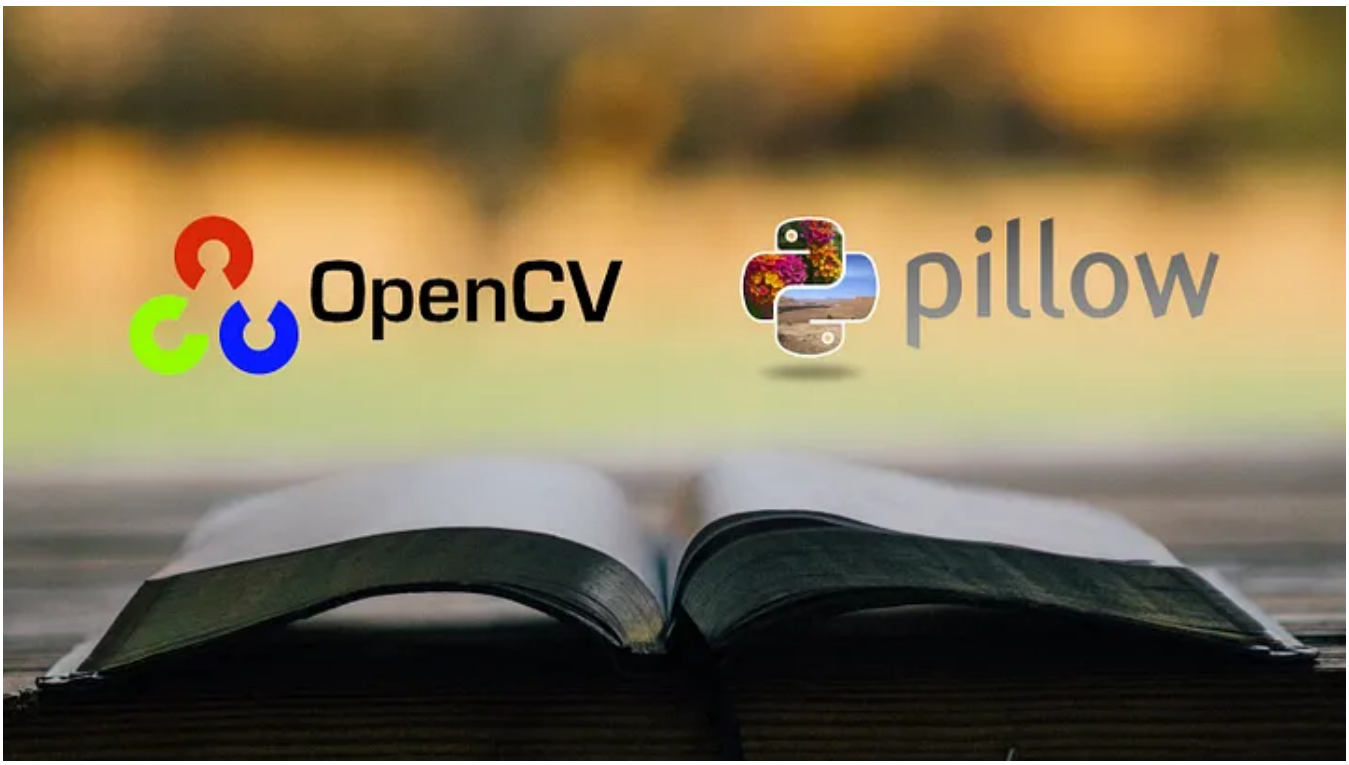
✨ Member-only story

The Ultimate Handbook for OpenCV & Pillow

Steins · Follow
Published in Analytics Vidhya
7 min read · Sep 12, 2021

Listen Share ••• More



Introduction

OpenCV and Pillow are the commonly used libraries in Python for image processing.

In this article, I will list out all the codes that are useful regarding OpenCV and Pillow side by side, so that you can compare them easily.

New Version

Please note that all the new content and full versions of this tutorial will be moved to CodoRaven.com > OpenCV vs Pillow.

Updates

- 15-Jun-22: Added **[13. RGBA to RGB]**.
- 16-Jul-22: Added "specify background color" to **[6. Rotate]**.
- 28-Jan-23: Updated `Image.ANTIALIAS` to `Image.Resampling.LANCZOS` **[5. Resize]**.
- 29-Jan-23: Updated `np.float` to `np.float32`; `Image.PERSPECTIVE` to `Image.Transform.PERSPECTIVE`; `Image.BICUBIC` to `Image.Resampling.BICUBIC` **[10. Warp / Transform]**.

0. Install & Import

To install Pillow and OpenCV, please type the following in your terminal (replace ***pip*** with ***pip3*** if your python3 is using pip3):

```
pip install numpy
pip install opencv-python
pip install Pillow
```

Before using them, you have to import them into your python file with the following codes:

```
from PIL import Image
import cv2
import numpy as np
```

> *Note: Since OpenCV images are actually NumPy arrays, we would also like to install and import the NumPy library.*

1. Read

Read/open a colorful image:

```
pil_img = Image.open("your_image.jpg")  # RGB
```

```
cv2_img = cv2.imread("your_image.jpg")  # BGR
```

Read/open a grayscale image:

```
pil_img = Image.open("your_image.jpg").convert("L")
cv2_img = cv2.imread("your_image.jpg", cv2.IMREAD_GRAYSCALE)
```

More examples:

- Read Image | OpenCV vs Pillow | Python
- Show Image | OpenCV vs Pillow | Python

2. Write

Write/save an image:

```
pil_img.save("new_image.jpg")
cv2.imwrite("new_image.jpg", cv2_img)
```

Write/save a JPEG image with specific quality:

```
pil_img.save("new_image.jpg", quality=95)
cv2.imwrite("new_image.jpg", cv2_img,
[int(cv2.IMWRITE_JPEG_QUALITY), 95])
```

More examples:

- Save Image | OpenCV vs Pillow | Python
- Save JPEG With Specific Quality | OpenCV vs Pillow | Python

3. Conversion

- Pillow image to OpenCV image:

```
cv2_img = np.array(pil_img)
cv2_img = cv2.cvtColor(cv2_img, cv2.COLOR_RGB2BGR)
```

- OpenCV image to Pillow image:

```
cv2_img = cv2.cvtColor(cv2_img, cv2.COLOR_BGR2RGB)
pil_img = Image.fromarray(cv2_img)
```

> *Note: OpenCV images are in BGR color format, while Pillow images are in RGB color format. So we have to manaully convert the color format from one to another.*

More examples:

- Convert Between OpenCV and PIL Images | Python

4. Shape / Size

Get the width & height & number of channels of an image.

- OpenCV:

```
if cv2_img.ndim == 2:
  height, width = cv2_img.shape
  channels = 1
else:
  height, width, channels = cv2_img.shape
```

- Pillow:

```
width, height = pil_img.size
```

```
cv2_img = np.array(pil_img)
if cv2_img.ndim == 2:
  channels = 1
else:
  channels = cv2_img.shape[-1]
```

> *Note: It is hard to get the number of channels directly from a Pillow image object, the easier way to do this would be to first convert it to an OpenCV image (ndarray) and then*

> *get the shape.*

More examples:

- Image Shape/Size | OpenCV vs Pillow | Python

5. Resize

Resize without preserving the aspect ratio:

```
pil_img_resized = pil_img.resize((NEW_WIDTH, NEW_HEIGHT))

cv2_img_resized = cv2.resize(cv2_img, (NEW_WIDTH, NEW_HEIGHT))
```

Resize and preserve the aspect ratio:

- OpenCV:

```
scale_ratio = 0.6
new_width = int(cv2_img.shape[1] * scale_ratio)
new_height = int(cv2_img.shape[0] * scale_ratio)
new_size = (new_width, new_height)

cv2_img_resized = cv2.resize(cv2_img, new_size,
interpolation=cv2.INTER_AREA)
```

- Pillow:

```
# scale ratio = min(max_width/width, max_height/height)
max_width = 256
max_height = 256
pil_img.thumbnail((max_width, max_height), Image.Resampling.LANCZOS)
```

More examples:

- Resize Image | OpenCV vs Pillow | Python

- Resize Image & Keep Aspect Ratio | OpenCV vs Pillow | Python

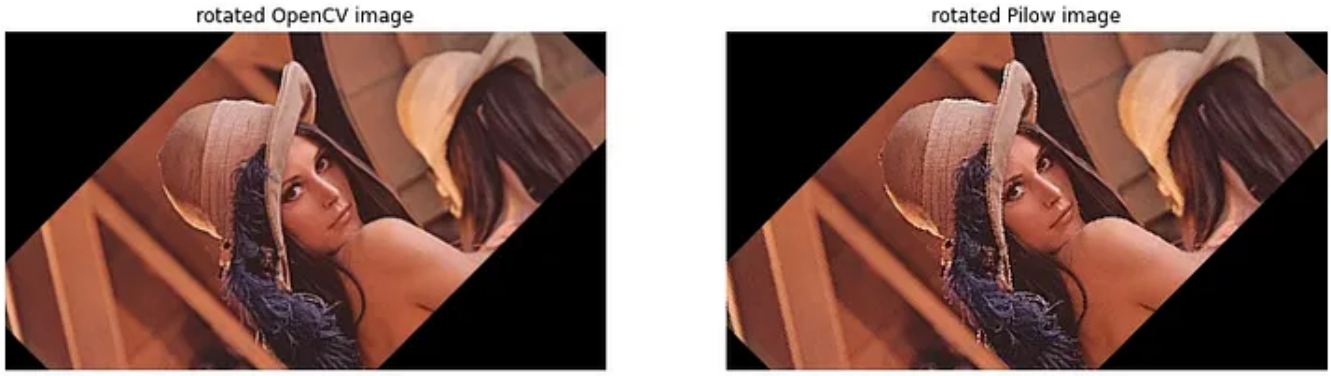
6. Rotate

Rotate an OpenCV image by 90, 180, 270 degrees

- OpenCV (only 90, 180, 270 degrees):

```
# rotate 90 degrees clockwise
rotated_cv2_img = cv2.rotate(cv2_img, cv2.ROTATE_90_CLOCKWISE)

# rotate 180 degrees
rotated_cv2_img = cv2.rotate(cv2_img, cv2.ROTATE_180)

# rotate 270 degrees clockwise (90 degrees counterclockwise)
rotated_cv2_img = cv2.rotate(cv2_img,
cv2.ROTATE_90_COUNTERCLOCKWISE)
```

Rotate an image by any angle



Rotate images without boundings

- OpenCV (any angle):

```
# rotate 45 degrees counterclockwise
h, w = cv2_img.shape[:2]
```

```
M = cv2.getRotationMatrix2D(center=(w/2, h/2), angle=45, scale=1.0)
rotated_cv2_img = cv2.warpAffine(cv2_img, M, (w, h))

# specify background color
rotated_cv2_img = cv2.warpAffine(cv2_img, M, (w, h),
borderMode=cv2.BORDER_CONSTANT, borderValue=(0,0,0))
```

- Pillow (any angle):

```
# rotate 45 degrees counterclockwise
rotated_pil_img = pil_img.rotate(45)

# specify background color
rotated_pil_img = pil_img.rotate(45, fillcolor=(0,0,0))
```

Rotate an image with bounding

Rotate images with bounding

- OpenCV:

> *Note: you need to install imutils by:* `pip install imutils`

```
import imutils

# rotate 45 degrees counterclockwise
rotated_cv2_img = imutils.rotate_bound(cv2_img, -45)
```

- Pillow:

```
rotated_pil_img = pil_img.rotate(45, expand=True)
```

More examples:

- Rotate Image | OpenCV vs Pillow | Python
- Rotate Image Without Cropping | OpenCV vs Pillow | Python

7. Flip

- OpenCV:

```
# flip vertically
flipped_cv2_img = cv2.flip(cv2_img, flipCode=0)

# flip horizontally
mirrored_cv2_img = cv2.flip(cv2_img, flipCode=1)

# flip vertically & horizontally (ratate 180 degrees)
rotated_cv2_img = cv2.flip(cv2_img, flipCode=-1)
```

- Pillow:

```
from PIL import Image, ImageOps

# flip vertically
flipped_pil_img = ImageOps.flip(pil_img)

# flip horizontally
mirrored_pil_img = ImageOps.mirror(pil_img)
```

More examples:

- Mirror/Flip Image | OpenCV vs Pillow | Python

8. Translate

Translate the image 10 pixels to the right and 20 pixels down

- OpenCV:

```
shift_x = 10
shift_y = 20
h, w = cv2_img.shape[:2]

M = np.float32([[1, 0, shift_x],[0, 1, shift_y]])
shifted_cv2_img = cv2.warpAffine(cv2_img, M, (w, h))
```

- Pillow:

```
shift_x = 10
shift_y = 20

shifted_pil_img = pil_img.rotate(0, translate=(shift_x, shift_y))
```

> *Note:* `pil_img.rotate(angle, translate)` *will translate before rotation.*

More examples:

- Image Translation | OpenCV vs Pillow | Python

9. Background Color

You can specify the background color when you rotate/translate the image.

Rotate with specified background-color

- OpenCV:

```
c = (127, 255, 255)  # BGR
cv2_img = cv2.warpAffine(cv2_img, M, (cols, rows), borderValue=c)
```

- Pillow:

```
c = (255, 255, 127)  # RGB
pil_img = pil_img.rotate(45, fillcolor=c)
```

More examples:

- Rotate/Translate Image with Background Color | OpenCV vs Pillow | Python

10. Warp / Transform

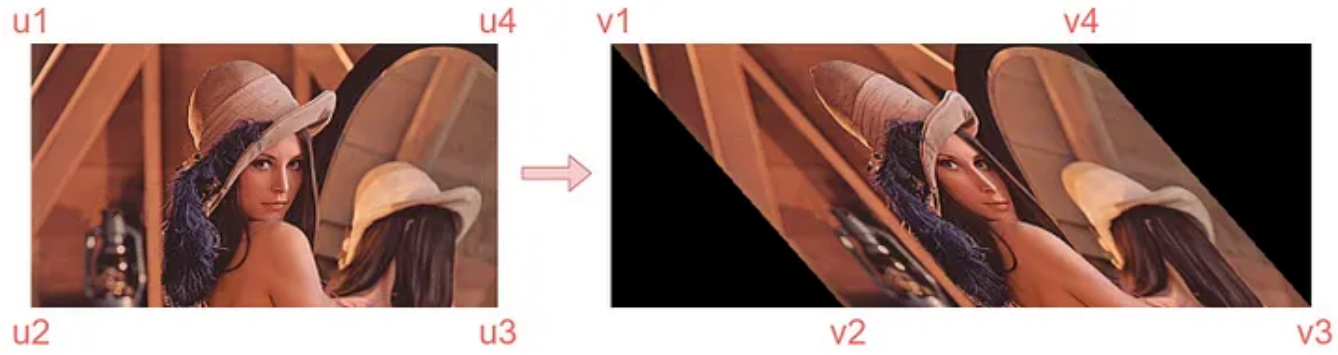
Warp images using the perspective transformation.

- OpenCV:

```
h, w = cv2_img.shape[:2]
shear_x = int(round(0.5 * w))
new_w = w + shear_x

src_rect = np.array([[0, 0], [0, h], [w, h], [w, 0]],
dtype=np.float32)
dst_rect = np.array([[0, 0], [shear_x, h], [new_w, h], [w, 0]],
dtype=np.float32)
```

```
M = cv2.getPerspectiveTransform(src_rect, dst_rect)
warped_cv2_img = cv2.warpPerspective(cv2_img, M, (new_w, h))
```

> *src_rect = np.array([u1, u2, u3, u4], dtype=np.float32)*
>
> *dst_rect = np.array([v1, v2, v3, v4], dtype=np.float32)*



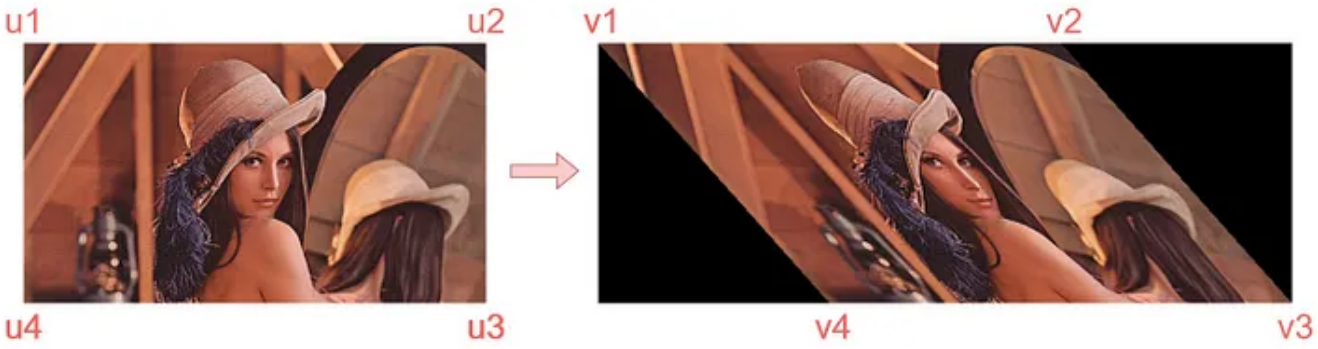
initial vertices & final vertices for src_rect & dst_rect

- Pillow:

```
def find_coeffs(pa, pb):
  matrix = []
  for p1, p2 in zip(pb, pa):
    matrix.append([p1[0], p1[1], 1, 0, 0, 0,
                   -p2[0]*p1[0], -p2[0]*p1[1]])
    matrix.append([0, 0, 0, p1[0], p1[1], 1,
                   -p2[1]*p1[0], -p2[1]*p1[1]])

  A = np.matrix(matrix, dtype=np.float32)
  B = np.array(pa).reshape(8)

  res = np.dot(np.linalg.inv(A.T * A) * A.T, B)
  return np.array(res).reshape(8)

w, h = pil_img.size
shear_x = int(round(0.5 * w))
new_w = w + shear_x
coeffs = find_coeffs([(0, 0), (w, 0), (w, h), (0, h)],
                     [(0, 0), (w, 0), (new_w, h), (shear_x, h)])

warped_pil_img = pil_img.transform((new_w, h),
Image.Transform.PERSPECTIVE, coeffs, Image.Resampling.BICUBIC)
```

> *Where `pa` is the four vertices in the current plane, and `pb` contains four vertices in the resulting plane.* ***(find_coeffs function, modified from here)***

```
coeffs = find_coeffs([u1, u2, u3, u4], [v1, v2, v3, v4])
```



Initial vertices & final vertices for find_coeffs(pa, pb)

More examples:

- Warp/Transform Image | OpenCV vs Pillow | Python

11. Base64

Read image file as base64:

```
import base64

with open("your_image.jpg", "rb") as f:
  base64_str = base64.b64encode(f.read())
```

Conversion between Pillow & base64:

```
import base64
from io import BytesIO
from PIL import Image

def pil_to_base64(pil_img):
  img_buffer = BytesIO()
  pil_img.save(img_buffer, format='JPEG')
  byte_data = img_buffer.getvalue()
  base64_str = base64.b64encode(byte_data)
  return base64_str

def base64_to_pil(base64_str):
  pil_img = base64.b64decode(base64_str)
  pil_img = BytesIO(pil_img)
```

```
    pil_img = Image.open(pil_img)
    return pil_img
```

Conversion between OpenCV & base64:

```
import base64
import numpy as np
import cv2

def cv2_base64(cv2_img):
  base64_str = cv2.imencode('.jpg', cv2_img)[1].tostring()
  base64_str = base64.b64encode(base64_str)
  return base64_str

def base64_cv2(base64_str):
  imgString = base64.b64decode(base64_str)
  nparr = np.fromstring(imgString, np.uint8)
  cv2_img= cv2.imdecode(nparr, cv2.IMREAD_COLOR)
  return cv2_img
```

More examples:

- Image & Base64 | OpenCV vs Pillow | Python

12. Read from URL

Read an image from a URL.

- OpenCV (without request headers):

```
import urllib.request

res = urllib.request.urlopen(url)
arr = np.asarray(bytearray(res.read()), dtype=np.uint8)
cv2_img = cv2.imdecode(arr, cv2.IMREAD_UNCHANGED)
```

- OpenCV (with request headers):

```
import urllib.request

headers = {'referer': 'https://www.google.com'}
req = urllib.request.Request(url, headers=headers)
res = urllib.request.urlopen(req)
```

```
arr = np.asarray(bytearray(res.read()), dtype=np.uint8)
cv2_img = cv2.imdecode(arr, cv2.IMREAD_UNCHANGED)
```

- Pillow (without request headers):

```
import requests

pil_img = Image.open(requests.get(url, stream=True).raw)
```

- Pillow (with request headers):

```
import requests

headers = {'referer': 'https://www.google.com'}
pil_img = Image.open(requests.get(url, headers=headers,
stream=True).raw)
```

More examples:

- Read Image From URL | OpenCV vs Pillow | Python

13. RGBA to RGB

Convert transparent pixels to white pixels (by pasting the RGBA image on a white RGB image).

- OpenCV:

```
def cv2_RGBA2RGB(img):
  b, g, r, a = cv2.split(img)
  alpha = a / 255
  r = (255 * (1 - alpha) + r * alpha).astype(np.uint8)
  g = (255 * (1 - alpha) + g * alpha).astype(np.uint8)
  b = (255 * (1 - alpha) + b * alpha).astype(np.uint8)
  new_img = cv2.merge((b, g, r))
  return new_img
```

- Pillow:

```
def pil_RGBA2RGB(img):
  img.load() # for png.split()
  bg = Image.new("RGB", img.size, (255, 255, 255))
  bg.paste(img, mask=img.split()[3]) # 3 is the alpha channel
  return bg
```

More examples:

- Correctly Convert RGBA to RGB | OpenCV vs Pillow | Python
- Convert RGBA to RGB | OpenCV vs Pillow | Python

Conclusion

These are all the codes that I think will be commonly used. I will update this list if I find any new useful functions in image processing in the future.

If I have left out anything important or you think there are still other useful functions that I can add to this handbook, please leave a comment and let me know.

My previous articles

Anime Illustration Colorization with Deep Learning — Part 1

This project aims to create...

medium.com

Anime Illustration Colorization with Deep Learning — Part 2

This is an update on my previous article Anime Illustration Colorization with Deep Learning — Part 1.

medium.com

References

- https://note.nkmk.me/en/python-pillow-rotate/
- https://www.pyimagesearch.com/2021/02/03/opencv-image-translation/
- https://www.pyimagesearch.com/2017/01/02/rotate-images-correctly-with-opencv-and-python/
- https://stackoverflow.com/questions/14177744/how-does-perspective-transformation-work-in-pil
- https://www.pyimagesearch.com/2014/05/05/building-pokedex-python-opencv-perspective-warping-step-5-6/
- https://note.nkmk.me/python-opencv-warp-affine-perspective/
- https://stackoverflow.com/questions/25618756/how-to-convert-alpha-channel-of-the-image-to-white-color-using-opencv-python
- https://stackoverflow.com/questions/9166400/convert-rgba-png-to-rgb-with-pil

Python Opencv Pillow Computer Vision Image Processing

Follow

Written by Steins

872 Followers · Writer for Analytics Vidhya

Developer & AI Researcher. Write about AI, web dev/hack. Be my referred member: https://medium.com/@steinsfu/membership. Support me: https://ko-fi.com/steinsfu

More from Steins and Analytics Vidhya

Steins

Stable Diffusion—ControlNet Clearly Explained!

Generating images from line art, scribble, or pose key points using Stable Diffusion and ControlNet.

6 min read · Jun 6, 2023

443 3

Kia Eisinga in Analytics Vidhya

How to create a Python library

Ever wanted to create a Python library, albeit for your team at work or for some open source project online? In this blog you will learn...

7 min read · Jan 27, 2020

2.2K 27

Shashank Singhal in Analytics Vidhya

Hadoop : How to install in 5 Steps in Windows 10

An easy to go guide for installing the Hadoop in Windows

7 min read · Mar 27, 2021

133 14

Steins

Stable Diffusion Clearly Explained!

How does Stable Diffusion paint an AI artwork? Understanding the tech behind the rise of AI-generated art.

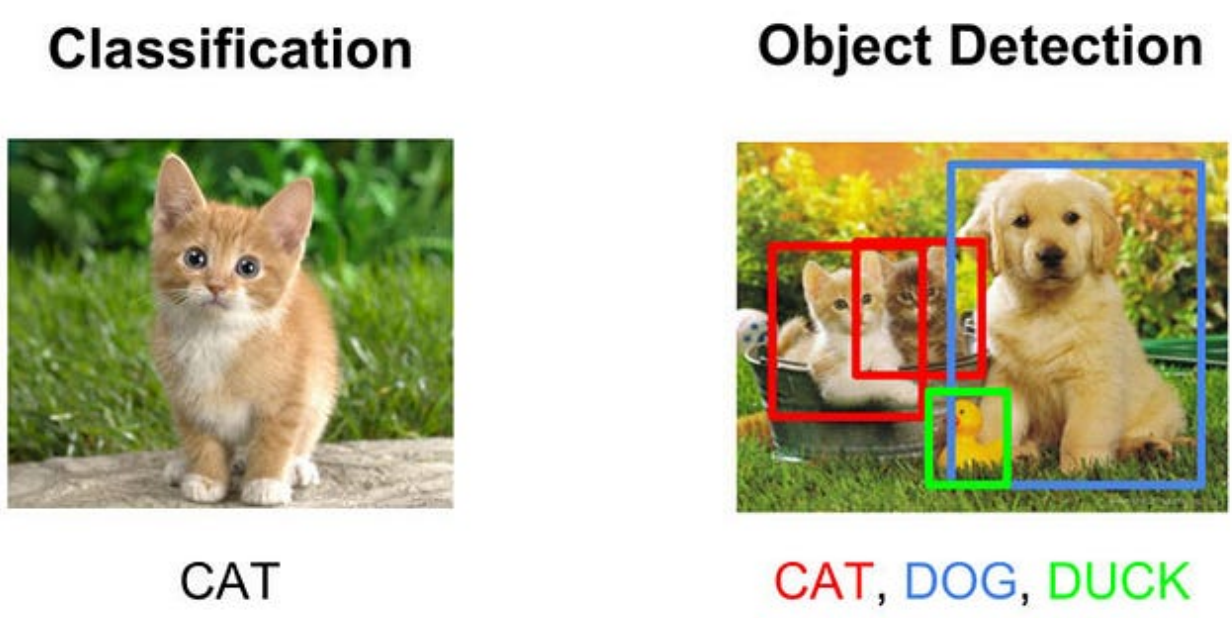
· 5 min read · Jan 2, 2023

1.4K 11

See all from Steins

See all from Analytics Vidhya

Recommended from Medium



Siddheshwar Harkal

Image Classification with YOLOv8

Ultralytics published the latest version of the YOLOv8 (You Only Look Once) model in January 2023 which is a new state-of-the-art (SOTA)...

5 min read · Aug 26, 2023

7

Henrique Vedoveli

Image Filtering Techniques in Image Processing—Part 1

1. Introduction

11 min read · Aug 26, 2023

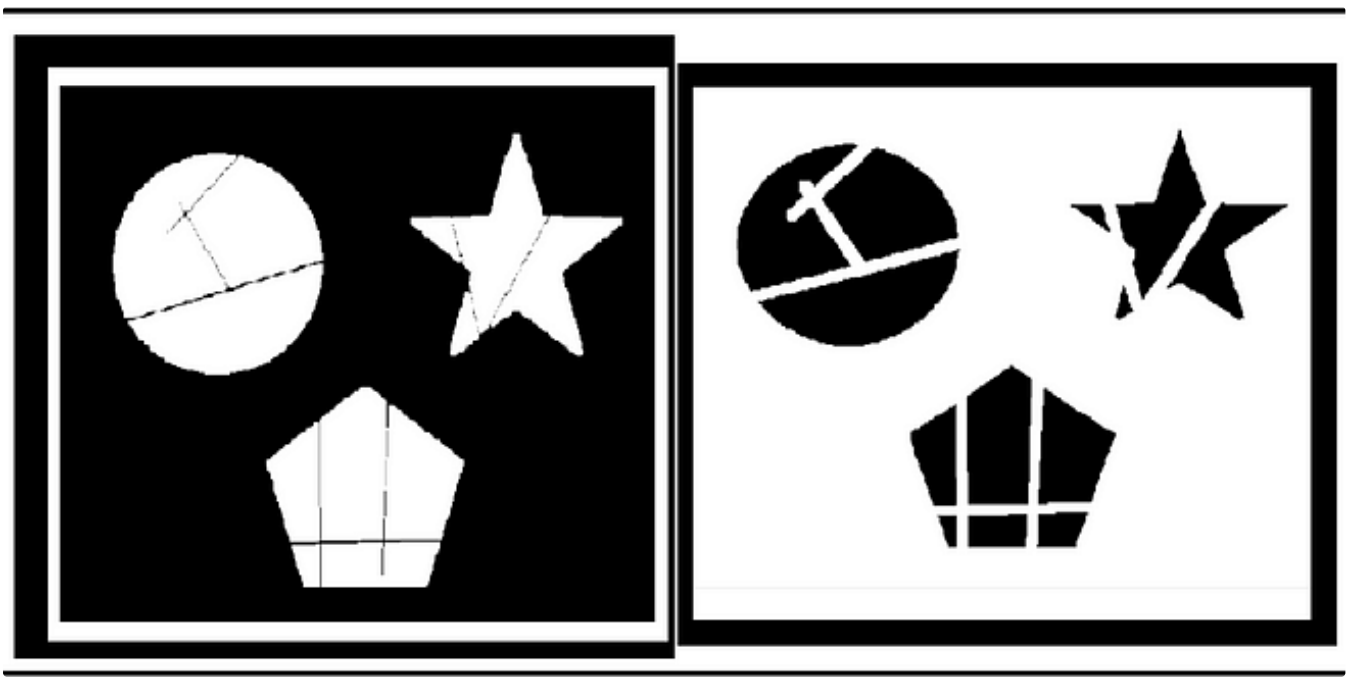
Nimrita Koul

Image Processing using OpenCV—Python

OpenCV

20 min read · Dec 20, 2023

56



Sasani S. Perera

OpenCV: Morphological Dilation and Erosion

Morphological operations are like magic tools for images. These operations can make images clearer, highlight important parts, or even...

6 min read · Aug 16, 2023

21

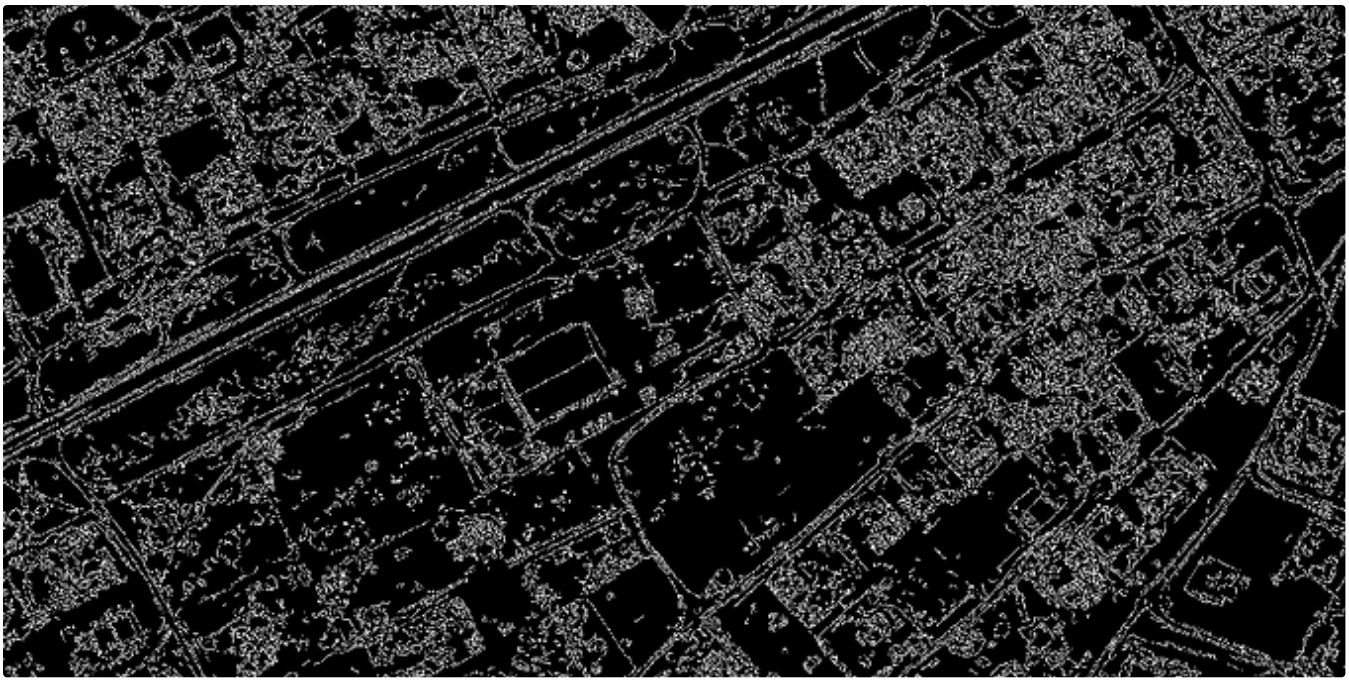
btd

Python Image Processing Toolbox: Essential Libraries and Packages

Image processing in Python is a vast and diverse field that involves manipulating and analyzing images using various techniques and...

· 3 min read · Nov 14, 2023

10



deepthipatric

Canny Edge Detection Using Python Libraries for Spatial Imagery Analysis

The Canny Edge Detection algorithm has been around for edge detection in images in image processing for a considerable period of time. In...

3 min read · Aug 8, 2023

19

See more recommendations