Image Search Engine
Resource Guide

Adrian Rosebrock

Image Search Engine: Resource Guide

Share this Guide

Do you know someone who is interested in computer vision and OpenCV? Please, feel free to share this guide with them. Just send them this link:

http://www.pyimagesearch.com/resources/

Copyright © 2016 Adrian Rosebrock, All Rights Reserved
Version 1.4, November 2016

Table of Contents

Introduction 4

Books 5

My Books & Courses 5

Beginner Books 6

Textbooks 7

Conferences 8

Python Libraries 9

NumPy 9

SciPy 9

matplotlib 9

PIL and Pillow 9

OpenCV 10

SimpleCV 10

mahotas 10

scikit-learn 10

scikit-image 10

ilastik 11

pprocess 11

h5py 11

Connect 12

Introduction

Hello! My name is Adrian Rosebrock from PyImageSearch.com. Thanks for downloading this ***Image Search Engine Resource Guide***.

A little bit about myself: I'm an entrepreneur who has launched two successful image search engines: ID My Pill, an iPhone app and API that identifies your prescription pills in the snap of a smartphone's camera, and Chic Engine, a fashion search engine for the iPhone. Previously, my company ShiftyBits, LLC. has consulted with the National Cancer Institute to develop image processing and machine learning algorithms to automatically analyze breast histology images for cancer risk factors.

I have a Ph.D in computer science, with a focus in computer vision and machine learning, from the University of Maryland, Baltimore County where I spent three and a half years studying. I graduated in May of 2014.

I'm here to share the tips, tricks, and hacks I've learned in the past 8 years working in the startup field and building computer vision, machine learning, and image search engines.

After reading this guide, I would be interested to hear what you thought of it. Did you try any of the books? Did you look download any of the Python packages? Please send me an email and let me know at adrian@pyimagesearch.com or you can visit my website at www.PyImageSearch.com and leave a comment. I look forward to hearing from you soon!

-Adrian Rosebrock

I can teach you computer vision...*in a single weekend.*

A single weekend?

I know, it sounds crazy.

But my book, *Practical Python and OpenCV + Case Studies* is your **guaranteed quick start guide** to learning the fundamentals of computer vision and image processing using Python and OpenCV.

So if you're ready to become an **OpenCV ninja in only a *single weekend***, just click the button below. You'll **master computer vision** in no time, I guarantee it.

I want to be a computer vision ninja

Books

Back when I was a Ph.D student, I've had to read a ton of books and papers. Some of these were reference books, some were very technical, and others simply gave a high level overview of computer vision. Having these books, whether in physical or PDF form, is invaluable -- I could quickly pull them open and get the information I needed.

While there aren't any substantial books on building image search engines, you can start by reading about *computer vision* - a larger topic that includes processing, analyzing, and understanding the content of images. Having a strong foundation of computer vision (or at least being familiar with the concept computer vision) will ***dramatically*** help you build image search engines of your own.

Of course, having an understanding of computer vision is not a requirement. If you are new to the field and don't have any experience, that's okay too. I like to create examples that are very hands on, that let you start building image search engines immediately, without getting lost in the details.

My Books & Courses

Besides regularly publishing computer vision tutorials, articles, and posts on the PyImageSearch blog, I have also authored a book, *Practical Python and OpenCV + Case Studies*, as well as launched a highly successful computer vision course: *PyImageSearch Gurus*.

Practical Python and OpenCV + Case Studies

Practical Python and OpenCV + Case Studies is broken down into two parts. Part I covers the basics of computer vision, image processing and OpenCV.

Part II applies the fundamentals of computer vision you learned from Part I to solve actual real-world problems such as ***face detection in images, videos, & webcam streams***, ***object tracking***, and ***book cover identification*** utilizing SIFT and keypoint matching.

If you are interested in learning more about *Practical Python and OpenCV + Case Studies*, **just click here**.

PyImageSearch Gurus

PyImageSearch Gurus is a *course* and *community* designed to take you from computer vision beginner to guru.

Because let's face it. Learning computer vision and OpenCV is *hard*. **But not anymore**.

You see, PyImageSearch Gurus has only one goal -- to make *developers, programmers, researchers, and students like yourself awesome at solving real-world computer vision problems*.

Whether you're just getting started in computer vision or you're a seasoned computer vision expert, PyImageSearch Gurus has **kick-ass tutorials, tons of code examples, and a dedicated community** that is **guaranteed to level-up your computer vision skills.**

Inside the PyImageSearch Gurus course we take an *in-depth dive* into topics such as:

- Computer vision and OpenCV basics
- Building your own custom object detector
- Automatic license plate recognition
- Deep learning
- Content-Based Image Retrieval (i.e. image search engines)
- Image classification
- Hadoop and big data tools for computer vision
- Image descriptors
- Building computer vision apps for your mobile device
- Hand gesture recognition.

To learn more about the PyImageSearch Gurus course, **just click here.**

Beginner Books

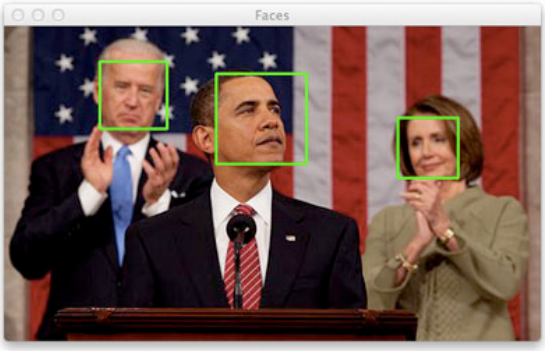
I recommend these books for someone just starting out in the field of computer vision and image search engines:

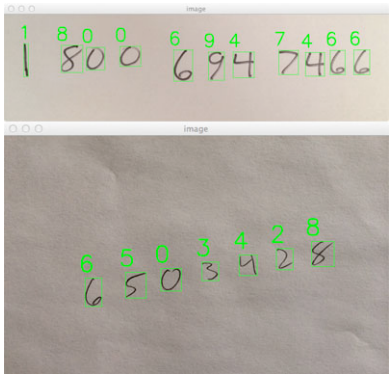
- Programming Computer Vision with Python: Tools and algorithms for analyzing images *by Jan Erik Solem*
- Practical Computer Vision with SimpleCV : The Simple Way to Make Technology See *by Kurt Demaagd, Anthony Oliver, Nathan Oostendorp, and Katherine Scott*
- OpenCV Computer Vision with Python *by Joseph Howse*
- Learning OpenCV: Computer Vision with the OpenCV Library *by Gary Bradski and Adrian Kaehler*
- OpenCV 2 Computer Vision Application Programming Cookbook *by Robert Laganière*
- Mastering OpenCV with Practical Computer Vision Projects *by Daniel Lélis Baggio, Shervin Emami, David Millán Escrivá, Khvedchenia Ievgen, Jasonl Saragih, and Roy Shilkrot*
- SciPy and NumPy: An Overview for Developers *by Eli Bressert*

Textbooks

If you already have an understanding of computer vision, whether from reading previous works, taking a few courses in college, or simply working in the field already, then these books will help give you a deeper understanding of computer vision:

- Computer Vision: A Modern Approach (2nd Edition) *by David A. Forsyth and Jean Ponce*
- Computer Vision *by Linda G. Shapiro and George C. Stockman*
- Computer Vision: Algorithms and Applications *by Richard Szeliski*
- Algorithms for Image Processing and Computer Vision *by J. R. Parker*
- Computer Vision: Models, Learning, and Inference *by Dr Simon J. D. Prince*
- Computer and Machine Vision, Fourth Edition: Theory, Algorithms, Practicalities *by E. R. Davies*





Don't be fooled. Computer vision isn't magic - you can learn how to solve *real-world computer vision problems.*

Do you want to **detect faces in images?** No problem.

Recognize handwriting in photos? I've got you covered.

Recognize the covers of books? Let me show you.

What about apply machine learning to **classify image contents?** You'll be up-to-speed in no time.

The bottom line is that *Practical Python and OpenCV* will jumpstart your computer vision education. You'll learn the secrets the pros use...*and become a pro yourself.*

Become an OpenCV Pro

Conferences

If you have experience in the computer vision or Content Based Image Retrieval (CBIR) field, then you might want to consider submitting original research papers to the following conferences:

- CVPR - Computer Vision and Pattern Recognition
- ICCV - International Conference on Computer Vision
- ECCV - European Conference on Computer Vision
- BMVC - British Machine Vision Conference
- ICIP - IEEE International Conference on Image Processing

Python Libraries

When I first became interested in computer vision and image search engines over eight years ago, I had no idea where to start. I didn't know which language to use, I didn't know which libraries to install, and the libraries I found I didn't know how to use. **I WISH** there had been a list like this one, detailing the best Python libraries to use for image processing, computer vision, and image search engines.

This list is by no means complete or exhaustive. It's just my favorite Python libraries that I use each and everyday for computer vision and image search engines. If you think that I've left an important one out, please leave me an email at adrian@pyimagesearch.com.

NumPy
NumPy is a library for the Python programming language that (among other things) provides support for large, multi-dimensional arrays. Why is that important? Using NumPy, we can express images as multi-dimensional arrays. Representing images as NumPy arrays is not only computational and resource efficient, but many other image processing and machine learning libraries use NumPy array representations as well. Furthermore, by using NumPy's built-in high-level mathematical functions, we can quickly perform numerical analysis on an image.

SciPy
Going hand-in-hand with NumPy, we also have SciPy. SciPy adds further support for scientific and technical computing. One of my favorite sub-packages of SciPy is the spatial package which includes a vast amount of distance functions and a kd-tree implementation. Why are distance functions important? When we "describe" an image, we perform feature extraction. Normally after feature extraction an image is represented by a vector (a list) of numbers. In order to compare two images, we rely on distance functions, such as the Euclidean distance. To compare two arbitrary feature vectors, we simply compute the distance between their feature vectors. In the case of the Euclidean distance, the smaller the distance the more "similar" the two images are.

matplotlib
Simply put, matplotlib is a plotting library. If you've ever used MATLAB before, you'll probably feel very comfortable in the matplotlib environment. When analyzing images, we'll make use of matplotlib, whether plotting the overall accuracy of search systems or simply viewing the image itself, matplotlib is a great tool to have in your toolbox.

PIL and Pillow
These two packages are good and what they do: simple image manipulations, such as resizing, rotation, etc. If you need to do some quick and dirty image manipulations definitely check out PIL and Pillow, but if you're serious about learning about image processing, computer vision, and image search engines, I would *highly* recommend that you spend your time playing with OpenCV and SimpleCV instead.

OpenCV
If NumPy's main goal is large, efficient, multi-dimensional array representations, then, by far, the main goal of OpenCV is real-time image processing. This library has been around since 1999, but it wasn't until the 2.0 release in 2009 did we see the incredible NumPy support. The library itself is written in C/C++, but Python bindings are provided when running the installer. OpenCV is hands down my favorite computer vision library, but it does have a learning curve. Be prepared to spend a fair amount of time learning the intricacies of the library and browsing the docs (which have gotten substantially better now that NumPy support has been added). If you are still testing the computer vision waters, you might want to check out the SimpleCV library mentioned below, which has a substantially smaller learning curve.

SimpleCV
The goal of SimpleCV is to get you involved in image processing and computer vision as soon as possible. And they do a great job at it. The learning curve is substantially smaller than that of OpenCV, and as their tagline says, "it's computer vision made easy". That all said, because the learning curve is smaller, you don't have access to as many of the raw, powerful techniques supplied by OpenCV. If you're just testing the waters, definitely try this library out.

mahotas
Mahotas, just as OpenCV and SimpleCV, rely on NumPy arrays. Much of the functionality implemented in Mahotas can be found in OpenCV and/or SimpleCV, but in some cases, the Mahotas interface is just easier to use, especially when it comes to their features package.

scikit-learn
Alright, you got me, Scikit-learn isn't an image processing or computer vision library — it's a machine learning library. That said, you can't have advanced computer vision techniques without some sort of machine learning, whether it be clustering, vector quantization, classification models, etc. Scikit-learn also includes a handful of image feature extraction functions as well.

scikit-image
Scikit-image is fantastic, but you have to know what you are doing to effectively use this library -- and I don't mean this in a "there is a steep learning curve" type of way. The learning curve is actually quite low, especially if you check out their gallery. The algorithms included in scikit-image (I would argue) follow closer to the state-of-the-art in computer vision. New algorithms right from academic papers can be found in scikit-image, but in order to (effectively) use these algorithms, you need to have developed some rigor and understanding in the computer vision field. If you already have some experience in computer vision and image processing, definitely check out scikit-image; otherwise, I would continue working with OpenCV and SimpleCV to start.

ilastik
I'll be honest. I've never used ilastik. But through my experiences at computer vision conferences, I've met a fair amount of people who do, so I felt compelled to put it in this list. Ilastik is mainly for image segmentation and classification and is especially geared towards the scientific community.

pprocess
Extracting features from images is inherently a parallelizable task. You can reduce the amount of time it takes to extract features from an entire dataset by using a multithreading/multitasking library. My favorite is pprocess, due to the simple nature I need it for, but you can use your favorite.

h5py
The h5py library is the de-facto standard in Python to store large numerical datasets. The best part? It provides support for NumPy arrays. So, if you have a large dataset represented as a NumPy array, and it won't fit into memory, or if you want efficient, persistent storage of NumPy arrays, then h5py is the way to go. One of my favorite techniques is to store my extracted features in a h5py dataset and then apply scikit-learn's MiniBatchKMeans to cluster the features. The entire dataset never has to be entirely loaded off disk at once and the memory footprint is extremely small, even for thousands of feature vectors.

Learning computer vision and OpenCV used to be ***hard.*** **But not anymore.**

PyImageSearch Gurus is a ***course*** and ***community*** designed to take you from computer vision *beginner* to *expert*. Guaranteed.

To find out how, click the button below to **take a (FREE) tour of the course & get 10 sample lessons:**

Claim your 10 (FREE) sample lessons!

Connect

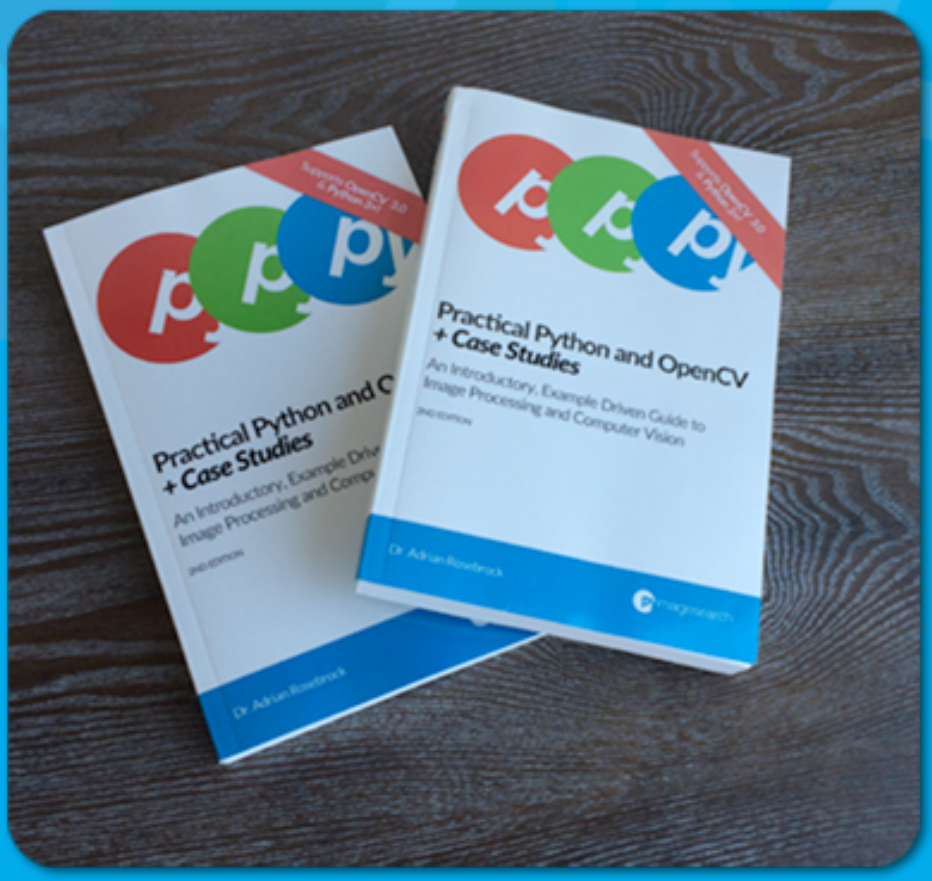
Want to find me online? Look no further:

Website:	http://www.PyImageSearch.com
Email:	adrian@pyimagesearch.com
Twitter:	@PyImageSearch
Facebook:	PyImageSearch
Google+:	+AdrianRosebrock
LinkedIn:	Adrian Rosebrock

For more information, please visit http://www.PyImageSearch.com

Hardcopy editions of *Practical Python and OpenCV + Case Studies.*

Now available.



I want to claim my hardcopy edition!