

Open in app ↗

Medium

🔍 Search



Image Courtesy: torange.biz

Introduction to Digital Image Processing in Python



Yedhu Krishnan · [Follow](#)

Published in Analytics Vidhya

3 min read · Sep 29, 2019

If you haven't read the first post in the series, read it here: [Lessons on Digital Image Processing \(#1\)](#).

In this second part, let's write some actual code. We will be writing all our code in Python.

Python makes processing and manipulating images very easy. [NumPy](#) and [SciPy](#) packages available for Python help us to perform scientific computing, mainly operations on matrices, which is of interest to us. We already saw images are just matrices in computers.

The easiest way to install SciPy and NumPy in a Linux machine is using Python package manager `pip`.

```
sudo apt-get install python3-pip
sudo pip3 install numpy scipy
```

Let's install two more packages which are necessary to load and display an image in Python.

```
sudo pip3 install imageio matplotlib
```

Now, to load and display an image, all you need is a few lines of code. Make sure you have an image file in the same directory where your code is residing.

```
1  import imageio
2  import matplotlib.pyplot as plt
3
4  # Load the image
5  image = imageio.imread('color_flower.jpg')
6
7  # Display the image
8  plt.imshow(image)
9  plt.show()
```

That's all. The code is self-explanatory. I have added necessary comments to help you understand.

We have a color image. How do we create a greyscale image from it?

We already know a color image has three values in each pixel positions for representing red, green and blue intensity values. In a greyscale image, there is only one value. All we need to do is to convert these three values into a single value.

How do we do that?

Just take the average of all the three values!

$$\text{grey pixel value} = (\text{red} + \text{green} + \text{blue}) / 3$$

Let us implement this in Python. We use a NumPy array to store the image in Python. NumPy provides a convenient function to take the average or mean on any axis. In the case of an RGB image, axis = 0 is row-wise, axis = 1 is column-wise and axis = 2 is channel-wise. We need a channel-wise sum.

```
gray_image = image.mean(axis = 2)
```

This one line of code converts the color image to a grayscale image. It does the same thing as the following lines of code:

```
1  (row, col, _) = image.shape
2
3  # Define an empty array for grey image
4  grey_image = np.zeros((row, col))
5
6  for i in range(row):
7      for j in range(col):
8          # Store the average of R, G, B values
9          grey_image[i, j] = image[i, j].mean()
10
11  plt.imshow(grey_image, cmap='gray'); plt.show()
```


The above code is much slower than the previous version. However, understanding this is important if you are new to this.

We just created the grey version of the image. Now, we can convert this into a binary image. A grey image pixel values range from 0 to 255. In binary, it can take only two values. In some representations, we use 0 and 1. Here, we'll use 0 (for black) and 255 (for white).

How do we convert a grey image into a binary image? The usual method is to set a threshold value τ . If the pixel value is above the threshold, we'll set it to 255. If it is below, we'll set it to 0. Here, we can take the threshold as the median grey value, which is 128.

Here is the extended version of the code.

```
1  binary_image = np.zeros((row, col))
2
3  for i in range(row):
4      for j in range(col):
5          if grey_image[i, j] <= 128:
6              pixel = 0
7          else:
8              pixel = 255
9
10         binary_image[i, j] = pixel
11
12  plt.imshow(binary_image, cmap='gray'); plt.show()
```

grey_to_binary.py hosted with ❤ by GitHub

[view raw](#)

And the short version is just two lines:

```
binary_image = grey_image.copy()
binary_image[binary_image <= 128] = 0
binary_image[binary_image > 128] = 255
```

That is it! NumPy takes care of updating all the pixel values to 0 or 255 depending on whether they are above or below the threshold.

Now we know about color, greyscale and binary images, and we know how to convert a color image to grey and then to binary. We will see more image processing

techniques and python code in the upcoming posts.

For now, please share your feedback and opinion about these posts.

Python

Image Processing

Numpy

Digital Image Processing



Follow

Written by Yedhu Krishnan

117 Followers · Writer for Analytics Vidhya

A reader and a programmer.

More from Yedhu Krishnan and Analytics Vidhya



Yedhu Krishnan

Image Manipulation in Python

If you haven't read the last post in the series, read it here: [The R, G, and B in an Image](#).

Oct 28, 2022 🖱 75



 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...

Jan 27, 2020 🖱 2.7K 💬 29




 Harikrishnan N B in Analytics Vidhya

Confusion Matrix, Accuracy, Precision, Recall, F1 Score

Binary Classification Metric

Dec 10, 2019  1.1K  6



 Yedhu Krishnan in Analytics Vidhya

Lessons on Digital Image Processing (#1)

This is the first one of a series of posts I am planning to write about image processing.

Mar 14, 2018  172



[See all from Yedhu Krishnan](#)

[See all from Analytics Vidhya](#)

Recommended from Medium

Amazon.com

Software Development Engineer

Seattle, WA

Mar. 2020 – May 2021

- Developed Amazon checkout and payment services to handle traffic of 10 Million daily global transactions
- Integrated Iframes for credit cards and bank accounts to secure 80% of all consumer traffic and prevent CSRF, cross-site scripting, and cookie-jacking
- Led Your Transactions implementation for JavaScript front-end framework to showcase consumer transactions and reduce call center costs by \$25 Million
- Recovered Saudi Arabia checkout failure impacting 4000+ customers due to incorrect GET form redirection

Projects

NinjaPrep.io (React)

- Platform to offer coding problem practice with built in code editor and written + video solutions in React
- Utilized Nginx to reverse proxy IP address on Digital Ocean hosts
- Developed using Styled-Components for 95% CSS styling to ensure proper CSS scoping
- Implemented Docker with Seccomp to safely run user submitted code with < 2.2s runtime

HeatMap (JavaScript)

- Visualized Google Takeout location data of location history using Google Maps API and Google Maps heatmap code with React
- Included local file system storage to reliably handle 5mb of location history data
- Implemented Express to include routing between pages and jQuery to parse Google Map and implement heatmap overlay



Alexander Nguyen in Level Up Coding

The resume that got a software engineer a \$300,000 job at Google.

1-page. Well-formatted.



Jun 1



24K



483





Abhishek Jain

Salt and pepper noise and how to remove it in machine learning

What is Salt and Pepper noise ?

Jun 16 🖱️ 2

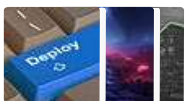


Lists



Coding & Development

11 stories · 865 saves



Predictive Modeling w/ Python

20 stories · 1616 saves



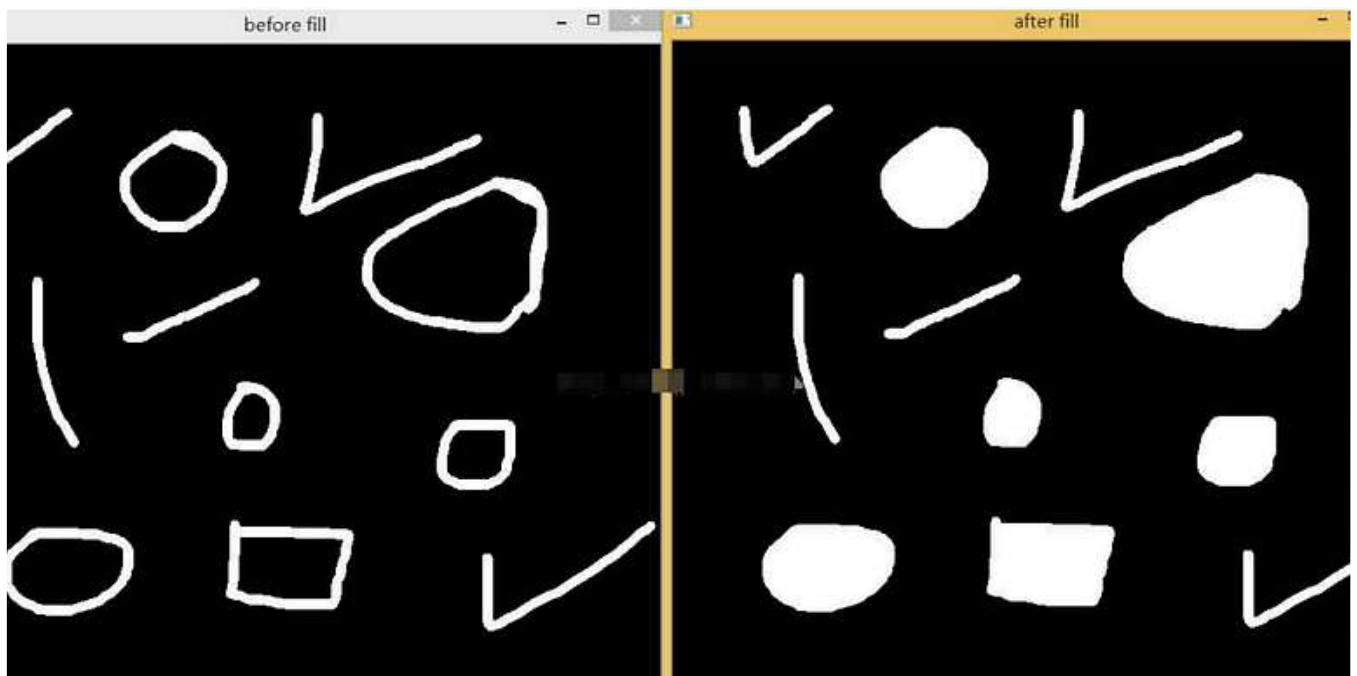
Practical Guides to Machine Learning


10 stories · 1972 saves



ChatGPT

21 stories · 848 saves



 PointCloud-Slam-Image-Web3 in Python & Other

Hole Filling of Opencv Binary Image use floodFill

Sometimes, we need to fill the image with holes, and the general idea is to use the findcontours function to find the outermost outline and...

✦ Jun 27



 Shaw Talebi in Towards Data Science

5 AI Projects You Can Build This Weekend (with Python)

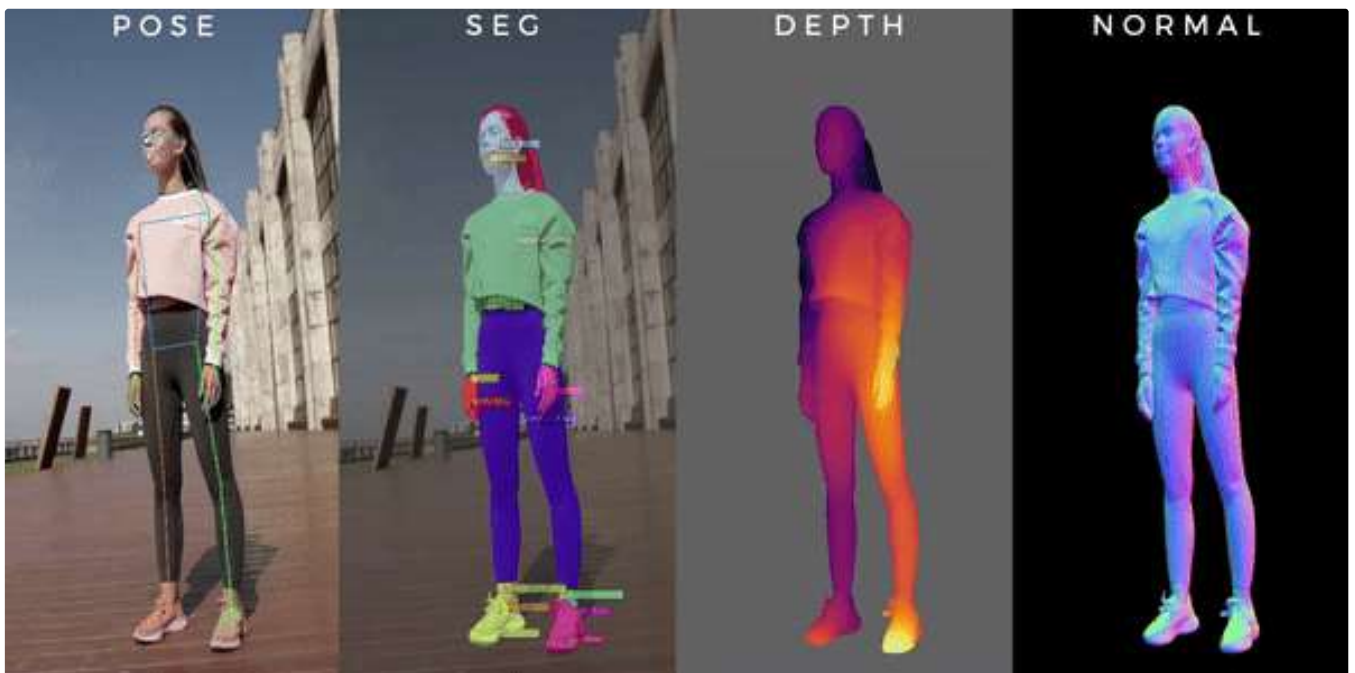
From beginner-friendly to advanced



J.

Parallel Image Processing with Python and CUDA

🌟 23h ago



 AI Papers Academy

Sapiens by Meta AI: Foundation for Human Vision Models

In this post we dive into Sapiens, a new family of computer vision models by Meta AI that show remarkable advancement in human-centric...

See more recommendations