Intro to the OpenCV Library

for TU Dresden Computer Vision 2 lecture and general use

(some slides are cc from 'opencv 3.0' Kirill Kornyakov, Itseez)

Topics

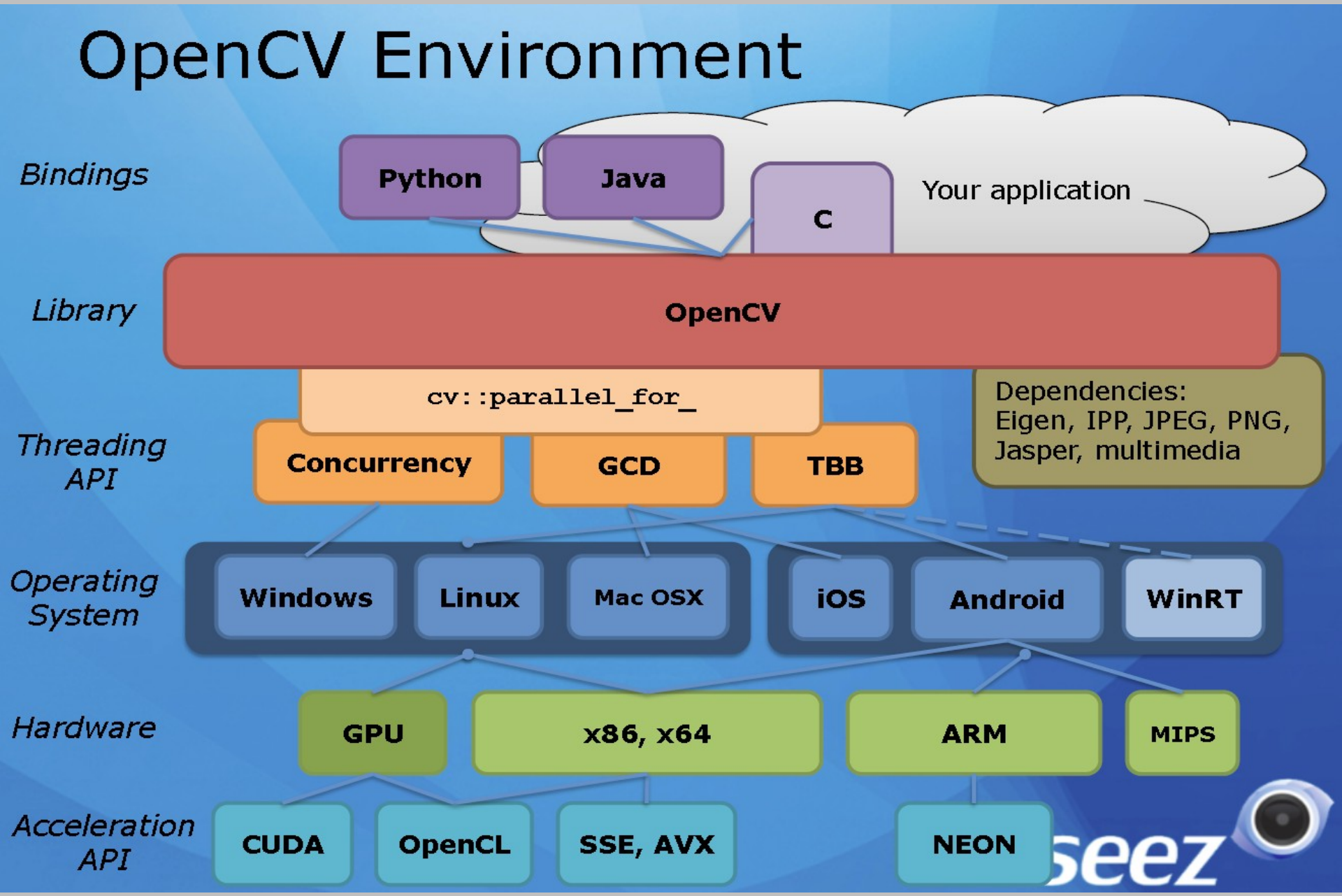
1. Why
2. What
3. Install
4. Example Project
5. Your Task
6. Your Questions → in Opal forum

Why OpenCV?

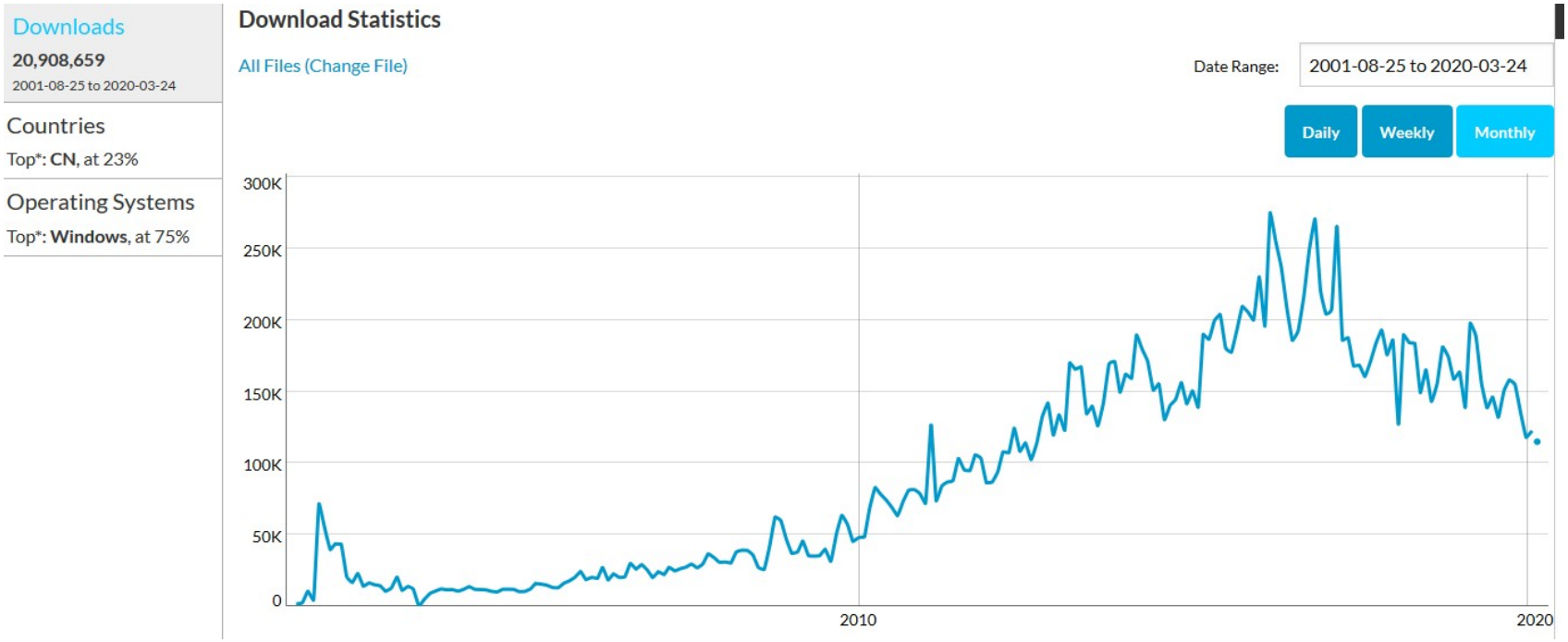
1. 2,500+ algorithms and functions
2. Cross-platform, portable API
3. Real-time performance
4. Liberal BSD license
5. fast and regular updates




OpenCV Environment



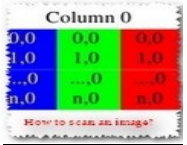
History



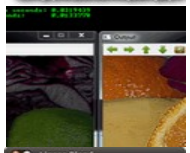
2001: Intel → open src, 2008: Willow Garage, itSeez, 2010: Nvidia,
2016: Intel buys itSeez, Machine Learning rise (people move from cv to ml)

What? core module tutorials:

Mat - The Basic Image Container



How to scan images, lookup tables and time measurement with OpenCV



Mask operations on matrices

Adding (blending) two images using OpenCV

Changing the contrast and brightness of an image

Basic Drawing

Random generator and text with OpenCV

Discrete Fourier Transform

File Input and Output using XML and YAML files

What? imgproc module tutorials:

Smoothing Images
Eroding and Dilating
More Morphology Transformations
Image Pyramids
Basic Thresholding Operations
Making your own linear filters!
Adding borders to your images
Sobel Derivatives
Laplace Operator
Canny Edge Detector
Hough Line Transform
Hough Circle Transform
Remapping
Affine Transformations

Histogram Equalization
Histogram Calculation
Histogram Comparison
Back Projection
Template Matching
Finding contours in your image
Convex Hull
Creating Bounding boxes and circles for contours
Creating Bounding rotated boxes and ellipses for contours
Image Moments
Point Polygon Test

What? Other modules:

Highgui:
Adding a Trackbar to our applications!
Video Input with OpenCV
Creating a video with OpenCV

calib3d:
Camera calibration

ml:
Support Vector Machines
Deep Neural Networks (dnn)
Principal Component Analysis

feature2d:
Harris corner detector
Shi-Tomasi corner detector
Creating your own corner detector and similarity measurement
Detecting corners location in subpixeles
Feature Description
Feature Matching with FLANN
Features2D + Homography to find a known object
Detection of planar objects

objdetect:
Cascade Classifier

What? code examples:

Hello World example shows an image:

```
int main(int argc, char** argv)
{
    Mat img = imread(argv[1], 1);
    imshow("", img);
    waitKey();
    return 0;
}
```

What? code examples:

```
int main(int argc, char** argv)
{
    Mat img, gray;
    img = imread(argv[1], 1);
    imshow("original", img);
    cvtColor(img, gray, COLOR_BGR2GRAY);
    GaussianBlur(gray, gray, Size(7, 7),
                 1.5);
    Canny(gray, gray, 0, 50);
    imshow("edges", gray);
    waitKey();
    return 0;
}
```

What? code examples:

Threshold:

```
Mat emptyPixImg = GrayImg < 1;
```

Image from (Camera- or) Directory-stream:

```
VideoCapture cap("TextureImages/Texture_%02d_inpaint.png");
Mat Img;
cap >> Img;
```

Create a 2D-Gaussian:

```
Mat Gauss2D = Mat::zeros(TemplateWidth, TemplateWidth,
                                                    CV_32FC1);
Gauss2D.at<float>( TemplateHW, TemplateHW) = 1.0;
GaussianBlur(Gauss2D, Gauss2D, Size(TemplateWidth,
                                TemplateWidth), sigma, sigma);
```

What? code examples:

pointer work to speed up inner loops (useful in debug mode):

(1)

```
int** iim = new int*[h];
for (y=0; y<h; y++)
{  iim[y] = IntegralImg.ptr<int>(y);
}
int diffy = 2*( iim[y][x+dx]    - iim[y][x-dx] )  +
                iim[y-dy][x-dx] - iim[y-dy][x+dx] +
                iim[y+dy][x-dx] – iim[y+dy][x+dx];
```

(2)

```
float *pCR, *pCRData = (float*) CorrResult.data;
*pCR  = pCRData + y*w;
for ( int x = TemplateWidth; x < w-TemplateWidth; x++ )
{  pCR[x] = ssd;    // write ssd result to result image
}
```

How?

1. Home: opencv.org
2. Documentation: docs.opencv.org
3. Q&A forum: answers.opencv.org
4. Report issues: github.com/opencv/opencv/issues
5. Develop: github.com/opencv/opencv

How? Install:

1. download:https://github.com/opencv/opencv/
2. run Cmake(gui), check/install add-ons and
   configure until all problems have gone
   create debug libs from source code, include Qt
   generate makefile or .sln
3. make / compile
4. sudo make install (depending on environment)

(I use win+visual studio only currently, otherwise debian and kdevelop, QtCreator may also be a choice if you do not already have your environment)

5. setup your IDE for OpenCV (set path, includes ...)
6. run example(s)

Thus you can run AND debug into any OpenCV example → good to read the code and for debugging

What YOU do::

1. download OpenCV source code from:
   https://github.com/opencv/opencv/
2. Install additionally Libs (Qt, Eigen, …) and
   compile OpenCV; create debug libs!
3. Setup IDE and run edges example

You have one week for this task.

What YOU do:

1. Set up your development environment and
   make a (simple ?) grey value transformation program.
2. You are free to use opencv and other example code you find,
3. but should
   - put it all together on your own
   - cite your source in a comment.
4. Good C++ coding style and a lot of comments!
5. Your code must compile without errors on Win
   and Linux systems (i.e. avoid Win-specific code).

Here is a link to the pdf for the details of coding task.
You have another week for this coding task,
i.e. until 2020-04-20 (maybe more because of Easter).