

# OpenCV (C++ vs Python) vs MATLAB for Computer vision

## Learn OpenCV

OCTOBER 30, 2015 BY SATYA MALLICK ([HTTPS://WWW.LEARNOPENCV.COM/AUTHOR/SPMALLICK/](https://www.learnopencv.com/author/spmallick/)).

**Tools Do Not**  
*A Craftsman*  
**MAKE**

([/wp-content/uploads/2015/10/tools-do-not-a-craftsman-make.jpg](#))

We often confuse our tools for our craft. Tools help you practice your craft, but they do not make you a good

craftsman. A good craftsman has many different tools in her pocket, and she judiciously uses the one that is appropriate for the job. She is not married to the tools. She is married to her craft.

Learn OpenCV

I am often amused by wars over programming languages. People have very strong opinions about which one is better. The truth is that a programmer should choose the language appropriate for their task. Want to rapidly build a web app ? Try Ruby on Rails or Django. Want to write high performance code for an embedded device. Try C.

In computer vision, we are faced with similar choices. Which tool should a computer vision engineer / programmer learn — OpenCV using C++, OpenCV using Python, or MATLAB ? The good news is that today we have a few options to choose from! A decade back there were no good libraries for computer vision. If you wanted to learn, you just picked up a book and started coding your own mini library of computer vision algorithms. Thankfully, things are much better now.

If you are a beginner, I would suggest using the path of least resistance and picking a tool you are familiar with. If you are a python programmer, use OpenCV with Python. If you know C++, use C++ with OpenCV. The same holds true for MATLAB. That said in a few months you will no longer be a beginner. You may be looking to apply this newly acquired knowledge to the real world. You could be thinking of a new side project, or looking for a new job in this area. It pays ( sometimes literally ) to know how to make your choices. So here are my thoughts.

## MATLAB for Computer Vision

Until recently Computer Vision was a research area in its infancy. People who worked in Computer Vision were

mostly in the academia or research labs. Their tool of choice was MATLAB, and for the longest time OpenCV paled in comparison to what MATLAB and its community had to offer. In the last 7 years, the tides have turned. Here are some of the pros of using MATLAB.

## Why Should You Use MATLAB for Computer Vision : The Pros

1. **Powerful matrix library** : Not only do we treat an image as a multi-dimensional matrix in computer vision, we use a heavy dose of linear algebra in various algorithms. MATLAB's linear algebra routines are very powerful and blazingly fast ( when used correctly ). Once we needed to solve a large sparse linear system as part of an algorithm. It was a one line code in MATLAB — something like  $X = A \setminus b$  . We were assuming that our C++ implementation will be at least 3x faster than the MATLAB implementation. The first version of our C++ implementation turned out to be slower than the MATLAB version! It took us days to match the performance of the MATLAB routine. So in MATLAB a single operator like backslash (  $\setminus$  ) is sometimes a collection of powerful algorithms that MATLAB appropriately chooses for you.
2. **Toolboxes** : What ever your needs, there is a toolbox for that. They have an **image processing** toolbox, a **computer vision** toolbox, and a **statistical and machine learning** one that provide implementations of a wide variety of very useful algorithms. The functions usually provide a clean and obvious interface. Many computer vision problems are often set up as optimization problems. You are trying to maximize or minimize some objective functions under certain constraints. The **optimization toolbox** in MATLAB provides excellent implementations of many optimization algorithms.

- 3. Visualization and debugging tools** : One of the joys of using MATLAB is that writing code, visualizing results, and debugging happens in one integrated environment. The environment makes you extremely productive.
- 4. Works with OpenCV** : You can interface with OpenCV using MATLAB's OpenCV Interface (<http://www.mathworks.com/matlabcentral/fileexchange/47953-computer-vision-system-toolbox-opencv-interface>).
- 5. Great documentation** : Even staunch supporters of OpenCV admit that its documentation sucks. In contrast, MATLAB comes with great documentation and examples that are easily accessible within the IDE. Great documentation alone can make coding in MATLAB 2x faster than using OpenCV. MATLAB functions are also better designed compared to OpenCV. E.g. compare this simple code for displaying an image in OpenCV and MATLAB.

```
1 // MATLAB
2 imshow(im);
3
4 // OpenCV
5 imshow("myWindow", im);
```

Most of the time, I just want to display an image. Call it “Image 1” for God’s sake instead of forcing me to type the window name ( “myWindow” ) over and over again.

- 6. Large research community** : As I have mentioned earlier, MATLAB is extremely popular in the academia. Latest research demos are often shared as MATLAB code. If you want to be on the cutting edge, you should be able to read MATLAB code.

# Why You Should Not MATLAB for Computer Vision : The Cons

1. **Cost is HUGE** : MATLAB is hideously expensive. Let us start with basic MATLAB (\$2,150) and throw in the computer vision toolbox (\$1,350). But the computer vision toolbox requires the image processing toolbox (\$1000). Add optimization (\$1,350) and machine learning toolboxes (\$1000). That brings your grand total to \$6850! Ok, now you have built your application and want to deploy it. Well, you gotta buy the MATLAB compiler (\$4,250). Oh you want it for two different operating systems ? That would be another \$4,250. MATLAB makes sense if you get it for a discounted price through your University, or your company has a license.
2. **Learning curve** : MATLAB is a matrix engine. There is a MATLAB way to write code which is different from general purpose programming languages like C++ or Python. And if you do not write code the MATLAB way, your code can be extremely slow.
3. **Slower runtime** : A typical MATLAB program runs many times slower than a C++ program. Built-in MATLAB routines can be very fast, but the code you write in MATLAB will usually run much slower. Often times people end up coding computationally intensive parts in C and integrating it with MATLAB code using [mex](http://www.mathworks.com/help/matlab/ref/mex.html) (<http://www.mathworks.com/help/matlab/ref/mex.html>).

## OpenCV (C++) for Computer Vision

OpenCV is my primary tool for developing computer vision applications. I have prototyped a lot in MATLAB,

but except on one occasion the production version has always been OpenCV based. Let's look at the pros and cons.

## Learn OpenCV

### Why should you use OpenCV (C++) for Computer Vision : The Pros

1. **It's free!** : Large parts of OpenCV are free — i.e. free as in beer, and free as in speech! You can use OpenCV freely in your commercial application, and you can view the source and fix issues if needed. You do not have to open source your project if you use OpenCV.
2. **Huge optimized library** : The collection of algorithms available in OpenCV dwarfs everything out there. The library is also optimized for performance. With OpenCV 3, you can use the [Transparent API \(/opencv-transparent-api/\)](#) to easily use OpenCL compliant devices (e.g. GPU) on your machine. Many algorithms have a CUDA implementation.
3. **Platforms and devices** : You can obviously use OpenCV in your desktop app or as the backend of your web application. Because of its focus on performance OpenCV (C/C++) is the vision library of choice in many embedded vision applications and mobile apps.
4. **Big community** : There is a big community of developers (47,000 or so) that use and support OpenCV. It has been downloaded more than 9 Million times. Unlike the MATLAB community that consists of

researchers, the OpenCV community is a mix of people from many fields and industries. The OpenCV development is funded by companies like Intel, AMD & Google. Needless to say, this blog is part of this vibrant community of people who are trying to help each other.

# Why should you not use OpenCV (C++) for Computer Vision : The Cons

1. **Difficult for beginners** : If you do not have programming experience in C++, using the OpenCV (C++) will be daunting. You are probably better off trying to use the Python instead.
2. **Weak documentation** : OpenCV documentation is bad. Sometimes you need to have a good understanding of the algorithm, and actually read the paper, because the documentation does not always explain what the parameters mean and how they effect the outcome. The documentation does not always come with sample code, and that makes it harder to understand. The sample code that comes with OpenCV, though very useful, is also not very well documented either. Let's just say this is work in progress, and people are chipping in to make it better.
3. **Small machine learning library** : A computer vision engineer frequently needs many machine learning routines. OpenCV has a small set of machine learning algorithms compared to the choices available when you are using OpenCV (Python).
4. **Visualization and debugging** : Debugging and visualizing is hard in any C++ environment. This is especially true if you are coming up with a new algorithm from scratch. Occasionally I dump data to disk, and analyze it using MATLAB.

## OpenCV (Python) for Computer Vision

I believe python bindings for OpenCV have contributed quite a bit to its popularity. It is an excellent choice for

learning Computer Vision, and is good enough for a wide variety of real world applications. Let's look at the pros and cons.

## Learn OpenCV

# Why should you use OpenCV (Python) for Computer Vision : The Pros

1. **Ease of use** : If you are a python programmer, using OpenCV (Python) would be very easy. Python is an easy language to learn ( especially compared to C++ ). It is also an excellent first language to learn.
2. **Python has become the language of scientific computing** : A few years back MATLAB was called the language of scientific computing. But now, with OpenCV, numpy (<http://www.numpy.org/>), scipy (<http://www.scipy.org/>), scikit-learn (<http://scikit-learn.org/>), and matplotlib (<http://matplotlib.org/>) Python provides a powerful environment for learning and experimenting with Computer Vision and Machine Learning.
3. **Visualization and debugging** : When using OpenCV (Python) you have access to a huge number of libraries written for Python. Visualization using matplotlib (<http://matplotlib.org/>) is about as good as MATLAB. I find debugging code in Python easier than in C++, but it does not quite match the super-easiness of MATLAB.
4. **Building web backend** : Python is also a popular language for building websites. Frameworks like Django (<https://www.djangoproject.com/>), Web2py (<http://www.web2py.com/>), and Flask (<http://flask.pocoo.org/>) allow you to quickly put together web apps. It is very easy to use OpenCV



(Python) along with these web frameworks. E.g. read this [tutorial \(/turn-your-opencv-code-into-a-web-api-in-under-10-minutes-part-1/\)](#) that explains how to turn your OpenCV code into a web api in under 10 minutes ([/turn-your-opencv-code-into-a-web-api-in-under-10-minutes-part-1/](#)).

## Learn OpenCV

# Why should you not use OpenCV (Python) for Computer Vision : The Cons

1. **Weak Documentation** : As I mentioned before, OpenCV(C++) documentation is not very good. But the python documentation is even worse. A novice user is left guessing how to use certain functions. E.g. at the time of writing this post it is almost impossible to find the python documentation for **cv2.Subdiv2D**. You will also find fewer tutorials for OpenCV (Python).
2. **Lack of support** : Companies that support OpenCV ( Intel, AMD, NVidia etc. ) have a dog in the fight when it comes to the C++ version of OpenCV. They want you to use OpenCV and to buy their hardware (CPUs/GPUs etc.) to run these algorithms. But OpenCV (Python) seems to be the proverbial red-headed step child that does not seem to get their attention. E.g. I have not yet figured out if the [Transparent API \(/opencv-transparent-api/\)](#) is supported in OpenCV ( Python ).
3. **Slower run time** : Compared to C++, your programs in Python will typically run slower. To add an extra punch you can use the GPU ( using CUDA or OpenCL ) in OpenCV (C++) and have code that runs 10x faster than the Python implementation.
4. **OpenCV is written in C/C++** : One of the great benefits of an open source library is your ability to modify them to suit your needs. If you want to modify OpenCV, you have to modify the C/C++ source.

# Summary

## Learn OpenCV

As engineers and craftsmen we need to use the right tool for the right job. Eventually, you have to learn all the tools in this trade. As a rule of thumb I use MATLAB / OpenCV (Python) for prototyping new algorithms, and use OpenCV (C++) in production.

## Subscribe

If you liked this article, please subscribe

(<https://bigvisionllc.leadpages.net/leadbox/143948b73f72a2%3A173c9390c346dc/5649050225344512/>) to our newsletter and receive a free

Computer Vision Resource

(<https://bigvisionllc.leadpages.net/leadbox/143948b73f72a2%3A173c9390c346dc/5649050225344512/>) guide.

In our newsletter we share OpenCV tutorials and examples written in C++/Python, and Computer Vision and Machine Learning algorithms and news.

**Subscribe Now**

(<https://bigvisionllc.leadpages.net/leadbox/143948b73f72a2%3A173c9390c346dc/5649050225344512/>)

8 Comments

Learn OpenCV

 Login ▾

 Recommend 8

 Share

Sort by Best ▾



Join the discussion...

## Learn OpenCV

LOG IN WITH

OR SIGN UP WITH DISQUS 

Name



**chitra korimilli** • 5 months ago

Hi!

I'm a beginner and am trying to do a image comparison process and recognizing if there's a mismatch in them, coming from a stream of continuous images from the server. Which is better for this- matlab or opencv? (and if opencv, C++ or python ?)

Thank you in advance.

^ | v • Reply • Share ›



**Kushashwa Ravi Shrimali** → chitra korimilli • 3 months ago

It really depends on your target application. MATLAB code, as mentioned, can not be used on other systems if they don't have it installed or don't have the license to use it online.

As far as C++ and Python are considered, both are flexible, and since you are a beginner, you should surely go for Python way of doing it. It's easy to understand and for beginners, it's a great start.

Thanks!

^ | v • Reply • Share ›



**Abdulrhman Mhuob** • a year ago

what is the best ide for paython that can run the most open source project in github

^ | v • Reply • Share ›



**Sharma** • 2 years ago

Hi I am new in this field I want to make an android app for real time licence plate recognition system using opencv can you suggest me what should I use opencv (c++ or python, have experience in both c++ and python) or javacv. I dont even know if they all are compatible with android

or not.

^ | v • Reply • Share ›

# Learn OpenCV



**almost90degree** → Sharma • 2 years ago

opencv has an android sdk

^ | v • Reply • Share ›



**Simon** • 3 years ago

Great write-up - thanks! What are your thoughts on the gaps when using Octave instead of MATLAB? There are some open source functions out there eg. <http://www.peterkovesi.com/....> How far short do these fall of the pricey toolboxes?

^ | v • Reply • Share ›



**Satya Mallick** Mod → Simon • 3 years ago

Thanks Simon. I have used Octave very rarely and it was a few years back, so I am unable to comment objectively. That said, I would not be using MATLAB if I did not have access to it through a company I am consulting for and would definitely give Octave a try.

^ | v • Reply • Share ›



**Edwin Francois** • 2 years ago

Great article. I'm curious about your thoughts on this:

There are now libraries for GPU acceleration for Python such as NumbaPro and PyCUDA. My understanding is that you can usually get more performance using C++, but is that still true with GPU acceleration for Python?

^ | v • Reply • Share ›

## ALSO ON LEARN OPENCV

### Batch Normalization in Deep Networks

3 comments • 2 months ago



**mark aj** — NICE TUTORIAL!!

Avatar

### Support Vector Machines (SVM)

### Convex Hull using OpenCV in Python and C++


3 comments • a month ago



**Amit** — Is there any implementation through which we can find convex hull around set of n-dimension points.

### SVM using Scikit-Learn in Python




8 comments • 2 months ago

 **Stephen Meschke** — The kernel trick is pretty neat!  
Avatarhttps://uploads.disquscdn.c...

2 comments • 2 months ago

**Learn OpenCV**  
 **Gaurav Kumar Singh** — Amazing!!!! Kush....thanks for sharing..  
Avatar

---

 **Subscribe**  **Add Disqus to your site**Add DisqusAdd  **Disqus' Privacy Policy**Privacy PolicyPrivacy

COPYRIGHT © 2018 · BIG VISION LLC