1.

from google.colab import files

uploaded = files.upload()

import cv2

import matplotlib.pyplot as plt

import numpy as np

from PIL import Image

import io

filename = next(iter(uploaded))

image = Image.open(io.BytesIO(uploaded[filename]))

image\_np = np.array(image)

image\_bgr = cv2.cvtColor(image\_np, cv2.COLOR\_RGB2BGR)

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

plt.figure(figsize=(10,4))

plt.subplot(1,2,1)

plt.imshow(cv2.cvtColor(image\_bgr, cv2.COLOR\_BGR2RGB))

plt.title("Original Image")

plt.axis('off')

plt.subplot(1,2,2)

plt.imshow(gray\_image, cmap='gray')

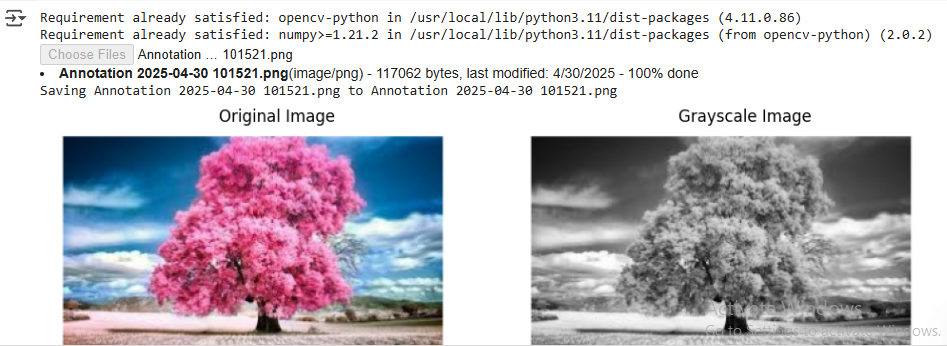
plt.title("Grayscale Image")

plt.axis('off')

plt.tight\_layout()

plt.show()

OUTPUT:



2.

from google.colab import files

uploaded = files.upload()

import cv2

import numpy as np

import matplotlib.pyplot as plt

from PIL import Image

import io

filename = next(iter(uploaded))

image = Image.open(io.BytesIO(uploaded[filename]))

image\_np = np.array(image)

image\_bgr = cv2.cvtColor(image\_np, cv2.COLOR\_RGB2BGR)

blurred\_image = cv2.GaussianBlur(image\_bgr, (11, 11), 0)

plt.figure(figsize=(10,4))

plt.subplot(1,2,1)

plt.imshow(cv2.cvtColor(image\_bgr, cv2.COLOR\_BGR2RGB))

plt.title("Original Image")

plt.axis('off')

plt.subplot(1,2,2)

plt.imshow(cv2.cvtColor(blurred\_image, cv2.COLOR\_BGR2RGB))

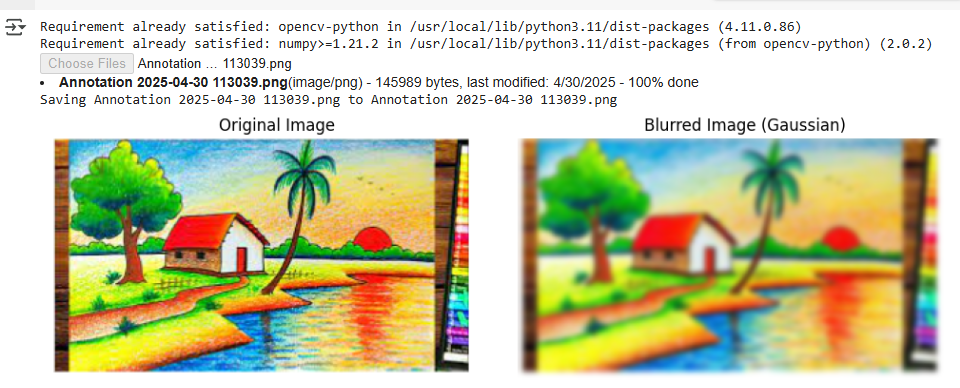
plt.title("Blurred Image (Gaussian)")

plt.axis('off')

plt.tight\_layout()

plt.show()

OUTPUT:



3.

from google.colab import files

uploaded = files.upload()

import cv2

import numpy as np

import matplotlib.pyplot as plt

from PIL import Image

import io

filename = next(iter(uploaded))

image = Image.open(io.BytesIO(uploaded[filename]))

image\_np = np.array(image)

image\_bgr = cv2.cvtColor(image\_np, cv2.COLOR\_RGB2BGR)

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

edges = cv2.Canny(gray\_image, threshold1=100, threshold2=200)

plt.figure(figsize=(10, 4))

plt.subplot(1, 2, 1)

plt.imshow(cv2.cvtColor(image\_bgr, cv2.COLOR\_BGR2RGB))

plt.title("Original Image")

plt.axis('off')

plt.subplot(1, 2, 2)

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

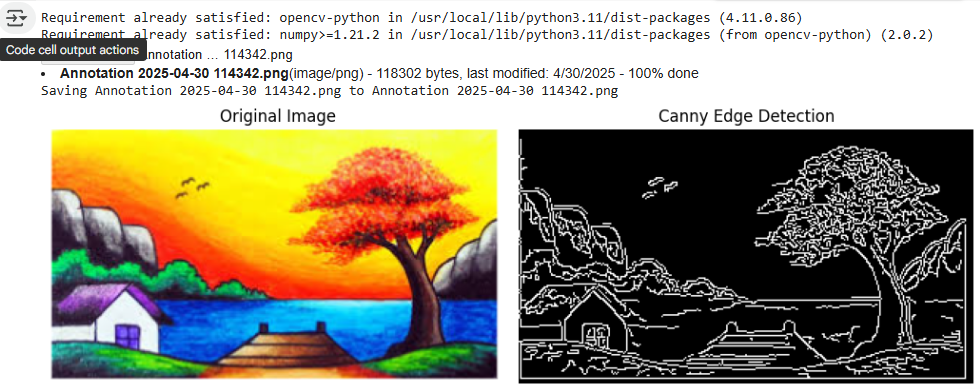
plt.title("Canny Edge Detection")

plt.axis('off')

plt.tight\_layout()

plt.show()

OUTPUT:



4.

import cv2

import numpy as np

import matplotlib.pyplot as plt

from google.colab import files

def display\_image(image, title='Image'):

    plt.imshow(cv2.cvtColor(image, cv2.COLOR\_BGR2RGB))

    plt.title(title)

    plt.axis('off')

    plt.show()

uploaded = files.upload()

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

image = cv2.imread(image\_path)

if image is None:

    print("Error: Could not read the image.")

else:

    # Display the original image

    display\_image(image, title='Original Image')

    # Define a kernel for dilation

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

    # Dilate the image

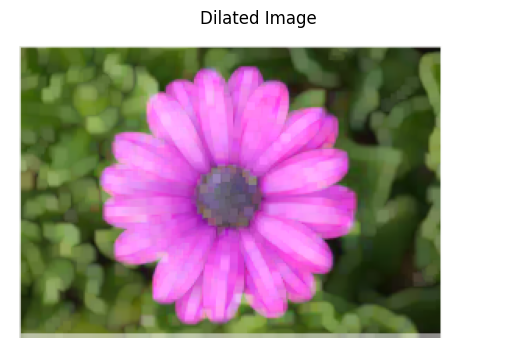
    dilated\_image = cv2.dilate(image, kernel, iterations=1)

    # Display the dilated image

    display\_image(dilated\_image, title='Dilated Image')

OUTPUT:





5.

import cv2

import numpy as np

import matplotlib.pyplot as plt

from google.colab import files

def display\_image(image, title='Image'):

    plt.imshow(cv2.cvtColor(image, cv2.COLOR\_BGR2RGB))

    plt.title(title)

    plt.axis('off')

    plt.show()

uploaded = files.upload()

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

image = cv2.imread(image\_path)

if image is None:

    print("Error: Could not read the image.")

else:

    display\_image(image, title='Original Image')

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

    eroded\_image = cv2.erode(image, kernel, iterations=1)

    display\_image(eroded\_image, title='Eroded Image')

OUTPUT:



