OpenCV Basics

Lecture #2

January 27, 2020
... and so it begins

Intel's computer vision library: applications in calibration, stereo, segmentation, tracking, gesture, face and object recognition

Authors  Gary R Bradski, Vadim Pisarevsky
Publication date  2000/6/13
Conference  2013 IEEE Conference on Computer Vision and Pattern Recognition
Volume  2
Pages  2796-2796
Publisher  IEEE Computer Society
Description  Intel's Microcomputer Research Lab has been developing a highly optimized Computer Vision Library (CVLib) that automatically detects processor type and loads the appropriate MMX™ technology assembly tuned module for that processor. MMX optimized functions are from 2 to 8 times faster than optimized C functions. We will be demonstrating various algorithms supported by CVLib and handing out CDs containing the library. 1. Background ... Over the past year and a half, Intel1 has developed a computer vision library (CVLib) to support real time vision ...

Total citations  Cited by 101
Versions and Choices

• CS Department Machines – needs setup

    [ross@denver ~]$ pwd
    /s/parsons/a/fac/ross
    [ross@denver ~]$ python3
    Python 3.7.5 (default, Oct 17 2019, 12:09:47)
    [GCC 9.2.1 20190827 (Red Hat 9.2.1-1)] on linux
    Type "help", "copyright", "credits" or "license" for more information.
    >>> import cv2
    Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    ModuleNotFoundError: No module named 'cv2'
    >>>

• Your machine is your domain

Fair Disclosure: Some of you are more experienced with Python OpenCV environments. Advice and suggestions are for sharing!

I will talk to SNA about this.
If you have a Mac

- Visualize results with Matplotlib, Bokeh, Datashader, and Holoviews

Anaconda 2019.10 for macOS Installer

Python 3.7 version
Download
64-Bit Graphical Installer (654 MB)
64-Bit Command Line Installer (424 MB)

Python 2.7 version
Download
64-Bit Graphical Installer (637 MB)
64-Bit Command Line Installer (409 MB)
My Environment

• Here is how it looks now I am setup

```
[Juniper:code/opencvPython/tutorial00] ross% conda activate cv2
(cv2) [Juniper:code/opencvPython/tutorial00] ross% ipython
Python 3.8.1 | packaged by conda-forge | (default, Jan 5 2020, 20:40:51)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.11.1 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import cv2

In [2]: cv2.__version__

Out[2]: '4.2.0'

In [3]:
```

Note that I have already setup a conda environment for using OpenCV
Briefly – What Went Into My Setup

1. Install Anaconda
2. Modify login script (.cshrc for me)
3. Create an environment, for example cv2
4. Activate the environment
5. Run: conda install -c conda-forge opencv
6. Run: conda install -n cv2 ipython to get python
Now, OpenCV ‘Hello World’

Getting Started with Images

Goals
- Here, you will learn how to read an image, how to display it and how to save it back
- You will learn these functions: `cv2.imread()`, `cv2.imshow()`, `cv2.imwrite()`
- Optionally, you will learn how to display images with Matplotlib

Using OpenCV

Read an image

Use the function `cv2.imread()` to read an image. The image should be in the working directory or a full path of image should be given.

Second argument is a flag which specifies the way image should be read.

- `cv2.IMREAD_COLOR`: Loads a color image. Any transparency of image will be neglected. It is the default flag.
- `cv2.IMREAD_GRAYSCALE`: Loads image in grayscale mode
- `cv2.IMREAD_UNCHANGED`: Loads image as such including alpha channel

**Note**

Instead of these three flags, you can simply pass integers 1, 0 or -1 respectively.
Tutorial 00 – Load and Show

• Sources of Complexity
• Windows are named
• waitKey command matters
• …
Next up … Image Container

Mat - The Basic Image Container

Goal

We have multiple ways to acquire digital images from the real world: digital cameras, scanners, computed tomography, and magnetic resonance imaging to name a few. In every case what we (humans) see are images. However, when transforming this to our digital devices what we record are numerical values for each of the points of the image.

For example in the above image you can see that the mirror of the car is nothing more than a matrix containing all the intensity values of the pixel points. How we get and store the pixels values may vary according to our needs, but in the end all images inside a computer world may be
MAT as of OpenCV 2.0 (and up)

- Memory management handled for us.
- A tremendous amount of information hiding.
- All and all – a good thing.
- All common types supported
  - Gray scale
  - Color (RGB, HSV, ..)
  - In case you wondered:
    - “There are more than 150 color-space conversion methods available in OpenCV.”
Basic Operations on Images

Goal

Learn to:

- Access pixel values and modify them
- Access image properties
- Setting Region of Image (ROI)
- Splitting and Merging images

Almost all the operations in this section is mainly related to Numpy rather than OpenCV. A good knowledge of Numpy is required to write better optimized code with OpenCV.
Tutorial01 – Print Part of an Image

```python
import cv2
import numpy as np

img = cv2.imread('IconFaceLv2.png')

def showPanda():
    cv2.namedWindow('panda',cv2.WINDOW_NORMAL)
    cv2.imshow('panda',img)
    cv2.waitKey(0)

def printEye () :
    eye = img[180:190, 210:220]
    print eye
```

```python
[ 39  58 101]
[ 44  67 113]
[ 36  60 106]

[[246 230 231]
 [236 224 224]
 [217 209 209]
 [188 185 194]
 [165 166 186]
 [129 140 167]
 [ 76  92 129]
 [ 44  63 106]
 [ 47  70 116]
 [ 37  61 107]]
```
Tutorial 02 – ROIs and Cropping

```python
# Introduction and overview of IPython's features.
# quickref -> Quick reference.
# help -> Python's own help system.
# object? -> Details about 'object', use 'object??' for extra details.

In [1]: execfile('tutorial02.py')

In [2]: cropPanda()
Select a ROI and then press SPACE or ENTER button!
Cancel the selection process by pressing c button!

```

```python
import cv2
import numpy as np

im = cv2.imread('IconFace.png')

def cropPanda():
    # Example from
    # https://www.learnopencv.com/opencv-cpp-python

    # Select ROI
    r = cv2.selectROI(im)

    # Crop image
    imCrop = im[int(r[1]):int(r[1]+r[3]), int(r[0]):int(r[0]+r[2])]

    # Display cropped image
    cv2.imshow("Image", imCrop)
    cv2.waitKey(0)
```

import cv2
import numpy as np

im = cv2.imread('IconFaceLv2.png')
def colorPanda():
    # Example extended from
    # Select ROI
    r = cv2.selectROI(im)
    # Turn pixels in the ROI Red!
    im[int(r[1]):int(r[1]+r[3]), int(r[0]):int(r[0]+r[2])] = [0,0,255]
    # Display altered image
    cv2.imshow("Image", im)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
import cv2
import numpy as np

def playDusty()
    cap = cv2.VideoCapture('IMG_3001.mp4')
    while(1):
        ret, frame = cap.read()
        if ret == True:
            cv2.imshow("Image", frame)
            k = cv2.waitKey(0) & 0xff
            if k == 27:
                break
    cv2.destroyAllWindows()
cap.release()
Assignment 1: Tracking

• In preparation for Assignment 1, please …
• Choose a partner.
  – If you seek help communicate with me individually through email
• Setup OpenCV so you can write/run code.
• Begin reviewing OpenCV tracking algorithms.
  – Here is a resource:
    • “Object Tracking using OpenCV (C++/Python)”