found. Check the file path.")


# Convert image to grayscale
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)


# Apply Canny edge detection
edges = cv2.Canny(gray, 100, 200)


# Display original and edge images
plt.figure(figsize=(8, 4))


plt.subplot(1, 2, 1)
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.title("Original Image")
plt.axis("off")


plt.subplot(1, 2, 2)
plt.imshow(edges, cmap="gray")
plt.title("Canny Edge Detection")
```

```
plt.axis("off")


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

OUTPUT:

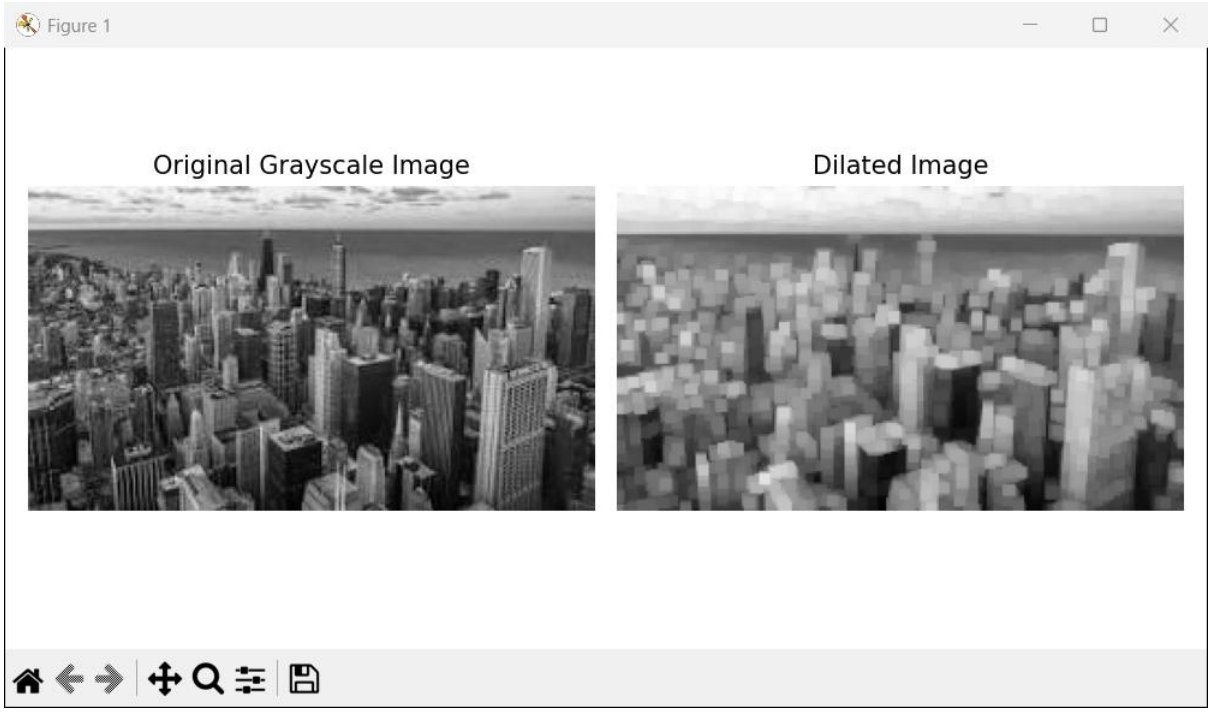


d. Read an image in python and Dilate an Image using Dilate function.

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


# Read the input image
img = cv2.imread(r"D:\New Folder\input.jpeg")


# Check if image is loaded
if img is None:
    raise FileNotFoundError("Image not found. Check the file path.")


# Convert image to grayscale
```

```
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Create a kernel for dilation
kernel = np.ones((5, 5), np.uint8)

# Apply dilation
dilated = cv2.dilate(gray, kernel, iterations=1)

# Display original and dilated images
plt.figure(figsize=(8, 4))

plt.subplot(1, 2, 1)
plt.imshow(gray, cmap="gray")
plt.title("Original Grayscale Image")
plt.axis("off")

plt.subplot(1, 2, 2)
plt.imshow(dilated, cmap="gray")
plt.title("Dilated Image")
plt.axis("off")

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

OUTPUT:



e. Read an image in python and Erode an Image using erode function.

PROGRAM:

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

# Read the input image
img = cv2.imread(r"D:\New Folder\input.jpeg")

# Check if image is loaded
if img is None:
    raise FileNotFoundError("Image not found. Check the file path.")

# Convert image to grayscale
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Create a kernel for erosion
```

```
kernel = np.ones((5, 5), np.uint8)

# Apply erosion
eroded = cv2.erode(gray, kernel, iterations=1)

# Display original and eroded images
plt.figure(figsize=(8, 4))

plt.subplot(1, 2, 1)
plt.imshow(gray, cmap="gray")
plt.title("Original Grayscale Image")
plt.axis("off")

plt.subplot(1, 2, 2)
plt.imshow(eroded, cmap="gray")
plt.title("Eroded Image")
plt.axis("off")

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

OUTPUT: