Начинаем работать с библиотекой OpenCV

Александр Валерьевич Шишков,
ведущий инженер компании
ITSEEZ

Содержание

- Общая информация
- Обзор функциональности
- Подготовка к работе
- Первые примеры
- Работы с матрицами

Краткая справка

- Страничка: http://opencv.org
- Фактически, самая популярная библиотека компьютерного зрения.
- Написана на C/C++, исходный код открыт, включает более 1000 функций/алгоритмов.
- Лицензия BSD (разрешается бесплатное использование дома, для учебы, на работе)
- Разрабатывается с 1998г, сначала в Интел, теперь в компании Itseez при активном участии сообщества.
- >10000000 загрузок (без учета трафика с github)
- Используется многими компаниями, организациями, ВУЗами, например в MagicLeap, NVidia, Intel, Google, Stanford …

спонсоры

www

issue tracker	code.opencv.org ⇒ github.com
wiki	code.opencv.org ⇒ github.com
Q&A forum	answers.opencv.org
online documentation	docs.opencv.org
news / social	opencv.org + FB/TW/G+
downloads	sourceforge.net/projects/opencvlibrary

Архитектура и разработка OpenCV

Languages:

C/C++
Python
Java

(Matlab, Ruby, Haskell, C#, Lua, Closure)

Acceleration Technologies:

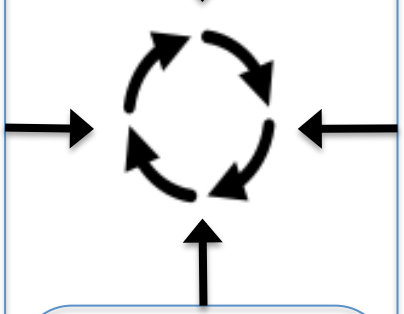
CUDA
OpenCL
parallel for (OpenMP, TBB…)
SIMD (SSE, NEON)

3rd-party libs:

zlib, png, tiff
jpeg, jasper, webp
Gtk, Qt, Cocoa, Win32
OpenNI, V4L … (20+ backends)
intel ipp, tbb
eigen
ffmpeg
Tesseract (OCR)
VTK
libmv
caffe (in progress)

Development:

Maintainers
Contributors



Build & QA:

CMake, Python
Doxygen
Github, Builbot
GTest

Modules:

“main” opencv:
core, imgproc, photo, video, calib3d features2d ml, flann objdetect, shape highgui, viz stitching, superres videostab …

opencv_contrib:
rgbd, tracking
text, reg
face, bioinspired
ximgproc, xphoto
xfeatures2d

Target Arch:

x86/x64
ARM
…

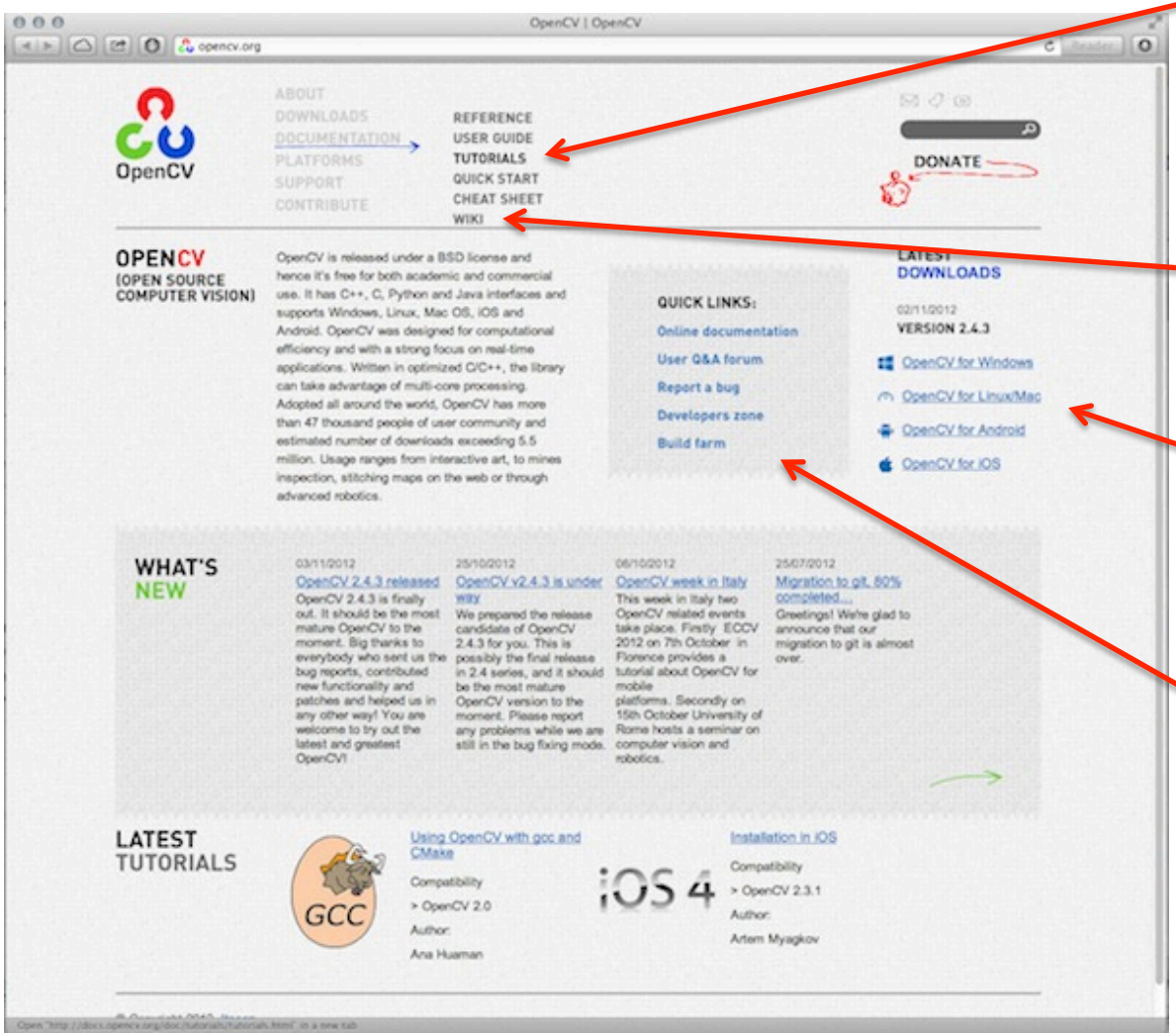
OS’es:

Windows
Linux
Android
OSX
iOS

QNX, BSD, …

С чего начать?

http://opencv.org:



Читаем уроки, Распечатываем cheatsheet.

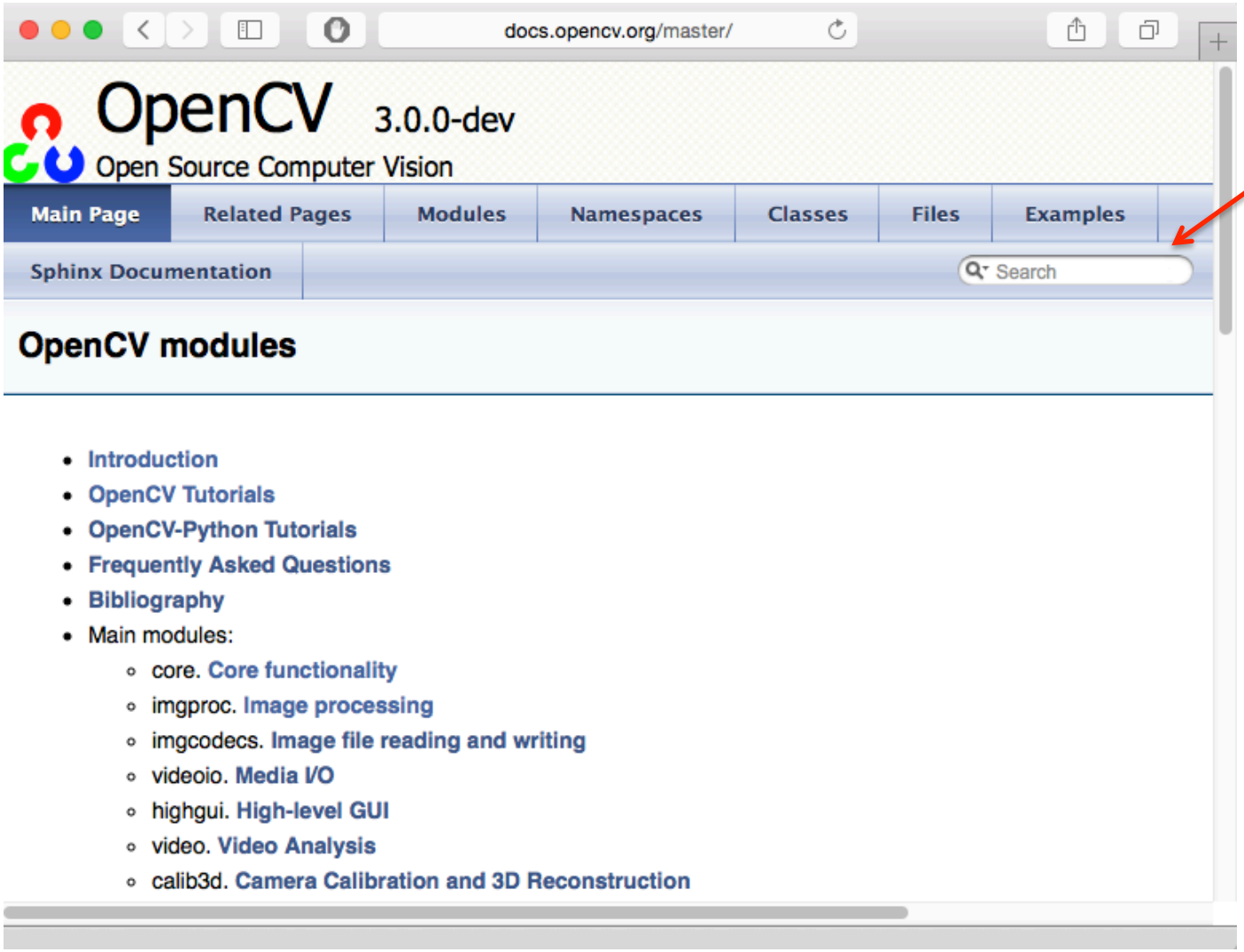
Ссылка WIKI ведет на github

Качаем

Другие сайты OpenCV (см. дальше)

Сайт с документацией

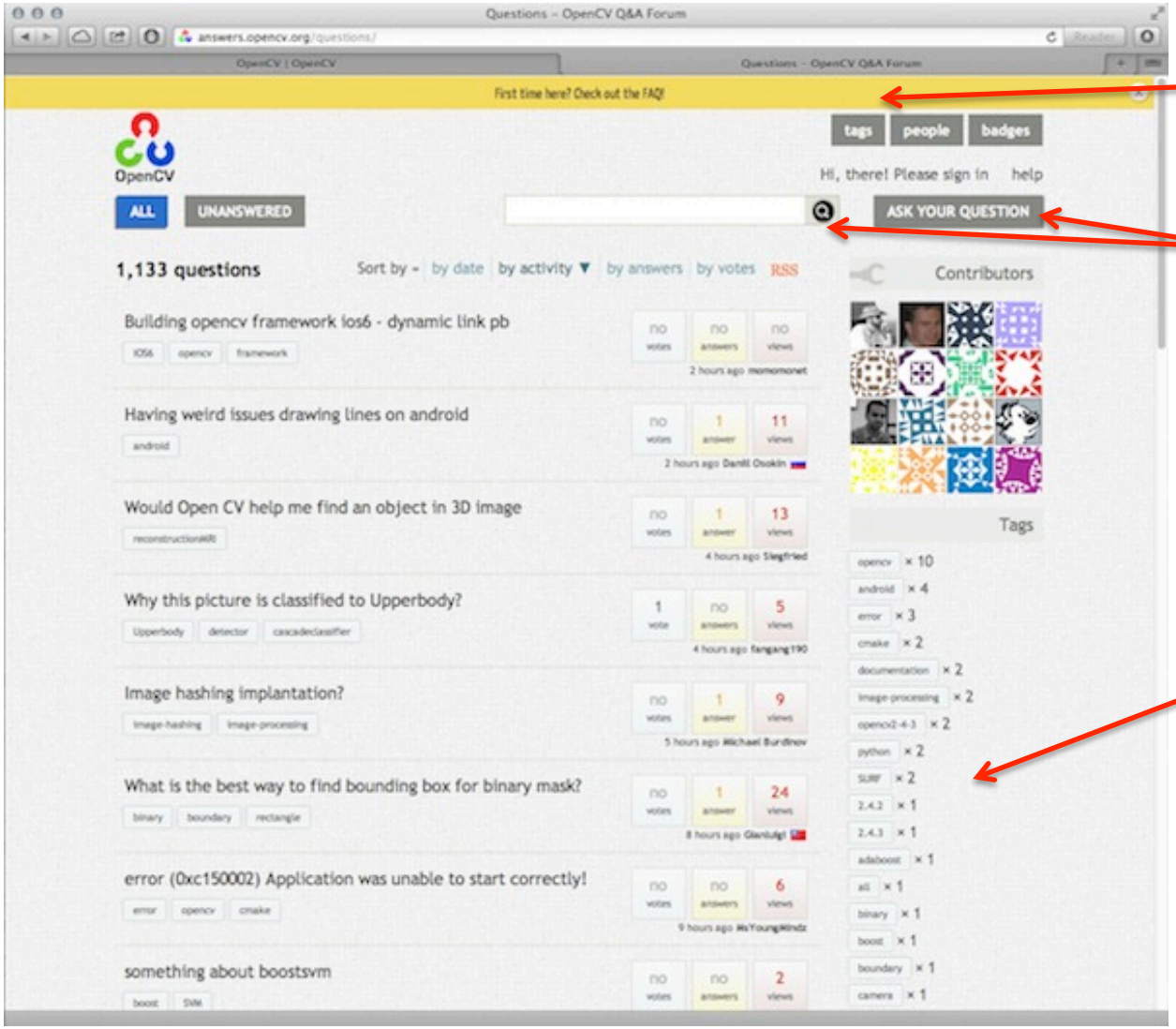
http://docs.opencv.org/master: справочник, руководство, уроки



Поиск по документации.

Задать вопрос, найти ответ

http://answers.opencv.org: сделан по аналогии со StackOverflow

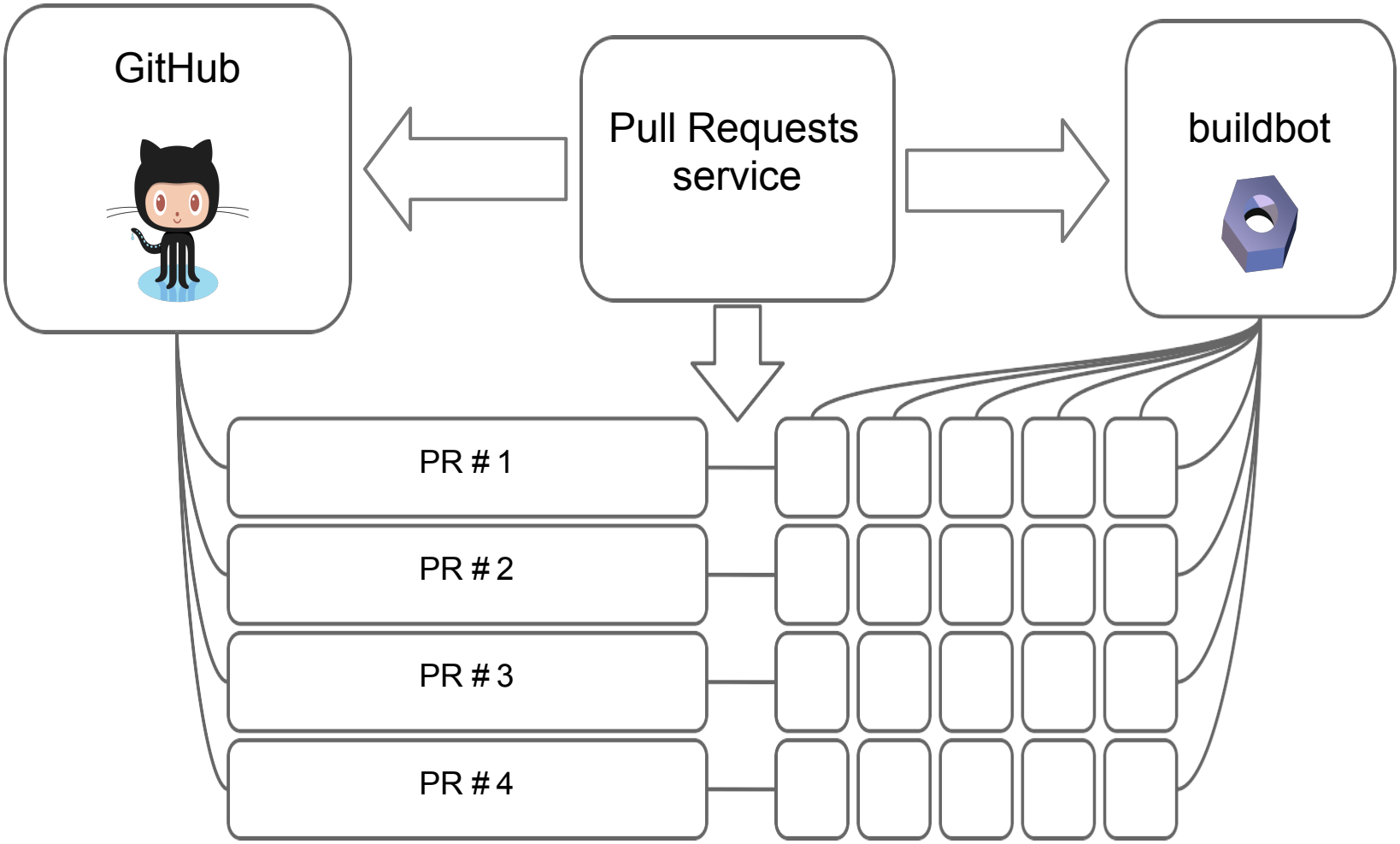


Начинаем с FAQ

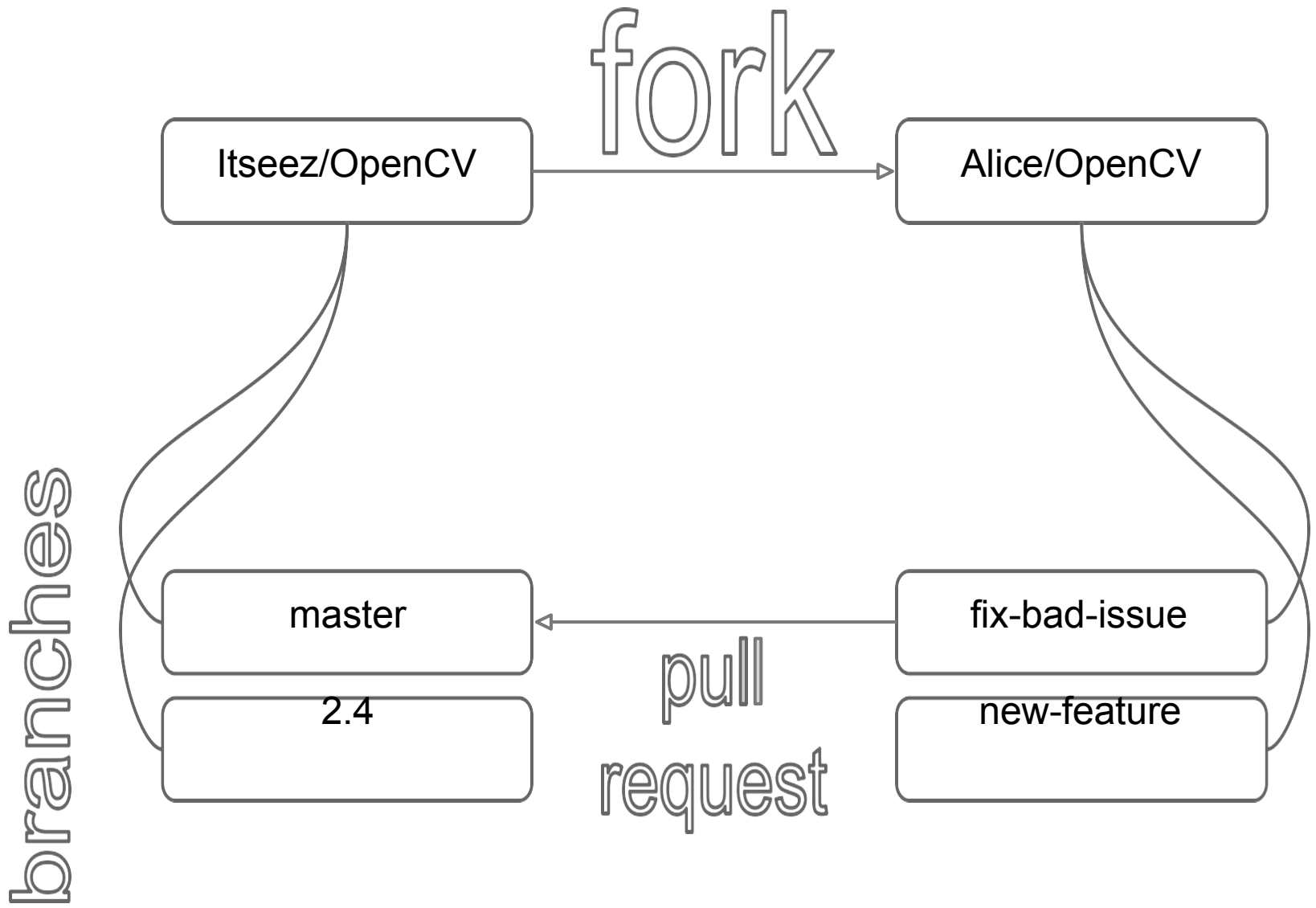
1. Поискать ответы
2. Задать свой вопрос

теги

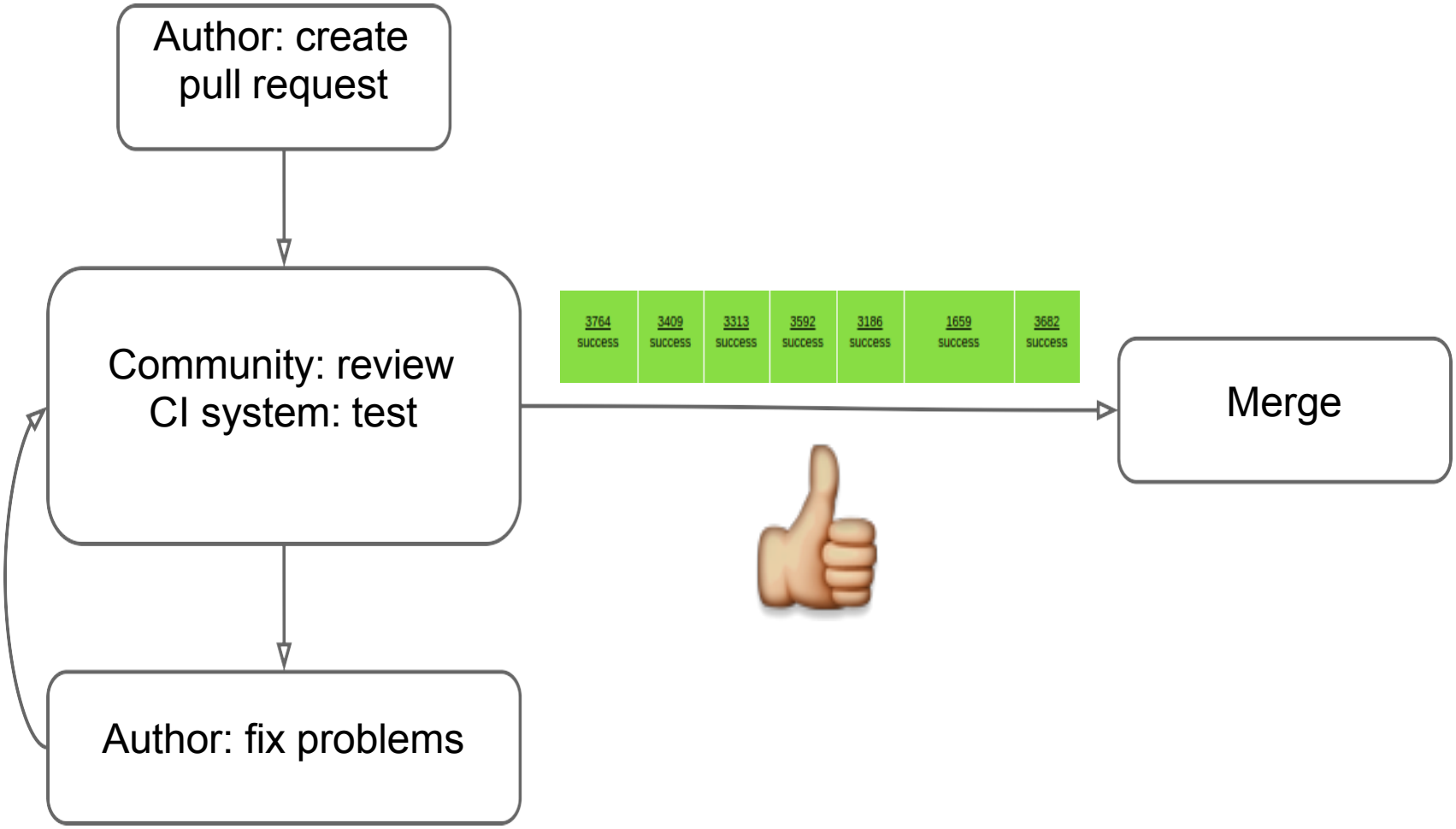
Pull Requests service



GitHub


fork
Itseez/OpenCV
Alice/OpenCV
master
2.4
fix-bad-issue
new-feature
pull
request
branches

Contribution process



Python/Java wrappers

```
CV_EXPORTS_W void sqrt(InputArray src, OutputArray dst);
```

  - automatic
  - easy to add new functionality
- disadvantages:
  - C++ parsing is hard
  - documenting
  - memory management

Java example

```java
import org.opencv.core.Core;
import org.opencv.core.Mat;
import org.opencv.core.CvType;
import org.opencv.core.Scalar;
class SimpleSample {
  static{ System.loadLibrary(Core.NATIVE_LIBRARY_NAME); }
  public static void main(String[] args) {
    System.out.println("Welcome to OpenCV " + Core.VERSION);
    Mat m = new Mat(5, 10, CvType.CV_8UC1, new Scalar(0));
    System.out.println("OpenCV Mat: " + m);
    Mat mr1 = m.row(1);
    mr1.setTo(new Scalar(1));
    Mat mc5 = m.col(5);
    mc5.setTo(new Scalar(5));
    System.out.println("OpenCV Mat data:\n" + m.dump());
  }
}
```

Python example

```
import cv2
import numpy as np
import sys
src = cv2.imread(sys.argv[1], 1)
img = cv2.cvtColor(src, cv2.COLOR_BGR2GRAY)
img = cv2.medianBlur(img, 5)
cimg = src.copy() # numpy function
circles = cv2.HoughCircles(img, cv2.HOUGH_GRADIENT, 1, 10
a, b, c = circles.shape
for i in range(b):
    cv2.circle(cimg, (circles[0][i][0], circles[0][i][1])
    cv2.circle(cimg, (circles[0][i][0], circles[0][i][1])
cv2.imshow("source", src)
cv2.imshow("detected circles", cimg)
cv2.waitKey(0)
```

DONATE

OPENCV (OPEN SOURCE COMPUTER VISION)

OpenCV is released under a BSD license and hence it's free for both academic and commercial use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV was designed for computational efficiency and with a strong focus on real-time applications. Written in optimized C/C++, the library can take advantage of multi-core processing. Enabled with OpenCL, it can take advantage of the hardware acceleration of the underlying heterogeneous compute platform. Adopted all around the world, OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 9 million. Usage ranges from interactive art, to mines inspection, stitching maps on the web or through advanced robotics.

QUICK LINKS:

- Online documentation
- User Q&A forum
- Report a bug
- Build farm
- Store

LATEST DOWNLOADS

2014-08-21

VERSION 3.0 ALPHA

- OpenCV for Windows
- OpenCV for Linux/Mac
- OpenCV for iOS

WHAT'S NEW

2014-08-21

OpenCV 3.0 alpha

OpenCV 3.0 alpha is released, with refined API, greatly improved performance on CPU, transparent acceleration on GPU and tons of new functionality in the new contrib repository.

2014-08-06

Ceemple

Ceemple, a JIT based C++ technical computing environment, now includes OpenCV, for rapid and easy development of optimized OpenCV C++ applications.

2014-08-05

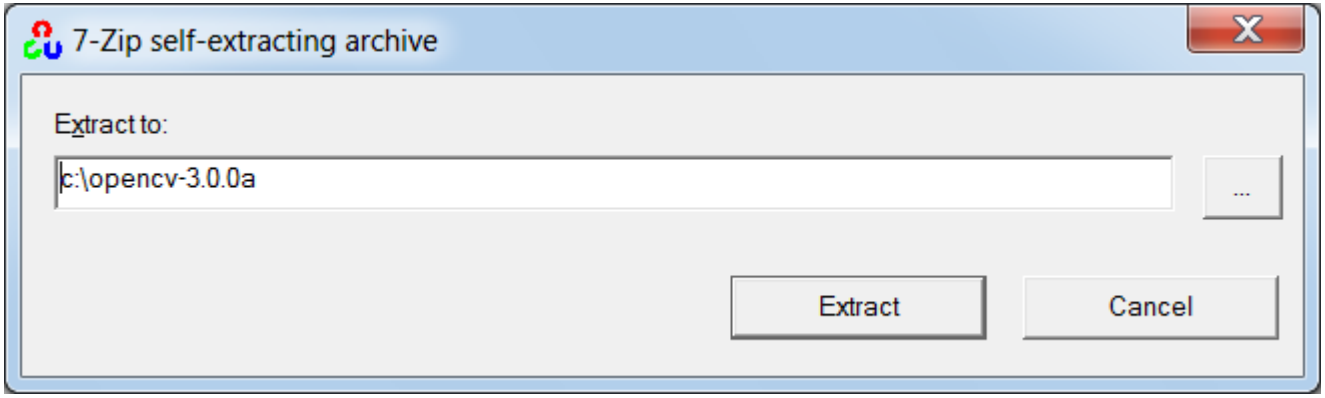
New OpenCV books

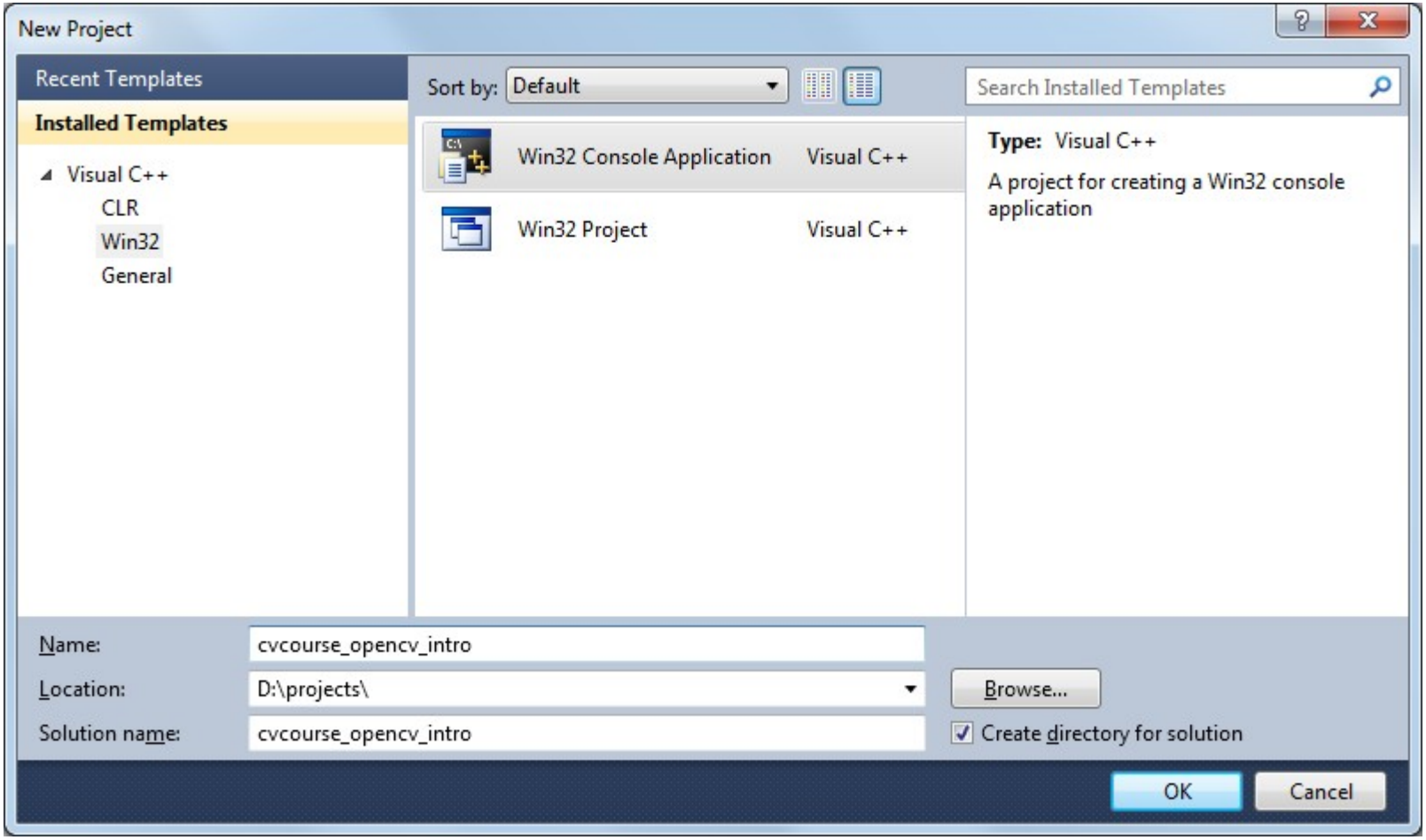
We are pleased to announce new books about OpenCV that show you how to use the Python bindings to solve actual, real-world problems.

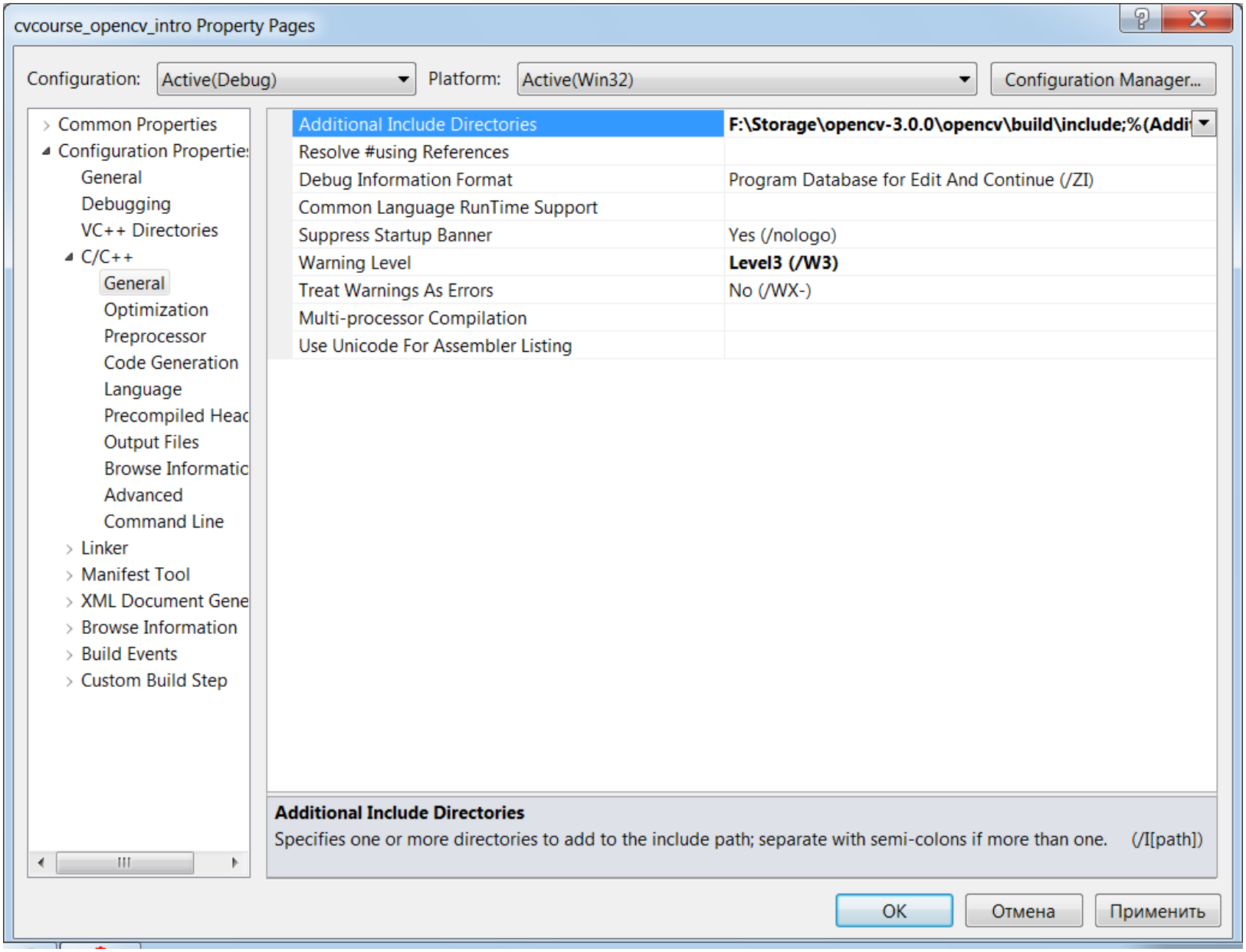
2014-08-04

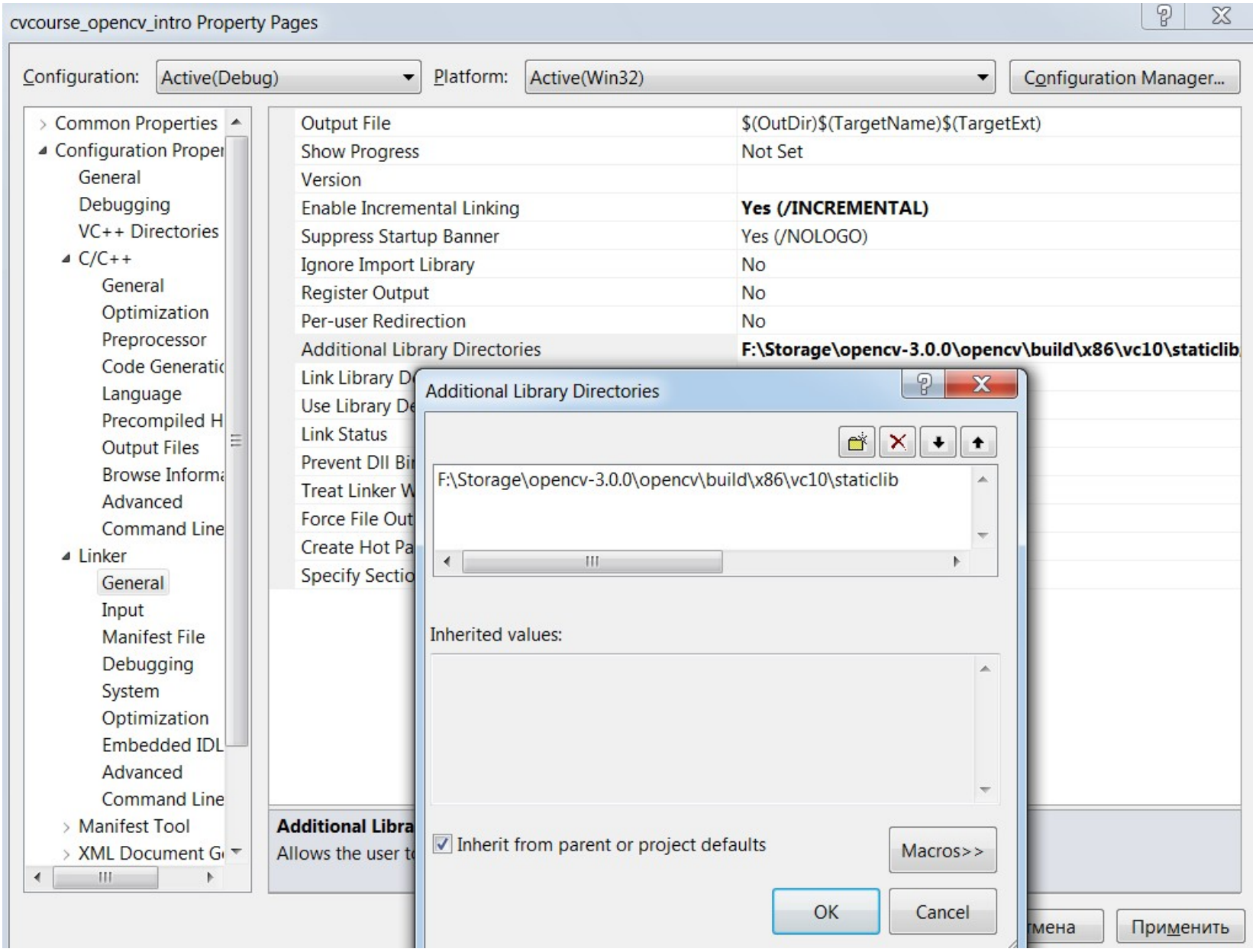
Cassandra ships fourth update of Development Platform

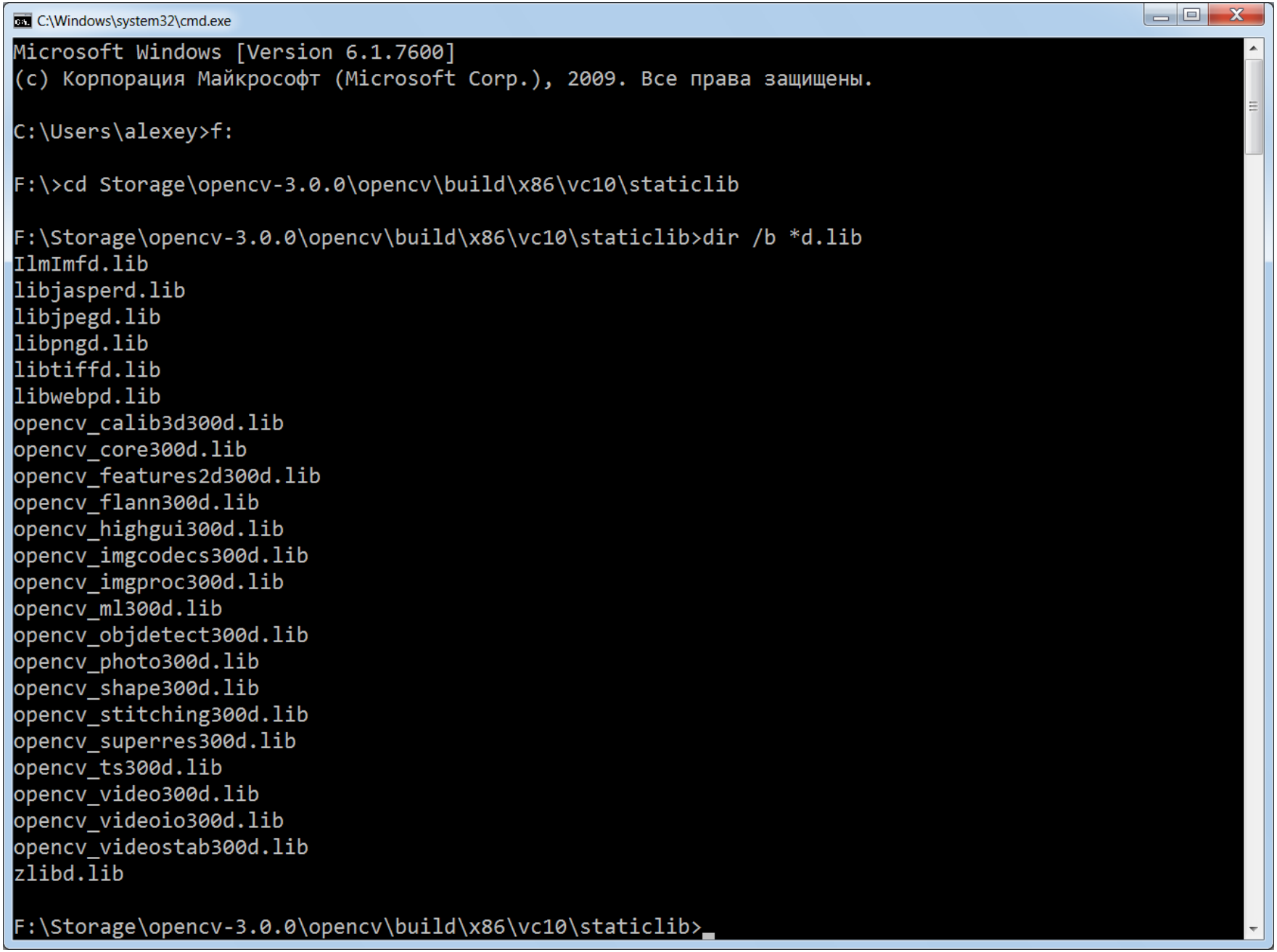
Cassandra Team is pleased to announce the immediate availability of a new Cassandra software update. This is a maintenance release for the 11 series of

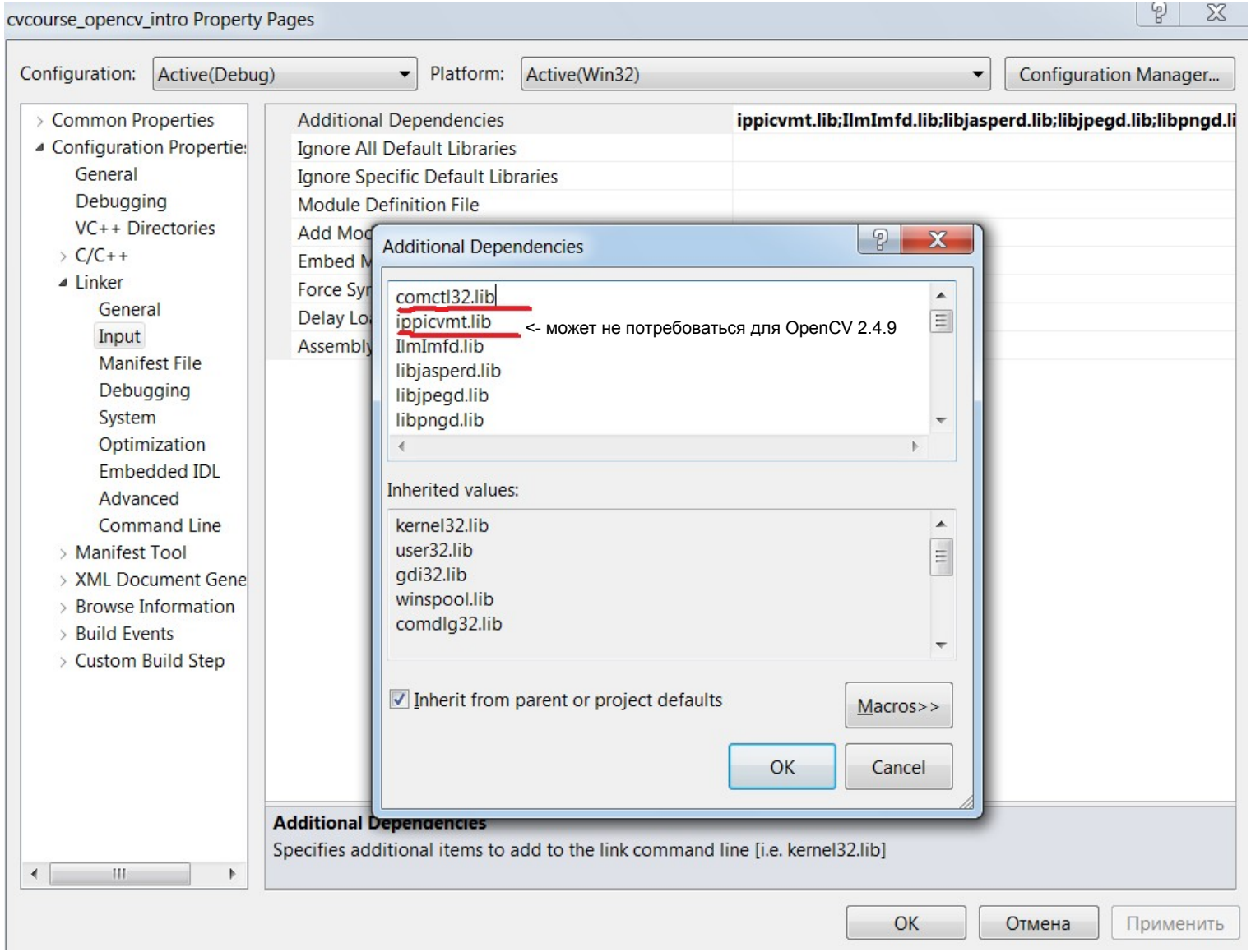

7-Zip self-extracting archive
Extract to:
c:\opencv-3.0.0a
...
Extract
Cancel

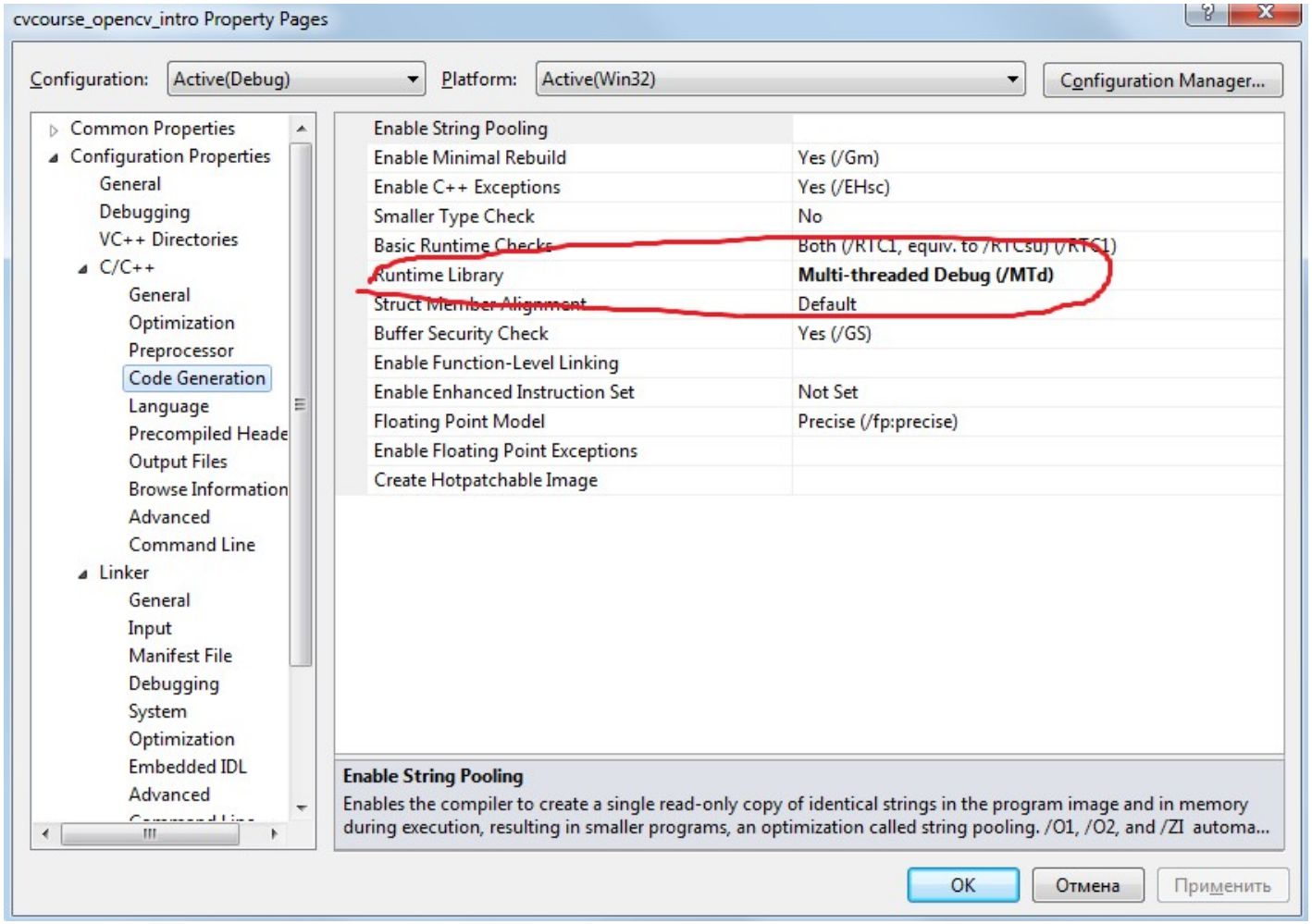

New Project
Recent Templates
Installed Templates
Visual C++
CLR
Win32
General
Sort by:
Default
Search Installed Templates
Win32 Console Application
Visual C++
Win32 Project
Visual C++
Type: Visual C++
A project for creating a Win32 console application
Name:
cvcourse_opencv_intro
Location:
D:\projects\
Browse...
Solution name:
cvcourse_opencv_intro
Create directory for solution
OK
Cancel


cvcourse_opencv_intro Property Pages
Configuration: Active(Debug)
Platform: Active(Win32)
Configuration Manager...
Common Properties
Configuration Properties
General
Debugging
VC++ Directories
C/C++
General
Optimization
Preprocessor
Code Generation
Language
Precompiled Head
Output Files
Browse Informatic
Advanced
Command Line
Linker
Manifest Tool
XML Document Gene
Browse Information
Build Events
Custom Build Step
Additional Include Directories
F:\Storage\opencv-3.0.0\opencv\build\include;%(Addi
Resolve #using References
Debug Information Format
Program Database for Edit And Continue (/ZI)
Common Language RunTime Support
Suppress Startup Banner
Yes (/nologo)
Warning Level
Level3 (/W3)
Treat Warnings As Errors
No (/WX-)
Multi-processor Compilation
Use Unicode For Assembler Listing
Additional Include Directories
Specifies one or more directories to add to the include path; separate with semi-colons if more than one. (/I[path])
OK
Отмена
Применить


cvcourse_opencv_intro Property Pages
Configuration: Active(Debug)
Platform: Active(Win32)
Configuration Manager...
Common Properties
Configuration Proper
General
Debugging
VC++ Directories
C/C++
General
Optimization
Preprocessor
Code Generatic
Language
Precompiled H
Output Files
Browse Informa
Advanced
Command Line
Linker
General
Input
Manifest File
Debugging
System
Optimization
Embedded IDL
Advanced
Command Line
Manifest Tool
XML Document G
Output File $(OutDir)$(TargetName)$(TargetExt)
Show Progress Not Set
Version
Enable Incremental Linking Yes (/INCREMENTAL)
Suppress Startup Banner Yes (/NOLOGO)
Ignore Import Library No
Register Output No
Per-user Redirection No
Additional Library Directories F:\Storage\opencv-3.0.0\opencv\build\x86\vc10\staticlib
Link Library D
Use Library De
Link Status
Prevent Dll Bi
Treat Linker W
Force File Out
Create Hot Pa
Specify Sectio
Additional Libra
Allows the user t
Additional Library Directories
F:\Storage\opencv-3.0.0\opencv\build\x86\vc10\staticlib
Inherited values:
Inherit from parent or project defaults
Macros>>
OK
Cancel
мена
Применить


C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7600]
(c) Корпорация Майкрософт (Microsoft Corp.), 2009. Все права защищены.
C:\Users\alexey>f:
F:\>cd Storage\opencv-3.0.0\opencv\build\x86\vc10\staticlib
F:\Storage\opencv-3.0.0\opencv\build\x86\vc10\staticlib>dir /b *d.lib
IlmImfd.lib
libjasperd.lib
libjpegd.lib
libpngd.lib
libtiffd.lib
libwebpd.lib
opencv_calib3d300d.lib
opencv_core300d.lib
opencv_features2d300d.lib
opencv_flann300d.lib
opencv_highgui300d.lib
opencv_imgcodecs300d.lib
opencv_imgproc300d.lib
opencv_ml300d.lib
opencv_objdetect300d.lib
opencv_photo300d.lib
opencv_shape300d.lib
opencv_stitching300d.lib
opencv_superres300d.lib
opencv_ts300d.lib
opencv_video300d.lib
opencv_videoio300d.lib
opencv_videostab300d.lib
zlibd.lib
F:\Storage\opencv-3.0.0\opencv\build\x86\vc10\staticlib>


cvcourse_opencv_intro Property Pages
Configuration: Active(Debug)
Platform: Active(Win32)
Configuration Manager...
Common Properties
Configuration Properties
General
Debugging
VC++ Directories
C/C++
Linker
General
Input
Manifest File
Debugging
System
Optimization
Embedded IDL
Advanced
Command Line
Manifest Tool
XML Document Gene
Browse Information
Build Events
Custom Build Step
Additional Dependencies
ippicvmt.lib;IlmImfd.lib;libjasperd.lib;libjpegd.lib;libpngd.li
Ignore All Default Libraries
Ignore Specific Default Libraries
Module Definition File
Additional Dependencies
comctl32.lib
ippicvmt.lib
<- может не потребоваться для OpenCV 2.4.9
IlmImfd.lib
libjasperd.lib
libjpegd.lib
libpngd.lib
Inherited values:
kernel32.lib
user32.lib
gdi32.lib
winspool.lib
comdlg32.lib
Inherit from parent or project defaults
Macros>>
OK
Cancel
Additional Dependencies
Specifies additional items to add to the link command line [i.e. kernel32.lib]
OK
Отмена
Применить


cvcourse_opencv_intro Property Pages
Configuration: Active(Debug)
Platform: Active(Win32)
Configuration Manager...
Common Properties
Configuration Properties
General
Debugging
VC++ Directories
C/C++
General
Optimization
Preprocessor
Code Generation
Language
Precompiled Heade
Output Files
Browse Information
Advanced
Command Line
Linker
General
Input
Manifest File
Debugging
System
Optimization
Embedded IDL
Advanced
Enable String Pooling
Enable Minimal Rebuild Yes (/Gm)
Enable C++ Exceptions Yes (/EHsc)
Smaller Type Check No
Basic Runtime Checks Both (/RTC1, equiv. to /RTCsu) (/RTC1)
Runtime Library Multi-threaded Debug (/MTd)
Struct Member Alignment Default
Buffer Security Check Yes (/GS)
Enable Function-Level Linking
Enable Enhanced Instruction Set Not Set
Floating Point Model Precise (/fp:precise)
Enable Floating Point Exceptions
Create Hotpatchable Image
Enable String Pooling
Enables the compiler to create a single read-only copy of identical strings in the program image and in memory during execution, resulting in smaller programs, an optimization called string pooling. /O1, /O2, and /ZI automa...
OK
Отмена
Применить

```
#include "stdafx.h"
#include <opencv2\opencv.hpp>

int _tmain(int argc, _TCHAR* argv[])
{
     cv::Mat img = cv::Mat::zeros(300, 300, CV_8UC3);
     cv::putText(img, "Hello, OpenCV!", cv::Point(10, 50),
                cv::FONT_HERSHEY_SIMPLEX, 1, cv::Scalar(0, 255, 0), 2);
     cv::imshow("img", img);
     cv::waitKey();
     return 0;
}
```


img
Hello, OpenCV!

Быстрый старт

1. Устанавливаем компилятор C/C++, Python 2.7.x (http://python.org), NumPy (http://numpy.scipy.org), cmake (http://cmake.org)
2. Клонируем репозиторий с github или качаем архив, например https://github.com/Itseez/opencv/tree/3.0.0, и аналогично opencv_contrib (только для OpenCV 3.x), строим OpenCV и *не инсталируем его*!

```
cmake –D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib/modules …
```

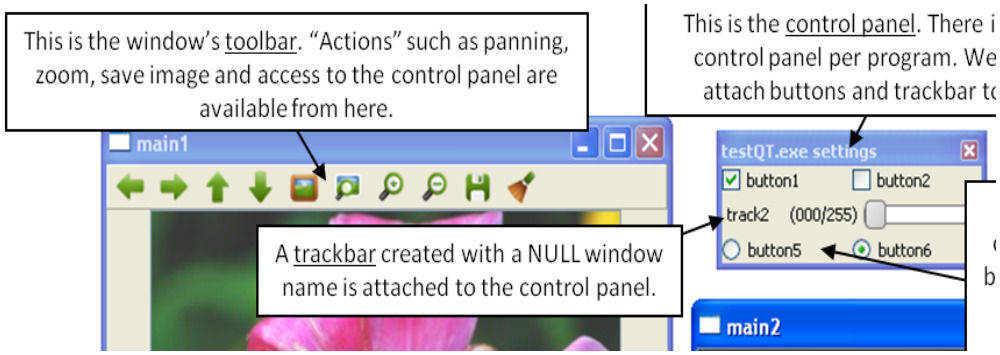
3. Создаем каталог с проектом и кладем туда следующий CMakeList.txt:

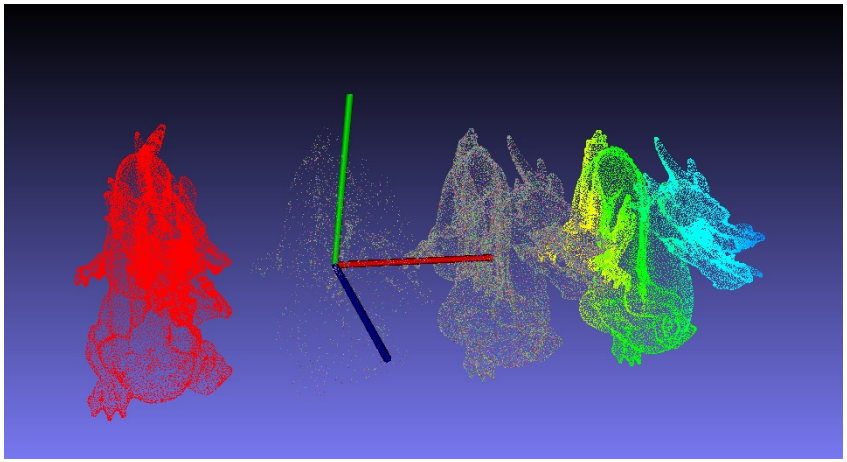
```
cmake_minimum_required(VERSION 2.8)
project(myopencv_sample)
find_package(OpenCV REQUIRED)
include_directories(${OpenCV_INCLUDE_DIRS})
set(the_target "myopencv_sample")
add_executable(${the_target} main.cpp) # add other .cpp
                                       # and .h files here
target_link_libraries(${the_target} ${OpenCV_LIBS})
```

4. Создаем main.cpp (см. дальше). Можно взять один из готовых примеров из opencv/samples/cpp.
5. Указываем cmake, где найти OpenCVConfig.cmake и генерируем проект или Makefile’s.
6. Открываем сгенерированный проект, строим.

HighGUI (=ui+imgcodec+videoio)

- Окна с “памятью”
- Обработка нажатий клавиш.
- Обработка событий от мыши.
- Слайдеры.
- Чтение/запись изображений
- Чтение/запись видео
- В случае наличия Qt – много дополнительных средств (тулбар, кнопки, зум, значения пикселей …)
- См. также модуль VIZ для визуализации 3D данных: http://habrahabr.ru/company/itseez/blog/217021/





Key OpenCV Classes

`Point_`	Template 2D point class
`Point3_`	Template 3D point class
`Size_`	Template size (width, height) class
`Vec`	Template short vector class
`Matx`	Template small matrix class
`Scalar`	4-element vector
`Rect`	Rectangle
`Range`	Integer value range
`Mat`	2D or multi-dimensional dense array (can be used to store matrices, images, histograms, feature descriptors, voxel volumes etc.)
`SparseMat`	Multi-dimensional sparse array
`Ptr`	Template smart pointer class

Point, Vec

```
typedef Point_<int> Point2i;
typedef Point2i Point;
typedef Point_<float> Point2f;
typedef Point_<double> Point2d;
```

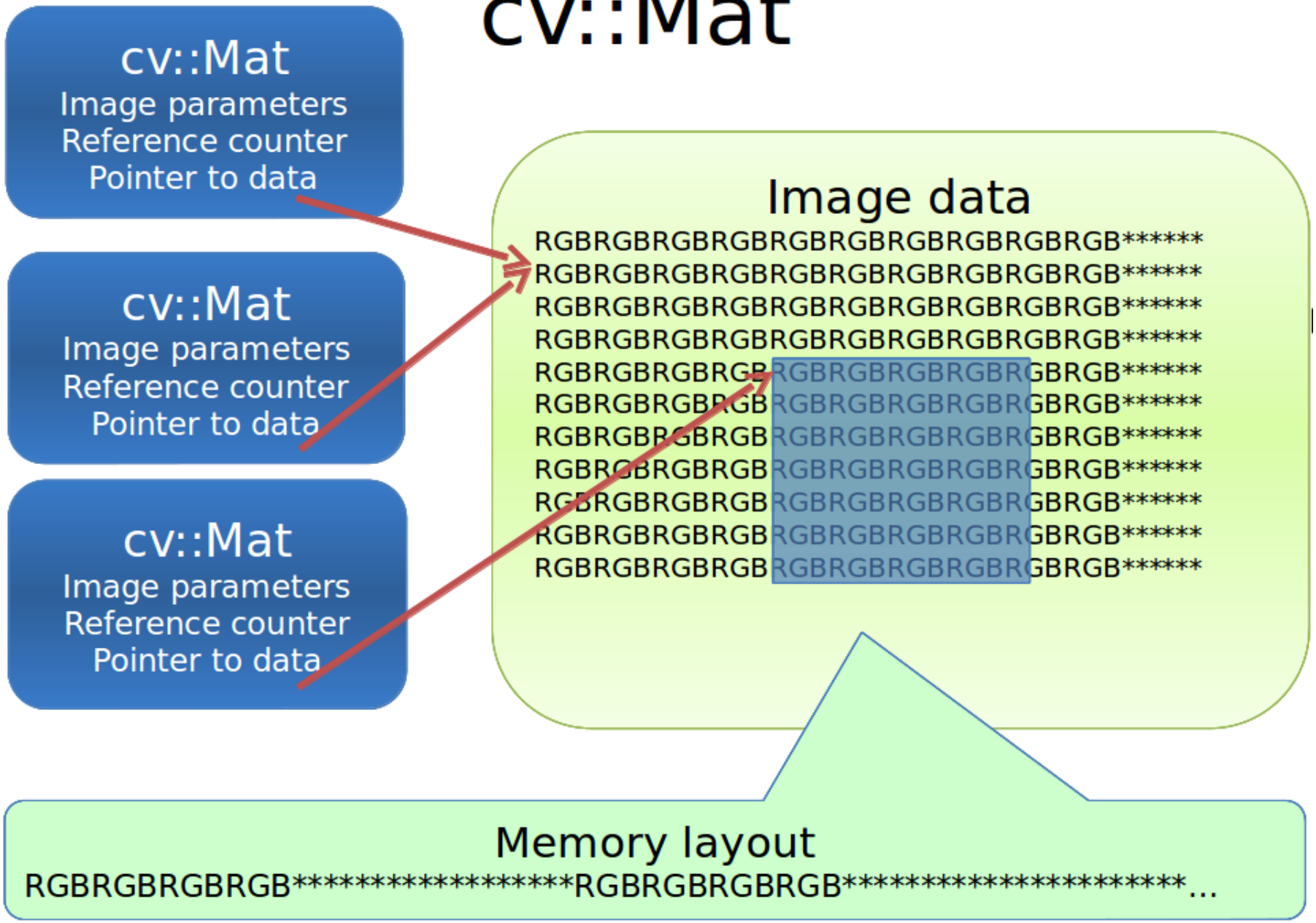
Пример:

```
Point2f a(0.3f, 0.f), b(0.f, 0.4f);
Point pt = (a + b)*10.f;
cout << pt.x << endl;
```

```
typedef Vec<uchar, 3> Vec3b;
typedef Vec<short, 3> Vec3s;
typedef Vec<int, 3> Vec3i;
typedef Vec<float, 3> Vec3f;
```

Используйте operator[] для доступа к элементам.

cv::Mat


cv::Mat
Image parameters
Reference counter
Pointer to data
cv::Mat
Image parameters
Reference counter
Pointer to data
cv::Mat
Image parameters
Reference counter
Pointer to data
Image data
RGBRGBRGBRGBRGBRGBRGBRGBRGB******
RGBRGBRGBRGBRGBRGBRGBRGBRGBRGB******
RGBRGBRGBRGBRGBRGBRGBRGBRGBRGB******
RGBRGBRGBRGBRGBRGBRGBRGBRGBRGB******
RGBRGBRGBRGBRGBRGBRGBRGBRGBRGB******
RGBRGBRGBRGBRGBRGBRGBRGBRGBRGB******
RGBRGBRGBRGBRGBRGBRGBRGBRGBRGB******
RGBRGBRGBRGBRGBRGBRGBRGBRGBRGB******
RGBRGBRGBRGBRGBRGBRGBRGBRGBRGB******
RGBRGBRGBRGBRGBRGBRGBRGBRGBRGB******
Memory layout
RGBRGBRGBRGB**********RGBRGBRGBRGB**********...

Работа с матрицами

Создание матрицы:

- Mat A(3,3,CV_32F) -- создает матрицу размера 3x3 с элементами типа float
- Mat A = Mat::zeros(3,3,CV_32F) -- матрица будет заполнена нулями
- Mat A = Mat::eye(3,3,CV_32F) -- единичная матрица
- A.rows, A.cols -- число строк и столбцов

Допустимые типы:

- 8U, 8S, 16U, 16S, 32S, 32F, 64F, 8U=8UC1, 8UC2, 8UC3, 8UC4...

Доступ к элементам:

- A.at<float>(i,j) -- доступ к элементу $A_{i,j}$ матрицы типа CV_32F
- A.at<Vec3b>(i,j)[0] -- доступ 1-му каналу элемента $A_{i,j}$ матрицы типа CV_8UC3

Операции:

- A.clone() -- возвращает копию матрицы
- A.setTo(...) -- устанавливает заданное значение для всех элементов (можно передавать, например, Scalar(255,255,255) для трехканальных матриц)
- A.copyTo(B) -- копирует матрицу A в матрицу B
- A.convertTo(...) -- конвертирует тип матрицы, со скалированием и сдвигом значений (опционально)

Работа с матрицами

Получение подматриц:

- A.row(...), A.col(...), A.rowRange(...), A.colRange(...), A(Rect(...))

Арифметические поэлементые операции:

- add(...), subtract(...), multiply(...), divide(...)

Бинарные поэлементые операции:

- bitwise_and(...), bitwise_or(...), bitwise_not(...), bitwise_xor(...)

Минимум и максимум:

- min(...), max(...), minMaxLoc(...)

Операции с каналами:

- split(...), merge(...), mixChannels(...)

Сумма, среднее...

- sum(...), mean(...), meanStdDev(), norm(...)

Алгебра:

- solve(...), det(...)

I/O

Writing and reading raster images

```
imwrite("myimage.jpg", image);
Mat image_color_copy = imread("myimage.jpg", 1);
Mat image_grayscale_copy = imread("myimage.jpg", 0);
```

The functions can read/write images in the following formats: ***BMP (.bmp), JPEG (.jpg, .jpeg), TIFF (.tif, .tiff), PNG (.png), PBM/PGM/PPM (.p?m), Sun Raster (.sr), JPEG 2000 (.jp2).*** *Every format supports 8-bit, 1- or 3-channel images. Some formats (PNG, JPEG 2000) support 16 bits per channel.*

Reading video from a file or from a camera

```
VideoCapture cap;
if(argc > 1) cap.open(string(argv[1])); else cap.open(0);
Mat frame; namedWindow("video", 1);
for(;;) {
   cap >> frame; if(!frame.data) break;
   imshow("video", frame); if(waitKey(30) >= 0) break;
}
```

imread, imshow

C++: Mat **imread**(const string& **filename**, int **flags**=1)

- **>0** Return a 3-channel color image -- CV_LOAD_IMAGE_COLOR
- **=0** Return a grayscale image -- CV_LOAD_IMAGE_GRAYSCALE
- **<0** Return the loaded image as is (with alpha channel) -- CV_LOAD_IMAGE_ANYDEPTH

The function determines the type of an image by the content, not by the file extension.
In the case of color images, the decoded images will have the channels stored in B G R order.

C++: void **imshow**(const string& **winname**, InputArray **mat**)

- **winname** – Name of the window.
- **image** – Image to be shown.

The function may scale the image, depending on its depth:

- If the image is 8-bit unsigned, it is displayed as is.
- If the image is 16-bit unsigned or 32-bit integer, the pixels are divided by 256. That is, the value range [0,255*256] is mapped to [0,255].
- If the image is 32-bit floating-point, the pixel values are multiplied by 255. That is, the value range [0,1] is mapped to [0,255].

don't forget to use waitKey() !!!

cv::Mat – многомерный многоканальный массив

cv::Mat A(h, w, CV_8UC3);
- Размеры, step
- Счетчик ссылок (=1)
- Указатель на данные

cv::Mat B = A;
- Размеры, step
- Счетчик ссылок (=2)
- Указатель на данные

cv::Mat C=A(roi);
- Размеры ROI, step
- Счетчик ссылок (=3)
- Указатель на данные

Расположение в памяти матрицы C

RGBRGBRGBRGB**********RGBRGBRGBRGB**********...

cv::Mat и std::vector

std::vector<Point3f> v(N);

XYZXYZXYZXYZXYZXYZXYZXYZXYZXYZXYZXYZXYZXYZ

Массив из N точек

cv::Mat a(v);

- Размеры (Nx1), step(=12)
- Счетчик ссылок (отсутствует)
- Указатель на данные

XYZ
XYZ
…
XYZ

Nx1 3-канальное изображение

core: операции над матрицами, “мини Matlab”

Mat::zeros, Mat::eye, Mat::ones, Mat::setTo, randu, randn – заполнение/ инициализация матриц элементов, объединение и выделение отдельных каналов.
Mat::operator (), Mat::row, Mat::col, Mat::rowRange, Mat::colRange, Mat::diag, Mat::reshape – выделение частей матриц и изменение формы без копирования
Mat::copyTo, Mat::clone, Mat::repeat, vconcat, hconcat, flip, transpose, randShuffle, split, merge, mixChannels – копирование и различные перемешивания
Mat::convertTo, normalize – преобразование к другому диапазону и/или к другому типу данных
add, subtract, multiply, divide, absdiff – поэлементные арифметические операции
bitwise_and, bitwise_or, bitwise_xor, bitwise_not – логические операции
compare, min, max – поэлементное сравнение, минимум, максимум
sum, mean, meanStdDev, norm, minMaxLoc – сбор статистики по матрице, сравнение матриц
gemm, Mat::dot, Mat::cross, solve, eigen, SVD, determinant, trace, solvePoly, solveLP, MinProblemSolver – линейная алгебра, нахождение корней полиномов, решение задач оптимизации
exp, log, sqrt, pow, cartToPolar, polarToCart – стандартные поэлементные математические операции
reduce, sort, sortIdx – операции над строками и столбцами
dft, dct – дискретные ортогональные преобразования

http://docs.opencv.org/opencv_cheatsheet.pdf -

здесь перечислено гораздо больше функций

Перебор элементов

```
// оцениваем “резкость” в выбранном ROI,
// например для реализации автофокуса
float contrast = 0.f;
for(int i = 0; i < subM.rows; i++) {
    const uchar* ptr = subM.ptr<uchar>(i);
    for(int j = 0; j < subM.cols; j++, ptr++) {
        int dx = ptr[1] – ptr[-1], dy = ptr[subM.step] – ptr[-subM.step];
        contrast += sqrtf((float)(dx*dx + dy*dy));
}

// вариант с итераторами
Mat_<uchar>::iterator it= subM.begin<uchar>(),
                      itEnd = subM.end<uchar>();
float contrast = 0.f;
for(; it != itEnd; ++it) {
   uchar* ptr = &(*it);
   int dx = ptr[1] – ptr[-1], dy = ptr[subM.step] – ptr[-subM.step];
   contrast += sqrtf((float)(dx*dx + dy*dy));
}
```

FileStorage: запись/чтение структурированных данных

// Записываем данные

```
1. { FileStorage fs("test.yml", FileStorage::WRITE);
2. fs << “intval” << 5 << “realval” << 3.1 << “str” << “ABCDEFGH”;
3. fs << “mtx” << Mat::eye(3,3,CV_32F);
4. fs << “mylist” << “[“ << 1 << 2 << 3 << 4 << 5 << “]”;
5. fs << “date” << “{:” << “month” << 12 << “day” << 31 << “year” << 2015 << “}”;
6. const uchar arr[] = {0, 1, 1, 0, 1, 1, 0, 1};
7. fs << “bitmask” << “[:”; fs.writeRaw(“u”, arr, 8);
8. fs << “]”; }
```

// И считываем их обратно

```
1. { FileStorage fs("test.yml", FileStorage::READ);
2. int intval = (int)fs[“intval”]; double realval = (double)fs[“realval”];
3. string str = (string)fs[“str”];
4. Mat mtx; fs[“mtx”] >> mtx;
5. vector<int> mylist; FileNode mylist_node = fs[“mylist”];
6. size_t n = mylist_node.size(); FileNodeIterator mylist_it = mylist_node.begin();
7. for( i = 0; i < n; i++, ++it) { mylist.push_back((int)*it); }
8. FileNode dn = fs[“date”]; int month = (int)dn[“month”], day=(int)dn[“day”],
   year=(int)dn[“year”];
9. vector<uchar> bitmask; fs[“bitmask”] >> bitmask; }
```

Функциональность основного OpenCV

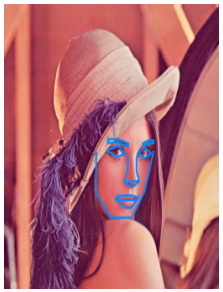
Базовая функциональность

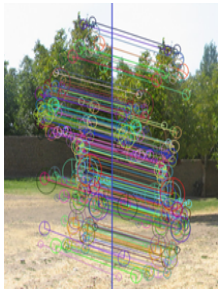
A + B
Ax = B
FFT(A)
<?xml>…

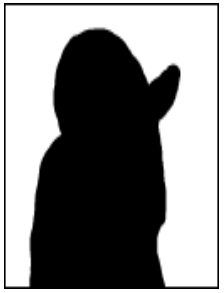
Обработка изображений


Фильтрация

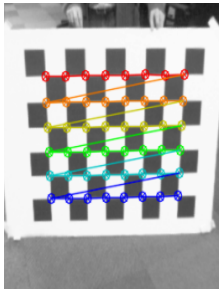

Трансформации

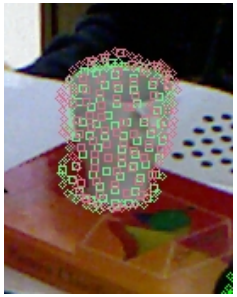

Ребра, контурный анализ

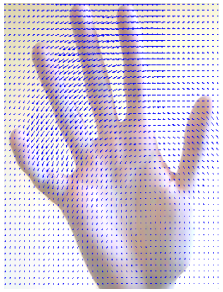

Особые точки


Сегментация

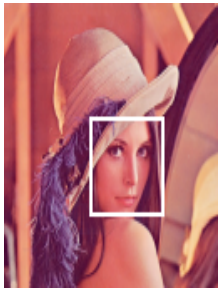
Видео, Стерео, 3D


Калибрация камер

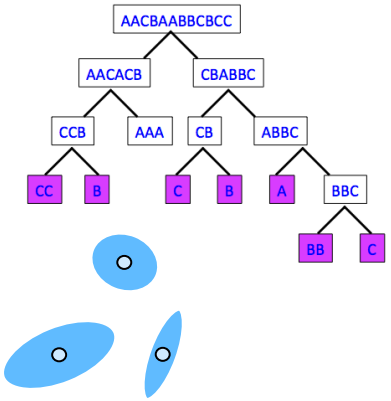

Вычисление положения в пространстве


Оптический поток


Построение карты глубины


Нахождение объектов

Машинное обучение



Обработка изображений

cvtColor

Converts an image from one color space to another.

void **cvtColor**(InputArray **src**, OutputArray **dst**, int **code**, int **dstCn**=0)

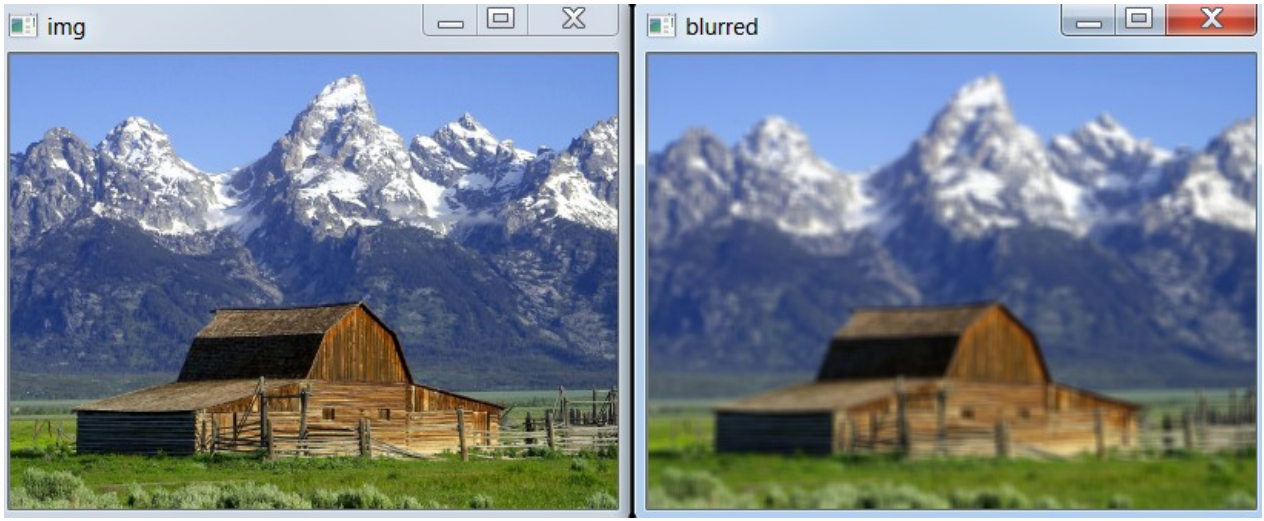
- **src** – input image: 8-bit unsigned, 16-bit unsigned (CV_16UC...), or single-precision floating-point.
- **dst** – output image of the same size and depth as src.
- **code** – color space conversion code, e.g. **cv::COLOR_BGR2GRAY**
- **dstCn** – number of channels in the destination image; if the parameter is 0, the number of the channels is derived automatically from src and code .

blur

Blurs an image using the normalized box filter.

void **blur**(InputArray **src**, OutputArray **dst**, Size **ksize**, Point **anchor**=Point(-1,-1), int **borderType**=BORDER_DEFAULT)

- **src** – input image; it can have any number of channels, which are processed independently, but the depth should be CV_8U,CV_16U, CV_16S, CV_32F or CV_64F.
- **dst** – output image of the same size and type as src.
- **ksize** – blurring kernel size.
- **anchor** – anchor point; default value Point(-1,-1) means that the anchor is at the kernel center.
- **borderType** – border mode used to extrapolate pixels outside of the image.

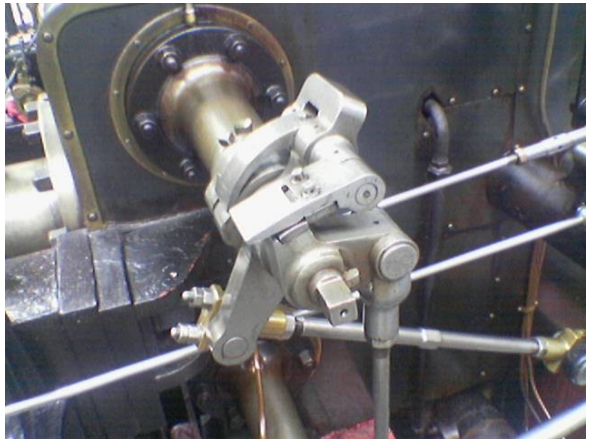


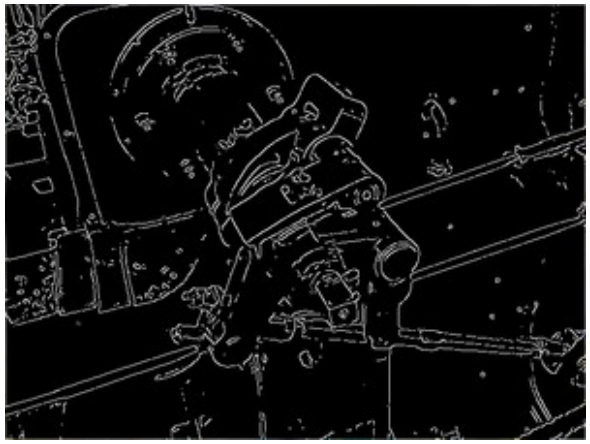
Canny

Finds edges in an image using the [Canny86] algorithm.

void **Canny**(InputArray **image**, OutputArray **edges**, double **threshold1**, double **threshold2**, int **apertureSize**=3, bool **L2gradient**=false)

- **image** – single-channel 8-bit input image.
- **edges** – output edge map; it has the same size and type as image .
- **threshold1** – first threshold for the hysteresis procedure.
- **threshold2** – second threshold for the hysteresis procedure.
- **apertureSize** – aperture size for the **Sobel()** operator.



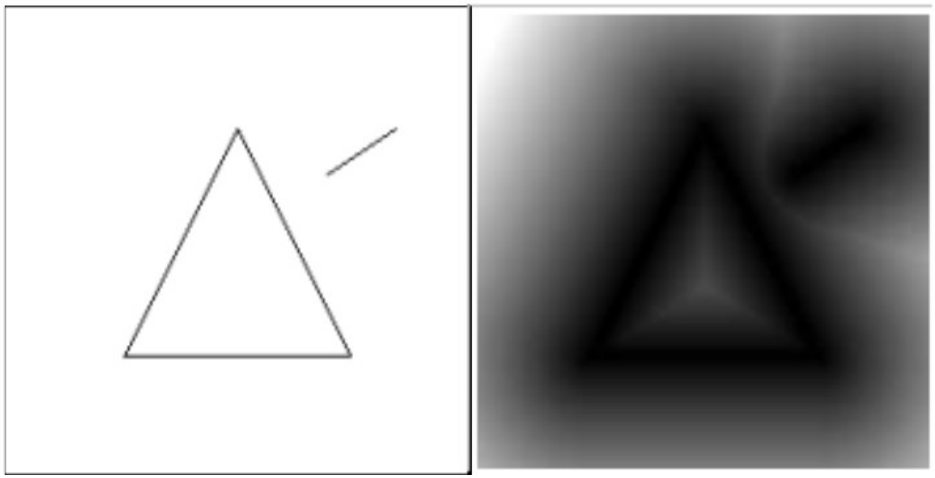


distanceTransform

Calculates the distance to the closest zero pixel for each pixel of the source image.

void **distanceTransform**(InputArray **src**, OutputArray **dst**, int **distanceType**, int **maskSize**)

- **src** – 8-bit, single-channel (binary) source image.
- **dst** – Output image with calculated distances. It is a 32-bit floating-point, single-channel image of the same size as src .
- **distanceType** – Type of distance. It can be CV_DIST_L1, CV_DIST_L2 , or CV_DIST_C .
- **maskSize** – Size of the distance transform mask. It can be 3, 5, or CV_DIST_MASK_PRECISE (the latter option is only supported by the first function). In case of the CV_DIST_L1 or CV_DIST_C distance type, the parameter is forced to 3 because a mask gives the same result as or any larger aperture.



equalizeHist

Equalizes the histogram of a grayscale image.

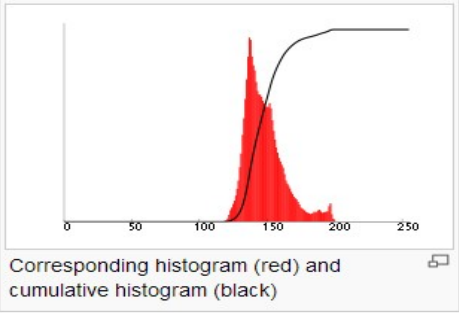
void **equalizeHist**(InputArray **src**, OutputArray **dst**)

- **src** – Source 8-bit single channel image.
- **dst** – Destination image of the same size and type as src.

1. Calculate the histogram H(i) for src.
2. Normalize the histogram so that the sum of histogram bins is 255.
3. Compute the integral of the histogram:
   F(i)=sum(H(j), 0<=j<i)
4. Transform the image using F(i) as a look-up table:
   dst(x,y)=F(src(x,y))

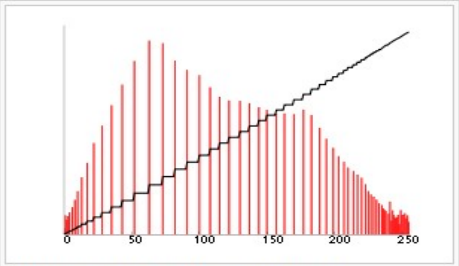
аналогия: цвет пикселя -- СВ, находим такое преобразование СВ, которое делает распределение равномерным

An unequalized image



Corresponding histogram (red) and cumulative histogram (black)

The same image after histogram equalization



Corresponding histogram (red) and cumulative histogram (black)

morphologyEx

- Performs advanced morphological transformations.
- void **morphologyEx**(InputArray **src**, OutputArray **dst**, int **op**, InputArray **kernel**, Point **anchor**=Point (-1,-1), int **iterations**=1, int **borderType**=BORDER_CONSTANT, const Scalar& **borderValue**=morphologyDefaultBorderValue())

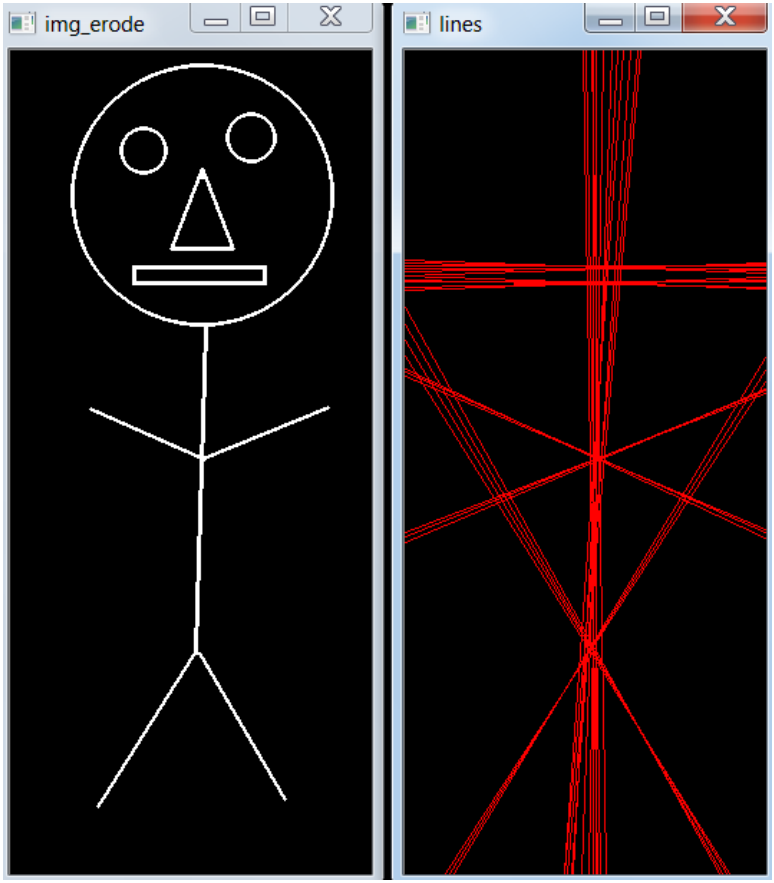
- **src** – Source image. The number of channels can be arbitrary. The depth should be one of CV_8U, CV_16U, CV_16S, CV_32F`or ``CV_64F.
- **dst** – Destination image of the same size and type as src .
- **element** – Structuring element.
- **op** – Type of a morphological operation that can be one of the following:
  - **MORPH_ERODE**, **MORPH_DILATE**, **MORPH_OPEN**, **MORPH_CLOSE**, **MORPH_GRADIENT**,
  - **iterations** – Number of times erosion and dilation are applied.
- **borderType** – Pixel extrapolation method.
- **borderValue** – Border value in case of a constant border. The default value has a special meaning.

HoughLines

Finds lines in a binary image using the standard Hough transform.

void **HoughLines**(InputArray **image**, OutputArray **lines**, double **rho**, double **theta**, int **threshold**, double **srn**=0, double **stn**=0)¶

- **image** – 8-bit, single-channel binary source image. The image may be modified by the function.
- **lines** – Output vector of lines. Each line is represented by a two-element vector (a,b). a is the distance from the coordinate origin (top-left corner of the image). b is the line rotation angle in radians.
- **rho** – Distance resolution of the accumulator in pixels.
- **theta** – Angle resolution of the accumulator in radians.
- **threshold** – Accumulator threshold parameter. Only those lines are returned that get enough votes.



watershed

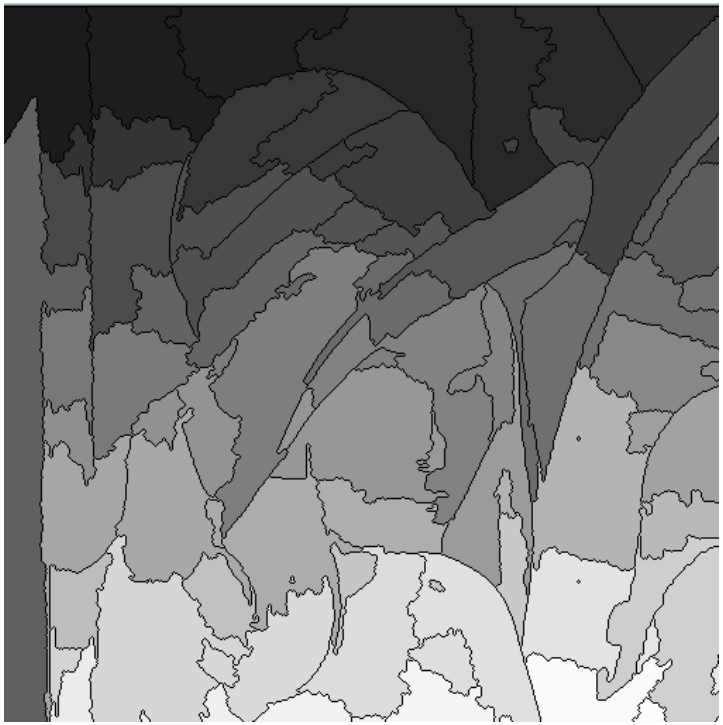
Performs a marker-based image segmentation using the watershed algorithm.

void **watershed**(InputArray **image**, InputOutputArray **markers**)

- **image** – Input 8-bit 3-channel image.
- **markers** – Input/output 32-bit single-channel image (map) of markers. It should have the same size as image.

Before passing the image to the function, you have to roughly outline the desired regions in the image markers with positive (>0) indices. So, every region is represented as one or more connected components with the pixel values 1, 2, 3, and so on.

watershed



matchTemplate

Compares a template against overlapped image regions.

void **matchTemplate**(InputArray **image**, InputArray **templ**, OutputArray **result**, int **method**)¶

- **image** – Image where the search is running. It must be 8-bit or 32-bit floating-point.
- **templ** – Searched template. It must be not greater than the source image and have the same data type.
- **result** – Map of comparison results. It must be single-channel 32-bit floating-point. If image is WxH and templ is w x h , then result is (W-w+1)x(H-h+1) .
- **method** – Parameter specifying the comparison method (see **docs.opencv.org !!!**).

CV_TM_SQDIFF: $$R(x,y) = \sum_{x',y'} (T(x',y') - I(x+x',y+y'))^2$$

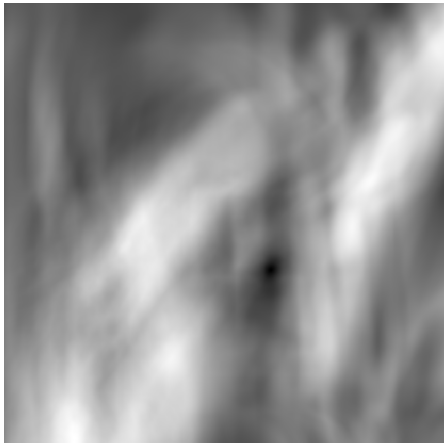
minMaxLoc

Finds the global minimum and maximum in an array.

void **minMaxLoc**(InputArray **src**, double* **minVal**, double* **maxVal**=0, Point* **minLoc**=0, Point* **maxLoc**=0, InputArray **mask**=noArray())

- **src** – input single-channel array.
- **minVal** – pointer to the returned minimum value; NULL is used if not required.
- **maxVal** – pointer to the returned maximum value; NULL is used if not required.
- **minLoc** – pointer to the returned minimum location (in 2D case); NULL is used if not required.
- **maxLoc** – pointer to the returned maximum location (in 2D case); NULL is used if not required.
- **mask** – optional mask used to select a sub-array.

matchTemplate & minMaxLoc



goodFeaturesToTrack

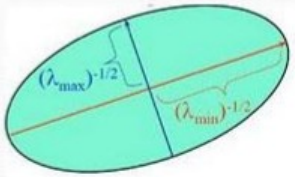
Нахождение угловых точек (Harris)

$$S(x,y) = \sum_u \sum_v (I(u,v) - I(u+x, v+y))^2$$

$$I(u+x, v+y) \approx I(u,v) + I_x(u,v)x + I_y(u,v)y$$

$$S(x,y) \approx \sum_u \sum_v (I_x(u,v)x + I_y(u,v)y)^2 \qquad S(x,y) \approx \begin{pmatrix} x & y \end{pmatrix} A \begin{pmatrix} x \\ y \end{pmatrix}$$

$$A = \sum_u \sum_v \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \langle I_x^2 \rangle & \langle I_x I_y \rangle \\ \langle I_x I_y \rangle & \langle I_y^2 \rangle \end{bmatrix}$$



1.Если λ1 ~ 0 & λ2 ~ 0 – однородная область.
2.Если λ1 ~ 0 & λ2 велико, то точка на ребре.
3.Если λ1 велико & λ2 велико, то точка является угловой.

Вместо подсчёта λi быстрее посчитать

$$M_c = \lambda_1 \lambda_2 - \kappa (\lambda_1 + \lambda_2)^2 = \det(A) - \kappa \operatorname{trace}^2(A)$$

0.04 - 0.15

goodFeaturesToTrack

Determines strong corners on an image.

void **goodFeaturesToTrack**(InputArray **image**, OutputArray **corners**, int **maxCorners**, double **qualityLevel**, double **minDistance**, InputArray **mask**=noArray(), int **blockSize**=3, bool **useHarrisDetector**=false, double **k**=0.04)¶

- **image** – Input 8-bit or floating-point 32-bit, single-channel image.
- **corners** – Output vector of detected corners.
- **maxCorners** – Maximum number of corners to return. If there are more corners than are found, the strongest of them is returned.
- **qualityLevel** – Parameter characterizing the minimal accepted quality of image corners. The parameter value is multiplied by the best corner quality measure, which is the minimal eigenvalue (see**cornerMinEigenVal()**) or the Harris function response (see **cornerHarris()**). The corners with the quality measure less than the product are rejected. For example, if the best corner has the quality measure = 1500, and the qualityLevel=0.01 , then all the corners with the quality measure less than 15 are rejected.
- **minDistance** – Minimum possible Euclidean distance between the returned corners.
- **mask** – Optional region of interest. If the image is not empty (it needs to have the type CV_8UC1 and the same size as image), it specifies the region in which the corners are detected.
- **blockSize** – Size of an average block for computing a derivative covariation matrix over each pixel neighborhood. See **cornerEigenValsAndVecs()** .
- **useHarrisDetector** – Parameter indicating whether to use a Harris detector (see **cornerHarris()**) or **cornerMinEigenVal()**.
- **k** – Free parameter of the Harris detector.

goodFeaturesToTrack

findContours

- Finds contours in a binary image.
- void **findContours**(InputOutputArray **image**, OutputArrayOfArrays **contours**, OutputArray **hierarchy**, int **mode**, int
- **method**)

- **image** – Source, an 8-bit single-channel image. Non-zero pixels are treated as 1's. Zero pixels remain 0's, so the image is treated as binary . You can use **compare()** , **inRange()** , **threshold()** , **adaptiveThreshold()** , **Canny()** , and others to create a binary image out of a grayscale or color one. The function modifies the image while extracting the contours.
- **contours** – Detected contours (type: vector<vector<Point> >). Each contour is stored as a vector of points.
- **hierarchy** – Optional output vector (type: vector<Vec4i>), containing information about the image topology.
- **mode** – Contour retrieval mode (if you use Python see also a note below).
  - **CV_RETR_CCOMP** retrieves all of the contours and organizes them into a two-level hierarchy. At the top level, there are external boundaries of the components. At the second level, there are boundaries of the holes. If there is another contour inside a hole of a connected component, it is still put at the top level.
- **method** – Contour approximation method (if you use Python see also a note below).
  - **CV_CHAIN_APPROX_SIMPLE** compresses horizontal, vertical, and diagonal segments and leaves only their end points. For example, an up-right rectangular contour is encoded with 4 points.

drawContours

Draws contours outlines or filled contours.

void **drawContours**(InputOutputArray **image**, InputArrayOfArrays **contours**, int **contourIdx**, const Scalar& **color**, int **thickness**=1, int **lineType**=8, InputArray **hierarchy**=noArray())

- **image** – Destination image.
- **contours** – All the input contours. Each contour is stored as a point vector.
- **contourIdx** – Parameter indicating a contour to draw. If it is negative, all the contours are drawn.
- **color** – Color of the contours.
- **thickness** – Thickness of lines the contours are drawn with. If it is negative (for example, thickness=CV_FILLED), the contour interiors are drawn.
- **lineType** – Line connectivity. See **line()** for details.
- **hierarchy** – Optional information about hierarchy. It is only needed if you want to draw only some of the contours.

integral

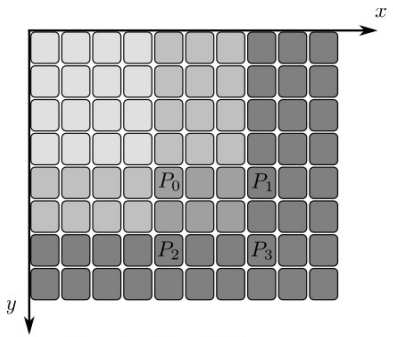
Calculates the integral of an image.

void **integral**(InputArray **src**, OutputArray **sum**, int **sdepth**=-1)

- **image** – input image as WxH, 8-bit or floating-point (32f or 64f).
- **sum** – integral image as (W+1)x(H+1), 32-bit integer or floating-point (32f or 64f).
- **sdepth** – desired depth of the integral and the tilted integral images, CV_32S, CV_32F, or CV_64F.

$$\mathtt{sum}(X,Y) = \sum_{x<X,y<Y} \mathtt{image}(x,y)$$

$$\sum_{x_1 \le x < x_2,\, y_1 \le y < y_2} \mathtt{image}(x,y) = \mathtt{sum}(x_2,y_2) - \mathtt{sum}(x_1,y_2) - \mathtt{sum}(x_2,y_1) + \mathtt{sum}(x_1,y_1)$$



$P_0 = \{y, x\} = \{4, 4\}$
$P_1 = \{y, x + w\} = \{4, 7\}$
$P_2 = \{y + h, x\} = \{6, 4\}$
$P_3 = \{y + h, x + w\} = \{6, 7\}$

аналогия: двухмерная СВ, плотность,
интегральная функция распределения

Задача адаптивной фильтрации

Задание

Цель в том, чтобы сгладить изображение, сохраняя при этом ребра. Один из возможных подходов -- адаптивная версия blur.

Необходимо реализовать описанный далее алгоритм, показать результаты работы и сдать код.

Алгоритм

1. Найти ребра на изображении с помощью метода Канни.

2. Построить карту расстояний D до найденных ребер.

3. В каждом пикселе [i,j] произвести фильтрацию усреднением с размером окна пропорциональным D [i,j]. Т.е. чем дальше от ребра, тем сильнее сглаживание.

4. Для ускорения вычислений следует воспользоваться интегральными изображениями (функция integral).

Входное изображение

Ожидаемый результат

Вопросы?