SunilOS

Image Processing

www.sunilos.com

www.raystec.com

Think IT Think Us RAYS

Image Processing

- Image Processing is a process to perform some operation on the image such as:
  - Change color of the image
  - Resize image
  - Crop the image etc.
- There are many libraries available for image processing. Some common libraries are following: we will Use OpenCv for Image Processing.

SunilOS

Image Processing Libraries

- **Scikit-image:**
  - It uses a Numpy array as an image object. Skimage transforms the image file into n-dimensional array objects. The ndarray can be type of integer and float. It is fast because written in c
- **OpenCV:**
  - It is written in C++ and Python. Opencv can work with Numpy, scipy and matplotlib. It is used in Image processing, face detection, Object detection etc.
- **Mahotas:**
  - It uses advanced features such as image processing. It can process 2D and 3D images. It can extract information from images. It has more than 100 functions for computer vision processing.

SunilOS

Image Processing Libraries

- **SimpleiTk:**
  - It treats images as set of points in a space. It support many dimensions such as 2D, 3D and 4D
- **SciPy:**
  - Scipy has a module named scipy.ndimage for image processing. Scipy basically used for scientific calculations
- **Pillow:**
  - It support many image formats such. It is advanced version of PIL.
- **Matplotlib:**
  - It is used for visualization or graph plotting, but we can use it for image processing. It does not support all image file formats. It can used for image altering

What is OpenCV?

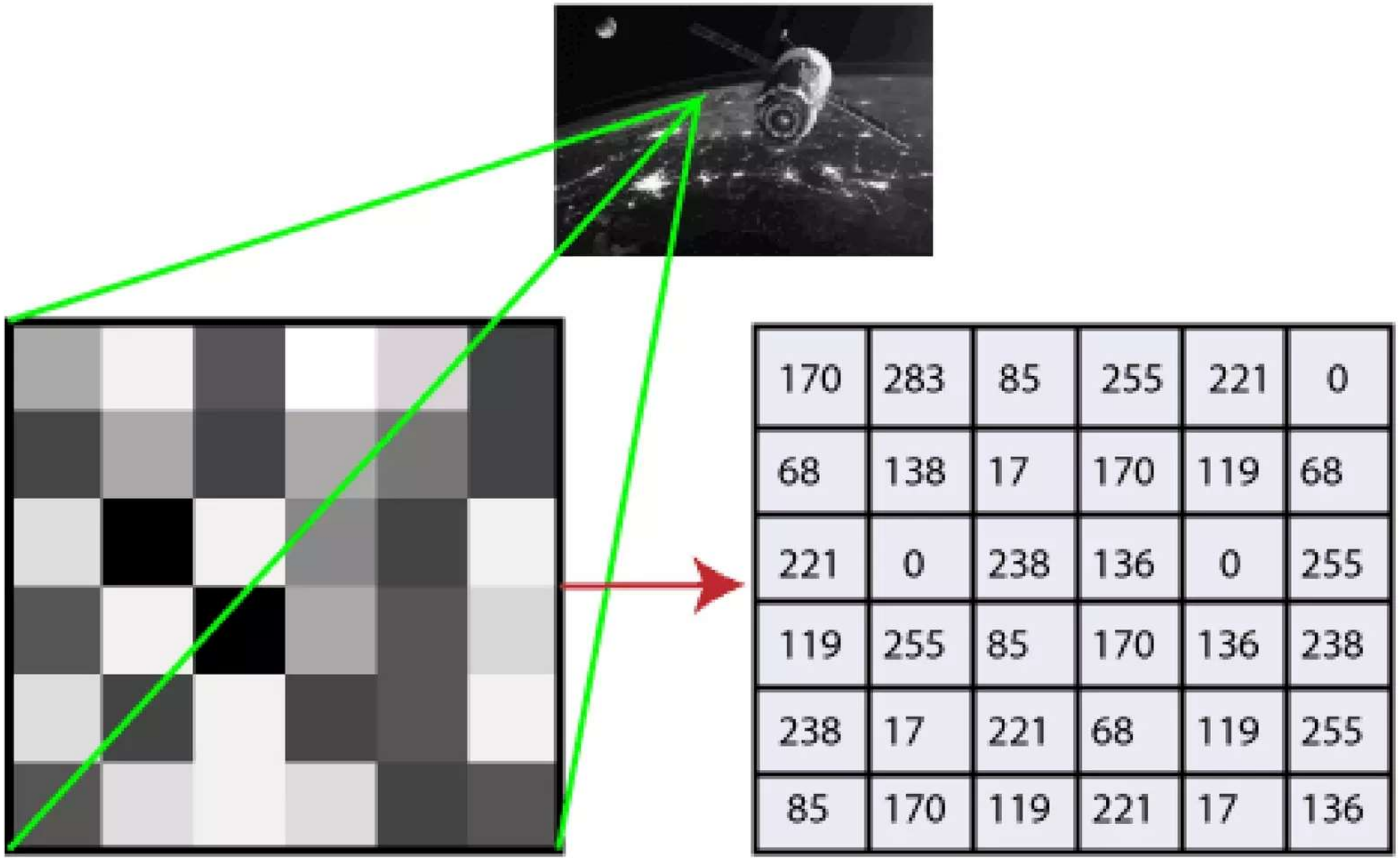
- OpenCV is a Python open-source library, which is used for computer vision in Artificial intelligence, Machine Learning, face recognition, etc.

SunilOS

How OpenCV Works:

- Human eye provides information whatever he sees. But the machine knows only numbers. The Machine converts images into numbers on the basis of pixel data and stores them in memory. The picture intensity represented as a number.

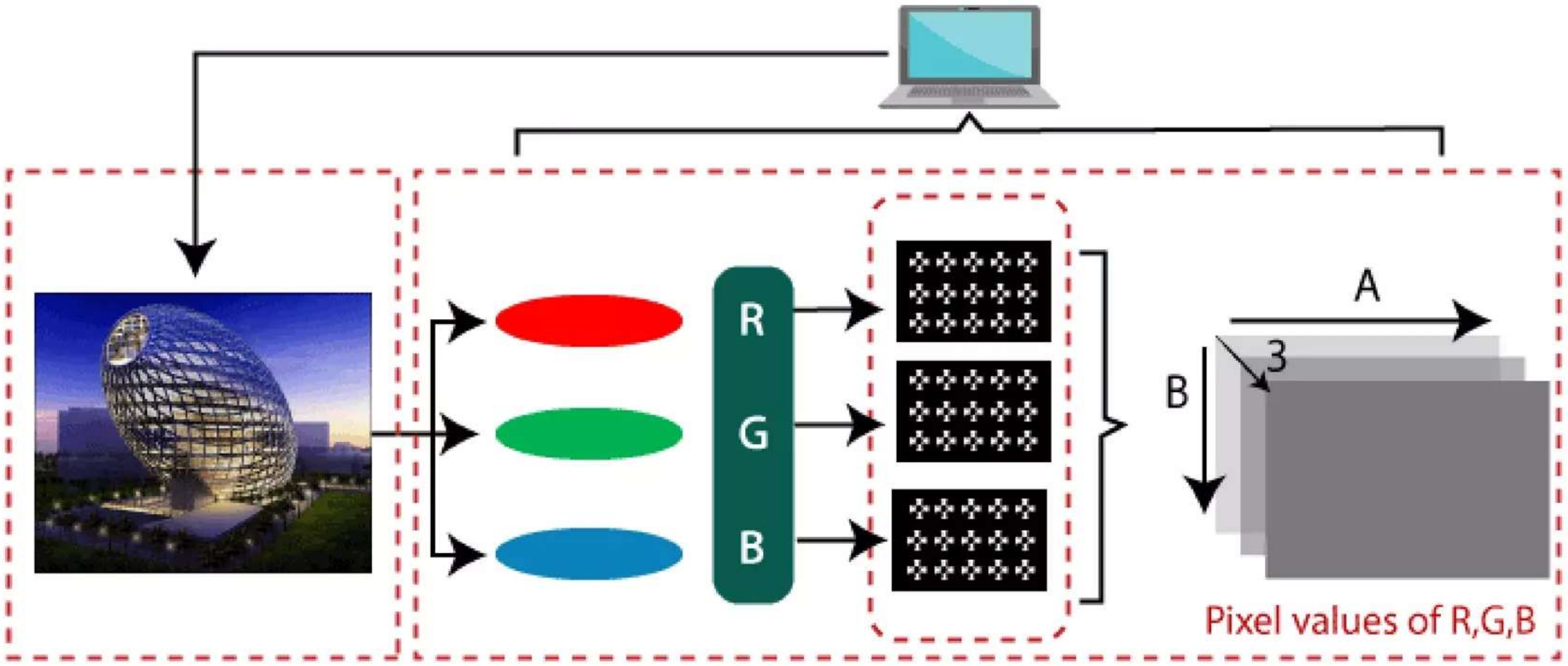
How OpenCV Works:



170	283	85	255	221	0
68	138	17	170	119	68
221	0	238	136	0	255
119	255	85	170	136	238
238	17	221	68	119	255
85	170	119	221	17	136

Representation of Image

- Gray scale: black and white images
- RGB: Colorful images



Why is OpenCV used for Computer Vision?

- ❑Free of cost
- ❑It is fast.
- ❑Requires less.
- ❑It is portable.

How to install OpenCv:

- Open command prompt and type
  - `pip install opencv-python`

- Open conda prompt
  - `conda install -c conda-forge opencv` in anaconda prompt

-

Supported File Format

- Window bitmaps - *.bmp, *.dib
- JPEG files - *.jpeg, *.jpg, *.jpe
- Portable Network Graphics - *.png
- Portable image format- *.pbm, *.pgm, *.ppm
- TIFF files - *.tiff, *.tif

Reading and writing Images

- `import cv2`
- **# read a file as grayscale**
- `img=cv2.imread("Desert.jpg",0)`
- `cv2.imshow("Image",img)`
- `cv2.waitKey(0)`

- **# write to a file**
- `res=cv2.imwrite("copy.jpg",img)`
- `print("status:",res)`

OpenCV Basic Operation on Images:

- ❑Access pixel values and modify them
- ❑Access Image Properties
- ❑Setting Region of Image
- ❑Splitting and merging images
- ❑Change the image color

Accessing Image Properties

- `import cv2`
- `img = cv2.imread("Desert.jpg",1)`
- `print("Height:",img.shape[0])`
- `print("Width:",img.shape[1])`
- `print("channel:",img.shape[2])`
- `print("size:",img.size)`

Making Borders for Images:

SunilOS

- `import cv2 as cv`
- `BLUE = [255,0,0]`
- `img1 = cv.imread("Desert.jpg",1)`
- `img =`
- `cv.copyMakeBorder(img1,50,10,10,10,cv.BORDER_CONSTANT, value=BLUE)`

SunilOS

OpenCV Resize the image:

```
❑ import cv2
❑ img = cv2.imread("Desert.jpg", 1)
❑ scale = 60
❑ width = int(img.shape[1] * scale / 100)
❑ height = int(img.shape[0] * scale / 100)
❑ dim = (width, height)
❑ print("Initial dimension ",img.shape)
❑ # resize image
❑ resized = cv2.resize(img, dim)
❑ print('Resized Dimensions : ', resized.shape)
❑ cv2.imshow("Resized image", resized)
❑ cv2.waitKey(0)
❑ cv2.destroyAllWindows()
```

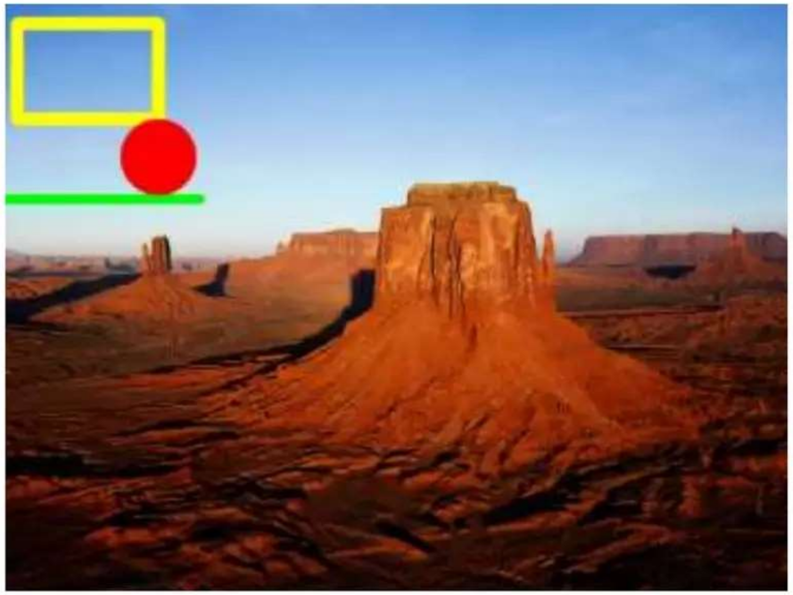
OpenCV Drawing Functions

- **cv2.circle():**
  - cv2.circle(img, center, radius, color[,thickness [, lineType[,shift]]])
- **cv2.rectangle():**
  - cv2.rectangle(img, pt1, pt2, color[, thickness[,lineType[,shift]]])
- **cv2.ellipse():**
  - cv2.ellipse(img, center, axes, angle, startAngle, endAngle, color[, thickness[, lineType[, shift]]])
- **cv2.ellipse:**
  - Cv2.ellipse(img, box, color[, thickness[, lineType]])
- **cv2.line:**
  - cv2.line(img, pt1, pt2, color[, thickness[,lineType[, shift]]])
- **Put text:**
  - cv2.putText(img, text, org, font, color)
- **To Draw Polylines:**
  - cv2.polyLine(img, polys, is_closed, color, thickness=1, lineType=8, shift=0)

SunilOS

Drawing some shapes on image

- ❑ `import numpy as np`
- ❑ `import cv2`
- ❑ `img = cv2.imread("Desert.jpg",1)`
- ❑ `cv2.rectangle(img,(15,25),(200,150),(0,255,55),15)`
- ❑ `cv2.imshow('image',img)`
- ❑ `cv2.waitKey(0)`
- ❑ `cv2.destroyAllWindows()`



SunilOS

Write Text to Image:

- import cv2
- font = cv2.FONT_HERSHEY_SIMPLEX
- **# Create a black image.**
- img = cv2.imread("Desert.jpg",1)
- cv2.putText(img,'Rays Technology',(10,50),font,2,(100,255,255),2)
- **#Display the image**
- cv2.imshow("image",img)
- cv2.waitKey(0)

Crop the Image

```
import cv2
img=cv2.imread("Desert.jpg")
y=300
x=200
h=100
w=200
crop = img[y:y+h, x:x+w]
# crop=img[300,500]
cv2.imshow("Image:",crop)
cv2.waitKey(0)
```

Crop rectangle image to circle

```
import cv2
img=cv2.imread("Koala.jpg")
img1=cv2.resize(img,(300,300))
h,w,c=img1.shape
print(h,w,c)
center=(w//2,h//2)
radius=180
color=(255,255,255)#white
thickness=80
cv2.circle(img1,center,radius,color,thickness)
cv2.imshow("Cropped",img1)
cv2.waitKey(0)
cv2.imwrite("Koala1.jpg",img1)
```

OpenCV Resize the image:

```
import cv2
img = cv2.imread("Desert.jpg", 1)
scale = 60
width = int(img.shape[1] * scale / 100)
height = int(img.shape[0] * scale / 100)
dim = (width, height)
print("Initial dimension ",img.shape)
```

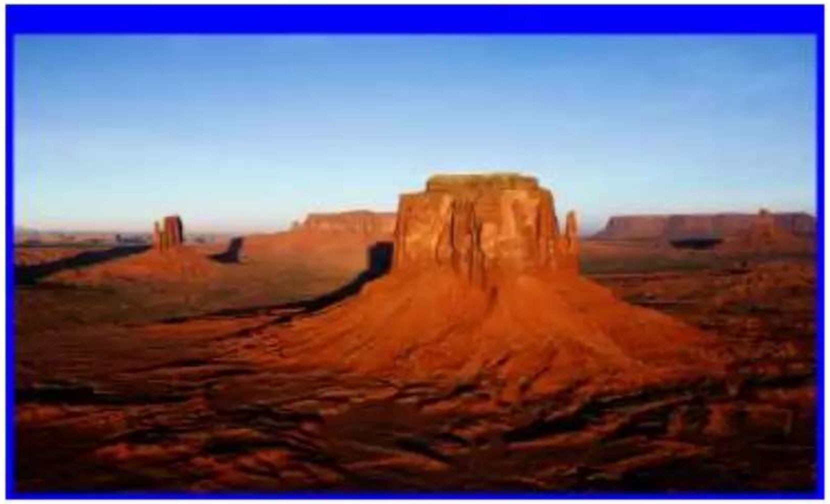
- **# resize image**

```
resized = cv2.resize(img, dim)
print('Resized Dimensions : ', resized.shape)
cv2.imshow("Resized image", resized)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

SunilOS

Making Borders for Images:

- import cv2 as cv
- BLUE = [255,0,0]
- img1 = cv.imread("Desert.jpg",1)
- replicate =cv.copyMakeBorder(img1,50,10,10,10,cv.BORDER_CONSTANT, value=BLUE)
- cv.imshow("replicate",replicate)
- Cv.waitKey(0)



OpenCV Canny Edge Detection:

- import cv2
- img = cv2.imread('Koala.jpg')
- edges = cv2.Canny(img, 100, 200)
- cv2.imshow("Edge Detected Image", edges)
- cv2.imshow("Original Image", img)
- cv2.waitKey(0) **# waits until a key is pressed**
- cv2.destroyAllWindows()