18BCS6201-CV Practical-2 (Gauri Prabhakar) (AI-ML-2)(B)

Aim: To demonstrate the use of different image processing functions using python and OpenCv.

In [1]:
```
# Importing necessary modules.
import cv2
import numpy as np
```

Grayscale

In [2]:
```
# Converting an image to Grayscale.
# Creating a variable to store the image using the '.imread()' function.
image = cv2.imread(r'C:\Users\gauri\Desktop\OpenCV Media\we bear bears.jpg')
# Creating a variable to store the grayscale image using the function '.cvtColor()'.
grayImage = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
# Returning the modified image.
cv2.imshow('Grayscale', grayImage)
# Setting up '.waitkey()' to wait for a specific time until any key is pressed.
cv2.waitKey(0)
# Destroying all windows.
cv2.destroyAllWindows()
```

Edge Detection

In [3]:
```
# Edge Detection on an image.
# Creating a variable to store the image using the '.imread()' function.
img = cv2.imread(r"C:\Users\gauri\Desktop\OpenCV Media\we bear bears.jpg", 0)
# Creating a variable to store the edge detected image using the function '.Canny()'.
edge_det = cv2.Canny(img, 100, 200)
# Returning the modified image.
cv2.imshow('Edge Detection', edge_det)
# Setting up '.waitkey()' to wait for a specific time until any key is pressed.
cv2.waitKey(0)
# Destroying all windows.
cv2.destroyAllWindows()
```

Average Blur

In [4]:
```
# Smoothing Techniques on an image.
# Average Blur.
# Creating a variable to store the image using the '.imread()' function.
img = cv2.imread(r"C:\Users\gauri\Desktop\OpenCV Media\we bear bears.jpg")
# Creating a variable to store the average blurred image using the function '.blur()'.
blur = cv2.blur(img, (5, 5))
# Returning the image.
cv2.imshow('Before Average Blur', img)
# Returning the modified image.
cv2.imshow('Average Blur', blur)
# Setting up '.waitkey()' to wait for a specific time until any key is pressed.
cv2.waitKey(0)
# Destroying all windows.
cv2.destroyAllWindows()
```

Median Blur

In [5]:
```
# Median Blur
# Creating a variable to store the image using the '.imread()' function.
img = cv2.imread(r"C:\Users\gauri\Desktop\OpenCV Media\we bear bears.jpg")
# Creating a variable to store the median blurred image using the function '.medianBlur()'.
median_blur = cv2.medianBlur(img, 5)
# Returning the image.
cv2.imshow('Before Median Blur', img)
# Returning the modified image.
cv2.imshow('Media Blur', median_blur)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Gaussian Blur

In [6]:
```
# Gaussian Blur
# Creating a variable to store the image using the '.imread()' function.
img = cv2.imread(r"C:\Users\gauri\Desktop\OpenCV Media\we bear bears.jpg")
# Creating a variable to store the gaussian blurred image using the function '.GaussianBlur()'.
Gaussian_blur = cv2.GaussianBlur(img, (5, 5), 0)
# Returning the image.
cv2.imshow('Before Gaussian Blur', img)
# Returning the modified image.
cv2.imshow('Gaussian Blur', median_blur)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Erosion

In [7]:
```
# Erosion on an image.
# Creating a variable to store the image using the '.imread()' function.
img = cv2.imread(r"C:\Users\gauri\Desktop\OpenCV Media\we bear bears.jpg")
# 'np.ones()' returns an array of 1's of the given shape, data type.
kernel = np.ones((5, 5), np.uint8)
# Creating a variable to store the eroded image using the function '.erode()'.
img_erosion = cv2.erode(img, kernel, iterations=1)
# Returning the image.
cv2.imshow('Before Erosion', img)
# Returning the modified image.
cv2.imshow('Erosion', img_erosion)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Dilation

In [8]:
```
# Dilation on an image.
# Creating a variable to store the image using the '.imread()' function.
img = cv2.imread(r"C:\Users\gauri\Desktop\OpenCV Media\we bear bears.jpg")
# 'np.ones()' returns an array of 1's of the given shape, data type.
kernel = np.ones((5, 5), np.uint8)
# Creating a variable to store the dilated image using the function '.dilate()'.
img_dilation = cv2.dilate(img, kernel, iterations=1)
# Returning the image.
cv2.imshow('Before Dilation', img)
# Returning the modified image.
cv2.imshow('Dilation', img_dilation)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Resizing and Cropping

In [9]:
```
# Cropping and Resizing on an image.
# Creating a variable to store the image using the '.imread()' function.
img = cv2.imread(r"C:\Users\gauri\Desktop\OpenCV Media\we bear bears.jpg")
# Returning the shape of the image in the form of an array.
print(img.shape)
# Resizing the image.
imgResize = cv2.resize(img, (1000, 500))
# Returning the shape of the modified image.
print(imgResize.shape)
# Cropping the image.
imgCropped = img[0:200, 200:500]
# Returning the image.
cv2.imshow("Before Resizing and Cropping", img)
# Returning the modified image.
cv2.imshow("Image Resize", imgResize)
# Returning the modified image.
cv2.imshow("Image Cropped", imgCropped)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
(410, 728, 3)
(500, 1000, 3)
```

Shape and Text

In [11]:
```
# Shape and Text on an image.
# 'np.ones()' returns an array of 1's of the given shape, data type.
img = np.zeros((512, 512, 3), np.uint8)
# Returning the image.
print('Before Modifications', img)
img[:] = 204, 204, 255
# Drawing a line on the image, it takes parameters as image, start point, end point, color and finally thickness.
cv2.line(img, (0, 0), (img.shape[1], img.shape[0]), (204, 0, 102), 2)
# Drawing a rectangle on the image, it takes parameters as image, start point, end point, color and finally thickness.
cv2.rectangle(img, (0, 0), (300, 350), (64, 64, 64), 2)
# Drawing a circle on the image, it takes parameters as image, center co-ordinates, radius, color and finally thickness.
cv2.circle(img, (400, 50), 30, (255, 255, 0), 5)
# Writing a text on the image, it takes parameters as image, text, co-ordinates, font, font scale, color, thickness.
cv2.putText(img, "Gauri Prabhakar", (200, 150), cv2.FONT_HERSHEY_COMPLEX_SMALL, 1, (255, 153, 51), 2)
# Returning the modified image.
cv2.imshow("Shape and Text", img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
Before Modifications [[[0 0 0]
  [0 0 0]
  ...
  [0 0 0]
  [0 0 0]
  [0 0 0]]

 [[0 0 0]
  [0 0 0]
  [0 0 0]
  ...
  [0 0 0]
  [0 0 0]
  [0 0 0]]

 [[0 0 0]
  [0 0 0]
  [0 0 0]
  ...
  [0 0 0]
  [0 0 0]
  [0 0 0]]

 ...

 [[0 0 0]
  [0 0 0]
  [0 0 0]
  ...
  [0 0 0]
  [0 0 0]
  [0 0 0]]

 [[0 0 0]
  [0 0 0]
  [0 0 0]
  ...
  [0 0 0]
  [0 0 0]
  [0 0 0]]

 [[0 0 0]
  [0 0 0]
  [0 0 0]
  ...
  [0 0 0]
  [0 0 0]
  [0 0 0]]]
```

In [ ]:
```
```