AI Applications mini project 2 - OpenCV

Fadil Smajilbasic & Fabian Freitag

What is it used for?

OpenCV stands for Open Computer Vision. It offers a very exhaustive framework for everything related to computer vision. OpenCV can be used for very basic tasks like opening an Image or a Video in a python project and editing these pictures. One option it offers, that will be used a lot is to greyscale an Image, since many of the other resources provided with OpenCV require an image to be in greyscale. In our example we use OpenCV to detect a Face in an Image. This could be extended to use a Webcam and detect Faces in a livestream.

OpenCV includes a statistical machine learning library that contains:

- Boosting
- Decision tree learning
- Gradient boosting trees
- Expectation-maximization algorithm
- k-nearest neighbor algorithm
- Naive Bayes classifier
- Artificial neural networks
- Random forest
- Support vector machine (SVM)
- Deep neural networks (DNN)

How do you use it

To use openCV you have to install it using: `pip install opencv-python` And then import it in your python file using: `import cv2`

OpenCV can be used to display images, draw shapes over images (rectangles, circles, lines, ellipses, polygons, and text), apply filter to images (greyscale, color filter, blur,...)

Since videos are just a sequence of images, the same modifications can be done to videos as well.

The Learning curve

The documentation of OpenCV, which can be found here for the latest version, is not very useful. It contains all the classes methods, parameters and return values and in some cases there are even examples, but it is rather chaotic and browsing trough the documentation you feel like your are in a maze and lost. While making the example project and doing our research we often found ourselves searching for explanations or solutions on online forums like stackoverflow or articles that describe the usage of some method better than the documentation itself, and we feel like that should not be the case with a well known library as OpenCV.

It was hard to acclimate to the usage of the library since OpenCV does not implement error handling codes which, as you can imagine, makes it very hard do debug the code.

OpenCV compared to MatLab

The main advantage of OpenCV beside its functionality is the fact that it is open source and much faster in execution (the speed ratio reaches more than 80 in some cases). But of course OpenCV has some downsides. Matlab is more convenient in developing, having a better documentation, and it has a more sophisticated data presentation.

Ease of use

One of OpenCVs disadvantages is the ease of use. As an example lets try to display a picture. If we try to display a picture in MatLab we need two lines of code. One to load the image and one to display it. In OpenCV we need around 10-15 lines of code. This is because a lot of the "housekeeping" that is done in the background by MatLab has to be done manually with OpenCV.

Speed

On the other hand, since MatLab is such a high level scripting language, it is very slow compared to good code written with OpenCV. As an example, MatLab might typically be able to process about 3-4 frames of a video per second, while the same application written with OpenCV might be able to process around 30 frames per second, which definitely gives it an edge especially in real-time applications.

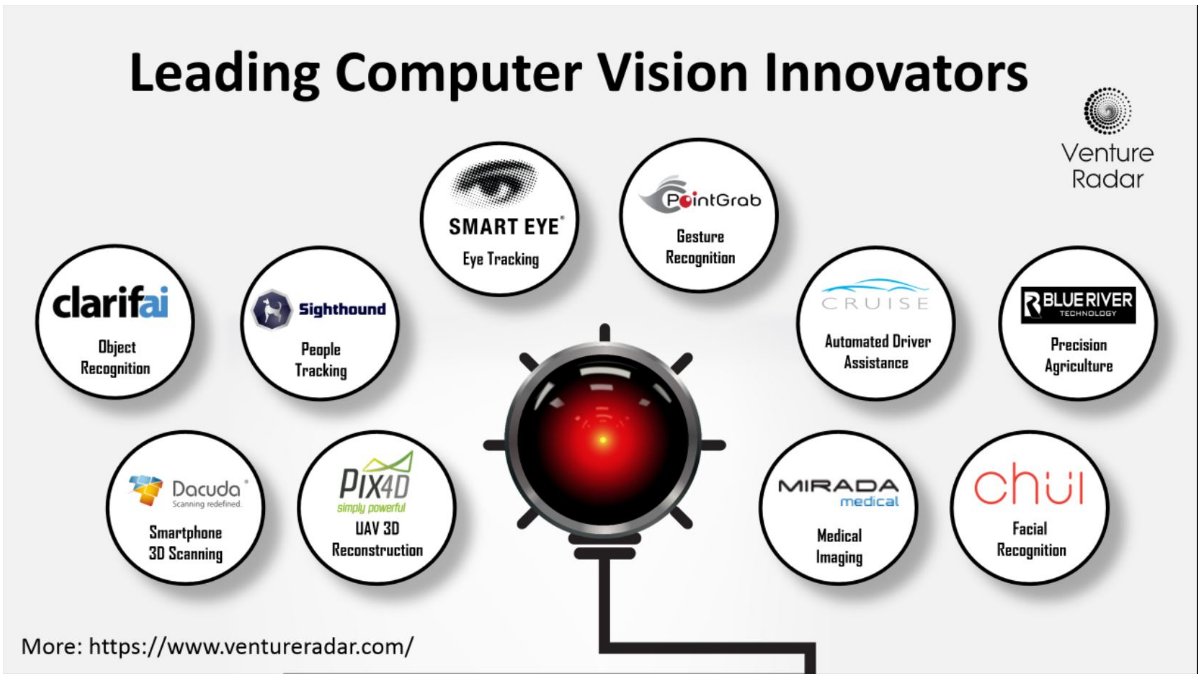
Resources needed

The same logic follows if we look at the resources needed to run an application. Because MatLab is is such a high level scripting language, it is inherently inefficient. Apparently it is by no means unheard of that MatLab uses over one Gigabyte of RAM, a comparable project with OpenCV only uses about 100 Megabytes of RAM, assuming your code does not have any Memory leaks.

In the end it is probably fair to say, that OpenCV has the strong potential to yield better results if you are willing to invest the time to learn where its pitfalls are, how to handle them and how to actually use it.

Developer adoption

OpenCV is well known to developers in general. The project initially released in June 2000 has since gained a lot of traction.



As seen in the picture above, OpenCV is utilized in many exciting applications such as object recognition and person tracking, 3D scanning, Automated driver assistance or medical imaging. The specific extend to which OpenCV is used in these applications is hard to track down, since while OpenCV is open source, applications that use the Library are themselves not open source.

OpenCV Licensing

Since the beginning of openCV, it operated under a BSD license. This License allowed everybody to use to use the library in any kind of project, be it a personal project, education, research or even commercial projects for free without any limitations. The Only problem with this license is, that if someone contributes code to the library, that they patented prior to this contribution, they could still sue someone for the commercial use of this particular code. To prevent this, OpenCV 4.5.0 and above, released in October 2020 uses an apache 2 license. While this change still allows the free use of OpenCV in any project, it also prevents contributors from suing anyone for using any part of the library in any kind of project.

Closing statements

We think that OpenCV is a very versatile and useful tool for programmers that have to work with images, whether they use the library for image manipulation or for object detection or other AI uses. With an improved documentation, this library would be the one to rule them all, but until then one has to either spend time searching the internet for answers or resort to a different library which is a shame.

Sources:

- Reference article
- OpenCV vs MATLAB article