DRAWING FUNCTION

Creating Images using numpy functions

Since image object is just an numpy array we can create an image directly from a numpy array usig few numpy fucntions.

In [ ]:
```
import numpy as np
import cv2

# Create a black image
img = np.zeros((512,512,3), np.uint8)
cv2.imshow("image",img)
cv2.waitKey(0)test
cv2.destroyAllWindows()
```

The above creates a black image.

We can also change the color of the Image by using array slicing.

In [ ]:
```
from cv2 import cv2
import numpy as np

arr = np.zeros((500,500,3),dtype = 'uint8')

# changing the B channel value of the numpy array to 255
# this causes the image to be blue in color
arr[:,:] = [255,0,0]

cv2.imshow('image',arr)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

The above code creates an Blue colored Image

Drawing Shapes

To draw shapes we will be using the image which we just created using an numpy array as a Canvas.

`cv2.line(), cv2.circle() , cv2.rectangle(), cv2.ellipse(), cv2.putText()`
we can use the above function to create the respective shapes or to put text on the image

In all the above functions, you will see some common arguments as given below:

- **img** : The image where you want to draw the shapes

- **color** : Color of the shape. for BGR, pass it as a tuple, eg: (255,0,0) for blue. For grayscale, just pass the scalar value.
- **thickness** : Thickness of the line or circle etc. If -1 is passed for closed figures like circles, it will fill the shape. default thickness = 1
- **lineType** : Type of line, whether 8-connected, anti-aliased line etc. By default, it is 8-connected. cv2.LINE_AA gives anti-aliased line which looks great for curves.

Drwaing a line

To draw a line, you need to pass starting and ending coordinates of line. We will create a black image and draw a blue line on it from top-left to bottom-right corners.

In [ ]:
```
import numpy as np
from cv2 import cv2

# Create a black image
img = np.zeros((512,512,3), np.uint8)

# Draw a diagonal blue line with thickness of 5 px
img = cv2.line(img,(0,0),(511,511),(255,0,0),5)
cv2.imshow("image",img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Drwaing a Rectangle

To draw a rectangle, you need top-left corner and bottom-right corner of rectangle. This time we will draw a green rectangle at the top-right corner of image.

In [ ]:
```
import numpy as np
import cv2

# Create a black image
img = np.zeros((512,512,3), np.uint8)

# Draw a diagonal blue line with thickness of 5 px
img = cv2.rectangle(img,(384,0),(510,128),(0,255,0),3)
cv2.imshow("image",img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Drwaing a Circle

To draw a circle, you need its center coordinates and radius.

In [ ]:
```
import numpy as np
import cv2

# Create a black image
img = np.zeros((512,512,3), np.uint8)
```

```
center = int(512/2)

# Draw a diagonal blue line with thickness of 5 px
img = cv2.circle(img,(center,center),250,(0,255,0),3)
cv2.imshow("image",img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Drwaing a Ellipse

TTo draw the ellipse, we need to pass several arguments. One argument is the center location (x,y). Next argument is axes lengths (major axis length, minor axis length). angle is the angle of rotation of ellipse in anti-clockwise direction. startAngle and endAngle denotes the starting and ending of ellipse arc measured in clockwise direction from major axis. i.e. giving values 0 and 360 gives the full ellipse. For more details, check the documentation of cv2.ellipse(). Below example draws a half ellipse at the center of the image.

- img (image on which to draw)
- center point
- major and minor axis length
- angle in terms of rotation of the elipse
- start angle
- stop angle
- color
- thickness

In [ ]:
```
import numpy as np
import cv2

# Create a black image
img = np.zeros((512,512,3), np.uint8)

center = int(512/2)

# Draw a diagonal blue line with thickness of 5 px
img = cv2.ellipse(img,(256,256),(100,50),90,0,360,(255,0,0),-1)
cv2.imshow("image",img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Drwaing a Polygon

To draw a polygon, first you need coordinates of vertices. Make those points into an array of shape ROWSx1x2 where ROWS are number of vertices and it should be of type int32. Here we draw a small polygon of with four vertices in yellow color.

In [ ]:
```
import numpy as np
import cv2
```

```
# Create a black image
img = np.zeros((512,512,3), np.uint8)

pts = np.array([[10,5],[20,30],[70,20],[50,10]], np.int32)
print(pts.shape)
print(pts)

print()

pts = pts.reshape((4,1,2))
print(pts.shape)
print(pts)

img = cv2.polylines(img,[pts],False,(0,255,255))

cv2.imshow("image",img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Drwaing a text on images

- To put texts in images, you need specify following things.
- Text data that you want to write
- Position coordinates of where you want put it (i.e. bottom-left corner where data starts).
- Font type (Check cv2.putText() docs for supported fonts)
- Font Scale (specifies the size of font)
- regular things like color, thickness, lineType etc. For better look, lineType = cv2.LINE_AA is recommended.

In [ ]:
```
import numpy as np
import cv2

# Create a black image
img = np.zeros((512,512,3), np.uint8)

font = cv2.FONT_HERSHEY_SIMPLEX
cv2.putText(img,'OpenCV',(10,400), font, 1,(255,255,255),2)

cv2.imshow("image",img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Creating simple animation with OpenCV and Numpy

In [ ]:
```
from cv2 import cv2
import numpy as np

img = np.zeros((500,500,3), np.uint8)

squ = np.full((50,50,3),127,np.uint8)

rep = np.zeros((50,1,3),np.uint8)
```

```
# img[450:,i:i+50,::] = squ

for i in range(1,450):

    img[450:,i-1:i,::] = rep
    #img[450:,i:i+50,::] = squ
    cv2.imshow('image',img)

    k = cv2.waitKey(50) & 0xFF
    if k == ord('m'):
        mode = not mode
    elif k == 27:
        break

cv2.destroyAllWindows()
```

In [ ]:

```
from cv2 import cv2
import numpy as np

img = np.zeros((500,500,3), np.uint8)


squ = np.full((50,50,3),127,np.uint8)

rep = np.zeros((50,1,3),np.uint8)

# img[450:,i:i+50,::] = squ

for i in range(1,450):

    img[450:,i-1:i,::] = rep
    img[450:,i:i+50,::] = squ
    cv2.imshow('image',img)

    k = cv2.waitKey(50) & 0xFF
    if k == ord('m'):
        mode = not mode
    elif k == 27:
        break

cv2.destroyAllWindows()
```

HOMEWORK

1. make a pattern with circles on the screen
2. Generate random circles on screen
3. make checkboxes on screen

HOMEWORK SOLUTION

In [28]:

```
#TASK 1
# TASk 2
import numpy as np
import cv2
```

```
# Create a black image
img = np.zeros((512,512,3), np.uint8)
# Draw a diagonal blue line with thickness of 5 px
for i in range(0,512,50):
    img = cv2.circle(img,(i,i),50,(0,255,0),3)
cv2.imshow("image",img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

In [26]:

```
# TASk 2
import numpy as np
import random
import cv2

# Create a black image
img = np.zeros((512,512,3), np.uint8)

center = int(512/2)

# Draw a diagonal blue line with thickness of 5 px
for i in range(0,100):
    x=random.randint(0,500)
    y=random.randint(0,500)
    r=random.randint(0,255)
    g=random.randint(0,255)
    b=random.randint(0,255)
    size=random.randint(0,50)

    img = cv2.circle(img,(x,y),size,(r,g,b),3)
cv2.imshow("image",img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

In [34]:

```
#TASK 3

import numpy as np
from cv2 import cv2

# Create a black image
img = np.zeros((512,512,3), np.uint8)

# Draw a diagonal blue line with thickness of 5 px
for i in range(0,512,50):
    img = cv2.line(img,(i,0),(i,512),(255,0,0),3)
    img2 = cv2.line(img,(0,i),(512,i),(255,0,0),3)
cv2.imshow("image",img)
cv2.imshow("image",img2)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

In [ ]: