OpenCV 3.0
Uses in Robotics and AR

Gary Bradski
VP Perception and Core Software,
Magic Leap
Director: OpenCV Foundation
OpenCV Thanks! For Key Support

• Intel
  – for getting it started and helping fund the challenge

• Google
  – for growing support in Google Summer of Code over the last 5 years

• Nvidia
  – Supporting Cuda version with lots of other help
Outline: OPENCV 3.0

• Intro
  – Learning OpenCV Version 2.0 coming by Aug
  – Announcing $50K Vision Challenge

• OpenCV Background

• OpenCV 3.0 High Level

• OpenCV 3.0 Modules

• Brand New in OpenCV
  – Robotics
  – Augmented Reality
Learning OpenCV V2.0

• Out in Summer 2014!
OpenCV $50K Vision Challenge

More information soon 10/03/2014

OpenCV is launching a community-wide challenge to update and extend the OpenCV library. An award pool of $50,000 will be provided to the best performing algorithms in the following 10 CV application areas:

- image segmentation,
- image registration,
- human pose estimation,
- SLAM,
- multi-view stereo matching,
- object recognition,
- face recognition,
- gesture recognition,
- action recognition, and
- text recognition.

We will soon provide code to read from existing data sets in each of these areas.

Conditions:

1. You may submit a new algorithm developed by yourself
2. You may submit an existing algorithm **whether or not developed by yourself** (as long as you re-implement it yourself or it already has a BSD or compatible license).
3. You must submit your winning code as an OpenCV pull request under a BSD or compatible license

   1. You acknowledge that your code may be included, with citation, in OpenCV

You may explicitly enter code for any work you have submitted to CVPR 2015 or its workshops. We will not unveil it until after CVPR.

Winners and prizes are at the sole discretion of the committee.

Timeline:

Submission Period:
Now - May 8th 2015

Winners Announcement:
June 8th 2015 at CVPR 2015

Contact:

For more information, go to [http://code.opencv.org/projects/opencv/wiki/VisionChallenge](http://code.opencv.org/projects/opencv/wiki/VisionChallenge)

Or mail:
open_cv_vision_challenges@googlegroups.com

The group is located at [https://groups.google.com/forum/#!en#forum/opencv_vision_challenges](https://groups.google.com/forum/#!en#forum/opencv_vision_challenges)
OpenCV Background
What is OpenCV

• **Open Source** **Computer Vision Library**

• Routines focused on real time image processing and 2D + 3D computer vision.
  – **On Linux, Windows, Mac, Android and iOS**
  – **C++, C, Java, Matlab and Python interfaces**

• **Free** for commercial or research use in whole or in part.
OpenCV License

• Based on BSD license
• Free for commercial and research use
• Does not force your code to be open
  – But you are very welcome to contribute back!
OpenCV History


Beta 1. Linux support

Release 1.0
Release 1.1
Release 2.0. C++
Release 2.1. Full Python support
Release 2.2. Android Support
Release 2.3. GPU Support. Modules
Release 2.4
Release 2.5
Release 3.0 Refactored

Main Current Sponsors:

Intel
Nvidia
Google

Renewed Intel Support
Willow Support
Nvidia Support


OpenCV Foundation

Google Summer of Code
Environments, Platforms

• Languages:
  – C++, C#, Python, C, Java

• Platforms:
OpenCV and Hardware Acceleration

• OpenCV was a central basis for OpenVX
  – a hardware abstraction layer
  – for embedded vision acceleration

OpenVX Supporters:

amazon.com  AMD  Aptina  ARM  AXIS Communications
BROADCOM  BDTi  CEVA  cognivue  Innovating With Vision  Huawei
itseez  Movidius  NVIDIA  Qualcomm  Renesas  Intel
Samsung  life augmented  Texas Instruments  videantis  Xilinx  Vivante
OpenCL™ performance in OpenCV 3.0

AMD A10-7850k (Kaveri) and Radeon HD7790

- MOG: CPU/C++ 2, iGPU/OCL 6, dGPU/OCL 7
- warpPerspective: CPU/C++ 2, iGPU/OCL 7, dGPU/OCL 4
- CascadeClassifier: CPU/C++ 4, iGPU/OCL 7, dGPU/OCL 4
- bilateralFilter: CPU/C++ 4, iGPU/OCL 16, dGPU/OCL 9
- cornerHarris: CPU/C++ 27, iGPU/OCL 28, dGPU/OCL 30
- CLAHE: CPU/C++ 4, iGPU/OCL 8, dGPU/OCL 5
- filter2D: CPU/C++ 30, iGPU/OCL 30, dGPU/OCL 10
- matchTemplate: CPU/C++ 2, iGPU/OCL 7, dGPU/OCL 4
- Sobel: CPU/C++ 4, iGPU/OCL 7, dGPU/OCL 4
Where is OpenCV Used?

- Academic and Industry Research
- Security systems
- Google Maps, Streetview
- Image/video search and retrieval
- Structure from motion in movies
- Machine vision factory production inspection systems
- Automatic Driver Assistance Systems
- Safety monitoring (Dam sites, mines, swimming pools)
- Robotics – personal, industrial, hobby
- Coin production in China
Popularity

Over 9M downloads!

Ramping to > 160K downloads/month
OpenCV Corporation

• Founded this July, 2012

• http://opencv.org (user site)

• http://code.opencv.org (developer site)

• Contribute (via Credit, debit or paypal):
  – http://tinyurl.com/7eujyo2

For corporate support And/or partnership, contact Garybradski@gmail.com

I am looking for entrepreneurial people to staff up OpenCV:

• Vision
• Business Dev
• Software
• Hardware
What’s In OpenCV

• High level
OpenCV Overview

> 2500 algorithms

Developer: http://code.opencv.org; User: http://opencv.org

General Image Processing Functions

Segmentation

Machine Learning:
- Detection,
- Recognition

Geometric descriptors

Transforms

Features

Tracking

Matrix Math

Image Pyramids

Camera calibration, Stereo, 3D

Utilities and Data Structures

Fitting
OpenCV Algorithm Modules

Overview

HighGUI:
I/O, Interface

Image Processing
Transforms
Fitting
Optical Flow Tracking
Segmentation

Calibration
Features VSLAM
Depth, Pose Normals, Planes, 3D Features
Object recognition Machine
Computational Photography

CORE:
Data structures, Matrix math, Exceptions etc.
Machine Learning Library (MLL)

CLASSIFICATION / REGRESSION
Fast Approximate NN (FLANN)
Extremely Random Trees
CART
Naïve Bayes
MLP (Back propagation)
Statistical Boosting, 4 flavors
Random Forests
SVM
Face Detector
(Histogram matching)
(Correlation)

CLUSTERING
K-Means
EM
(Mahalanobis distance)

TUNING/VALIDATION
Cross validation
Bootstrapping
Variable importance
Sampling methods

http://opencv.org
Modules

Algorithmic
- core, imgproc, calib3d, video, ml, objdetect, features2d
- photo, stitching, videostab, superres
- contrib, legacy, nonfree, flann

GPU
- gpu, ocl

Infrastructure
- highgui, world
- python, java
- ts, androidcamera
C:

```c
double calcGradients(const IplImage *src,
                      int aperture_size = 7)
{
    CvSize sz = cvGetSize(src);

    IplImage* img16_x = cvCreateImage(sz, IPL_DEPTH_16S, 1);
    IplImage* img16_y = cvCreateImage(sz, IPL_DEPTH_16S, 1);
    cvSobel(src, img16_x, 1, 0, aperture_size);
    cvSobel(src, img16_y, 0, 1, aperture_size);

    IplImage* imgF_x = cvCreateImage(sz, IPL_DEPTH_32F, 1);
    IplImage* imgF_y = cvCreateImage(sz, IPL_DEPTH_32F, 1);
    cvCartToPolar(img16_x, imgF_x);
    cvCartToPolar(img16_y, imgF_y);

    IplImage* magnitude = cvCreateImage(sz, IPL_DEPTH_32F, 1);
    cvScale(imgF_x, imgF_y, magnitude);
    double res = cvSum(magnitude).val[0];

    cvReleaseImage(&magnitude);
    cvReleaseImage(&imgF_x);
    cvReleaseImage(&imgF_y);
    cvReleaseImage(&img16_x);
    cvReleaseImage(&img16_y);

    return res;
}
```

C++:

```cpp
double contrast_measure(Mat& img) {
    Mat dx, dy;
    Sobel(img, dx, 1, 0, 3, CV_32F);
    Sobel(img, dy, 0, 1, 3, CV_32F);
    magnitude(dx, dy, dx);
    return sum(dx)[0];
}
```
OpenCV Architecture and Development

Languages:
C
C++
Python
CUDA
JAVA (plans)

Technologies:
CUDA
SSE
TBB

3rd party libs:
Eigen
IPP
Jasper
JPEG, PNG
OpenNI
QT
TBB
VideoInput

Development:
Maintainers
Contributors

QA:
Buildbot
Google Tests

Modules:
Core
ImgProc
HighGUI
GPU
ML
ObjDetect
Video
Calib3D
Features2D
FLANN

Target archs:
X86
X64
ARM
CUDA

Target OS:
Windows
Linux
Mac OS
Android
Web resources

opencv.org, docs.opencv.org, answers.opencv.org
Development infrastructure
What’s In OpenCV 3.0

• Modules
**OpenCV Modules: Core**

**Core**

Data structures, 
Matrix math, 
Exceptions etc.

**Key OpenCV Classes**

- **Point**
- **Point3**
- **Size**
- **Vec**
- **Mat**
- **Scalar**
- **Rect**
- **Range**
- **SparseMat**
- **Pfr**

**Matrix Basics**

- Create a matrix
- Create a matrix initialized with a constant
- Create a matrix initialized with specific values
- Initialize a random matrix

**Convert matrix to/from other structures**

- Access matrix elements

**Matrix Manipulations: Copying, Shuffling, Part Access**

- `src.copyTo(dst)` Copy matrix to another one
- `src.convertTo(dst,type,shift)` Scale and convert to another datatype

**Simple Matrix Operations**

OpenCV implements most common arithmetical, logical and other matrix operations, such as:

- `add()`, `subtract()`, `multiply()`, `divide()`, `absdiff()`, `bitwise_and()`, `bitwise_or()`, `bitwise_xor()`, `max()`, `min()` etc.

Example: Alpha compositing function:

```cpp
void alphaCompose(const Mat& rgb1, const Mat& rgb2, Mat& rgbdest) {
    Mat a1(rgb1.size(), rgb1.type()), r1;
    Mat a2(rgb2.size(), rgb2.type());
    int mixCoeff[3] = {1, 0, 3, 1, 3, 2, 3, 2, 3};
    mixChannels(&rgb1, 1, &a1, 1, mixCoeff, 4);
    mixChannels(&rgb2, 1, &a2, 1, mixCoeff, 4);
    subtract(a1, rgb1, a1, rgb1);
    bitwise.or(a1, a2, rgb2, a1, rgb2, a2, 1, mixCoeff, 4);
    multiply(a1, rgb1, a1, 2, 1, mixCoeff, 4);
    multiply(a2, rgb2, a2, 2, 1, mixCoeff, 4);
    add(a1, a2, rgbdest);
}
```

- `sum()`, `mean()`, `meanStdDev()`, `norm()`, `countNonZero()`, `minMaxLoc()`, 
  - various statistics of matrix elements.
- `exp()`, `log()`, `pow()`, `sqrt()`, `cvtColor()`, `polarToCart()`
  - the classical math functions.
- `scaleAdd()`, `transpose()`, `getPerspectiveTransform()`, `solve()`, `determinant()`, `trace()`, `eigen()`, `SVD`,
  - the algebraic functions + SVD class.
- `dft()`, `idft()`, `dct()`, `idct()`
  - discrete Fourier and cosine transformations

For some operations a more convenient algebraic notation can be used, for example:

```cpp
Mat delta = (J.t() * J) + lambda * Mat::eye(J.cols, J.cols, J.type());
```

... implements the core of Levenberg-Marquardt optimization algorithm.

**It**

- `f1`, `f2`, `f3`
- `s1`, `s2`, `s3`
- `c1`, `c2`, `c3`
- `g1`, `g2`, `g3`
- `b1`, `b2`, `b3`
- `L1`, `L2`, `L3`
- `en`, `ex`, ...

OpenCV Modules: HighGUI

HighGUI: I/O, Interface
OpenCV Modules: Fitting

- Convex Hull
- Delaunay
- 2D Rigid Objects
- Ellipse
- 3D Rigid Objects
int main(…){
    CvCapture* capture = <…> ?
        cvCaptureFromCAM(camera_id) :
        cvCaptureFromFile(path);
    if( !capture ) return -1;
    for(;;) {
        IplImage* frame=cvQueryFrame(capture);
        if(!frame) break;
        // … copy and process image
        cvCalcOpticalFlowPyrLK( …)
        cvShowImage( “LkDemo”, result );
        c=cvWaitKey(30); // run at ~20-30fps speed
        if(c >= 0) {
            // process key
        }
    cvReleaseCapture(&capture
}

I(x + dx, y + dy, t + dt) = I(x, y, t);
-∂I / ∂t = ∂I / ∂x · (dx / dt) + ∂I / ∂y · (dy / dt);

G · ∂X = b,
∂X = (∂x, ∂y), G = ∑[I_x^2, I_x I_y ; I_x I_y, I_y^2], b = ∑I_t [I_x I_y ]
OpenCV Modules: Segmentation

- Watershed
- Grab Cut
- Background subtraction
- Color

https://www.youtube.com/watch?v=OxmDonZja74
http://www.youtube.com/watch?v=Ktrjh5-KLKo
OpenCV Modules: Calibration

Homography

3D view of checkerboard

Un-distorted image

http://www.youtube.com/watch?v=DrXIQfQHFv0

http://www.youtube.com/watch?v=PuWQNcReIEc
Read two input images:

Mat img1 = imread(argv[1], CV_LOAD_IMAGE_GRAYSCALE);

Detect keypoints in both images:

// detecting keypoints
FastFeatureDetector detector(15);
vector<KeyPoint> keypoints1;
detector.detect(img1, keypoints1);

Compute descriptors for each of the keypoints:

// computing descriptors
SurfDescriptorExtractor extractor;
Mat descriptors1;
extractor.compute(img1, keypoints1, descriptors1);

Now, find the closest matches between descriptors from the first image:

// matching descriptors
BruteForceMatcher<L2<float> > matcher;
vector<DMatch> matches;
matcher.match(descriptors1, descriptors2, matches);
OpenCV Modules: Depth, Pose

Depth, Pose
Normals, Planes, 3D Features

Some examples of 3D stereo depth maps:

Left - right feature alignment:
OpenCV Modules: Obj Rec/ML

Object recognition
Machine learning

https://www.youtube.com/watch?v=_RF0VpR4xog

VSLAM

http://youtu.be/i1uUuWwbIcc
OpenCV Modules: Comp Photog

- Image Stitching (Occipital Corp.)
- Textural Inpainting
- Tilt-shift
Brand New in OpenCV 3.0
User Contrib Module

• Thanks to Google Summer of Code!!
  – Supporting 15 interns!

Accepted pull requests:

1. Extended Python interface
2. 3D object recognition and pose
3. KAZE features
4. Car detection
5. Computational photography
6. Custom calibration and planar AR
7. Dense optical flow
8. New line segment detector
9. Haze removal, depth estimation
10. GPU accelerated dense optical flow
11. DTAM & pose estimation
12. PNP pose detection
13. Visual saliency filters
14. Text detection and reading in wild
15. TLD tracker
OpenCV Examples

• Industrial Perception
• Magic Leap
Industrial Perception

- Sensor driven,
- Real time planning
- Applied to distribution
Magic Leap

• Augmented Reality done right
• Lots of computer vision (We’re hiring)

• Gesture recognition demo
Gesture ... was going to be live demo
Language Modules

• GPU/Cuda
• Android
• iOS
• Python
• Java
OpenCV GPU Module:

- Image processing building blocks:
  - Color conversions
  - Geometrical transforms
  - Per-element operations
  - Integrals, reductions
  - Template matching
  - Filtering engine
  - Feature detectors

- High-level algorithms:
  - Stereo matching
  - Face detection
  - Feature matching
OpenCV GPU Module Example

Mat frame;
VideoCapture capture(camera);
cv::HOGDescriptor hog;
hog.setSVMDetector(cv::HOGDescriptor
::getDefaultPeopleDetectorector());
capture >> frame;

vector<Rect> found;
hog.detectMultiScale(frame, found,
1.4, Size(8, 8), Size(0, 0),
1.05, 8);

Mat frame;
VideoCapture capture(camera);
cv::gpu::HOGDescriptor hog;
hog.setSVMDetector(cv::HOGDescriptor
::getDefaultPeopleDetectorector());
capture >> frame;

GpuMat gpu_frame;
gpu_frame.upload(frame);

vector<Rect> found;
hog.detectMultiScale(gpu_frame,
found,
1.4, Size(8, 8), Size(0, 0),
1.05, 8);

• Designed very similar!
OpenCV GPU Module

Performance

Tesla C2050 (Fermi) vs. Core i5-760 2.8GHz (4 cores, TBB, SSE)

– Average speedup for primitives: **33x**
  - For “good” data (large images are better)
  - Without copying to GPU

What can you get from your computer?

- opencv\samples\gpu\performance
OpenCV Android Module

- **OpenCV 2.4 for Android:**
  - Native Android Camera Support
  - Multithreading
  - Java API (soon)
  - Tegra HW Optimizations (soon)

Wiki with the latest information:

http://opencv.org/platforms/android.html

Support/discussion group::: https://groups.google.com/group/android-opencv
OpenCV iOS Module

• Full support
OpenCV Python Module

• Full Python interface

• Example: Depth image from Kinect:

```python
import numpy
import cv
from freenect import sync_get_depth as get_depth, sync_get_rgb as get_video
while True:
    (depth, _), (rgb, _) = get_depth(), get_video()
    depth = depth.astype(numpy.uint8)
    cv.ShowImage("depth", depth)
    cv.ShowImage("depth", rgb)
```
OpenCV Java Module
Book and Foundation
Learning OpenCV V2.0

• Out in Summer 2014!
## OpenCV Foundation Support

**SUPPORT (an answer within the amount of opening days):**
- **level 0:** 1K support within 1 week
- **level 1:** 5K support within 1 week and dedicated machine on build farm
- **level 2:** 10K support within 3 days and dedicated machine on build farm
- **level 3:** 20K support within 24h and dedicated machine on build farm
- **level 4:** 30K support within 24h, dedicated machine on build farm and fixes when errors happen on the machine

### SPONSORSHIP:

<table>
<thead>
<tr>
<th>Level</th>
<th>Amount</th>
<th>Benefits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diamond</td>
<td>$250K</td>
<td>Level 4 support. Can direct OpenCV development/Strategy/priorities. Board position. Able to brainstorm solutions to proprietary problems with the team. Front page logos</td>
</tr>
<tr>
<td>Platinum</td>
<td>$100K</td>
<td>Level 3. Board position, strong influence on priorities real time support as above. Front page logos</td>
</tr>
<tr>
<td>Gold</td>
<td>$50K</td>
<td>Level 2. Advisory board (suggest priorities). Quarterly brainstorm sessions</td>
</tr>
<tr>
<td>Silver</td>
<td>$25K</td>
<td>Level 1. Advisory board, bi-yearly brainstorm logo on workshops</td>
</tr>
<tr>
<td>Titanium</td>
<td>$10K</td>
<td>Level 0. Logo on prize sponsorship</td>
</tr>
<tr>
<td>Bronze</td>
<td>$5K</td>
<td>Logo on bounties</td>
</tr>
<tr>
<td>Contributor</td>
<td>&lt;$1K</td>
<td>Contributor page</td>
</tr>
</tbody>
</table>