

Начинаем работать с библиотекой OpenCV

Вадим Писаревский,
ведущий инженер компании
Itseez



Содержание

- Общая информация, обзор
- Подготовка к работе
- Первые примеры
- Работа с матрицами
- Более сложная функциональность, примеры

Краткая справка

- Страница: <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 Itseez при активном участии сообщества.
- >10,000,000 загрузок (без учета трафика с github)
- Используется многими компаниями, организациями, ВУЗами, например в MagicLeap, NVidia, Intel, Google, Stanford ...



спонсоры

Архитектура и разработка OpenCV

Languages:

C/C++
Python
Java

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

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)

Acceleration Technologies:

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

Development:

Maintainers
Contributors

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, arm64
MIPS
...

OS'es:

Windows
Linux
Android
OSX
iOS

QNX, BSD, ...

Будет отдельный доклад

С чего начать?

[http://opencv.org:](http://opencv.org)

The screenshot shows the official OpenCV website. In the top navigation bar, there are links for ABOUT, DOWNLOADS, DOCUMENTATION, PLATFORMS, SUPPORT, CONTRIBUTE, REFERENCE, USER GUIDE, TUTORIALS, QUICK START, CHEAT SHEET, and WIKI. A red arrow points from the 'TUTORIALS' link to the 'WHAT'S NEW' section below. Another red arrow points from the 'WIKI' link to the 'LATEST DOWNLOADS' section. In the 'WHAT'S NEW' section, there are four items: 'OpenCV 2.4.3 released' (03/10/2012), 'OpenCV v2.4.3 is under way' (20/10/2012), 'OpenCV week in Italy' (06/10/2012), and 'Migration to git, 80% completed...' (25/07/2012). Below this, there's a 'LATEST TUTORIALS' section featuring 'Using OpenCV with gcc and CMake' (GCC logo) and 'Installation in iOS' (iOS 4 logo). The bottom of the page includes a footer with a link to the 'Tutorialspoint.com' site.

Читаем уроки,
Распечатываем
cheatsheet.

Ссылка WIKI ведет на
сайт для разработчиков
<http://code.opencv.org>

Качаем

Другие сайты OpenCV
(см. дальше)

Сайт с документацией

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

The screenshot shows a web browser window displaying the OpenCV 3.0.0-dev documentation. The title bar reads "docs.opencv.org/master/". The main content area features the OpenCV logo and the text "OpenCV 3.0.0-dev" above "Open Source Computer Vision". Below this is a navigation menu with tabs: "Main Page" (which is selected), "Related Pages", "Modules", "Namespaces", "Classes", "Files", and "Examples". A red arrow points from the text "Поиск по документации." to the search bar, which contains the placeholder "Search". The main content area is titled "OpenCV modules" and lists several sections:

- [Introduction](#)
- [OpenCV Tutorials](#)
- [OpenCV-Python Tutorials](#)
- [Frequently Asked Questions](#)
- [Bibliography](#)
- Main modules:
 - [core. Core functionality](#)
 - [imgproc. Image processing](#)
 - [imgcodecs. Image file reading and writing](#)
 - [videoio. Media I/O](#)
 - [highgui. High-level GUI](#)
 - [video. Video Analysis](#)
 - [calib3d. Camera Calibration and 3D Reconstruction](#)

Поиск по
документации.

Задать вопрос, найти ответ

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

The screenshot shows the 'Questions - OpenCV Q&A Forum' page. At the top, there's a navigation bar with links for 'OpenCV / OpenCV' and 'Questions - OpenCV Q&A Forum'. Below the navigation is a yellow banner with the text 'First time here? Check out the FAQ!' and three buttons: 'tags', 'people', and 'badges'. A red arrow points from the text 'Начинаем с FAQ' to this banner.

The main content area displays a list of questions:

- Building opencv framework los6 - dynamic link pb
- Having weird issues drawing lines on android
- Would Open CV help me find an object in 3D image
- Why this picture is classified to Upperbody?
- Image hashing implantation?
- What is the best way to find bounding box for binary mask?
- error (0xc150002) Application was unable to start correctly!
- something about boostsvm

Below each question, there are buttons for 'no votes', 'no answers', and 'no views', along with a timestamp (e.g., '2 hours ago', '4 hours ago').

To the right of the questions is a sidebar titled 'Contributors' which lists several users with their profile pictures and a grid of colorful icons representing different skills or interests.

At the bottom right of the sidebar is a section titled 'Tags' which lists various tags used in the questions, such as 'opencv' (x 10), 'Android' (x 4), 'error' (x 3), 'make' (x 2), 'documentation' (x 2), 'image processing' (x 2), 'opencv2-4.3' (x 2), 'python' (x 2), 'SURF' (x 2), '2.4.2' (x 1), '2.4.3' (x 1), 'adebook' (x 1), 'ai' (x 1), 'binary' (x 1), 'boot' (x 1), 'boundary' (x 1), and 'cameras' (x 1). A red arrow points from the text 'теги' to this 'Tags' section.

Начинаем с FAQ

1. Поискать ответы
2. Задать свой вопрос

теги

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 инсталлируем его!*

```
cmake -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib/modules ...
```

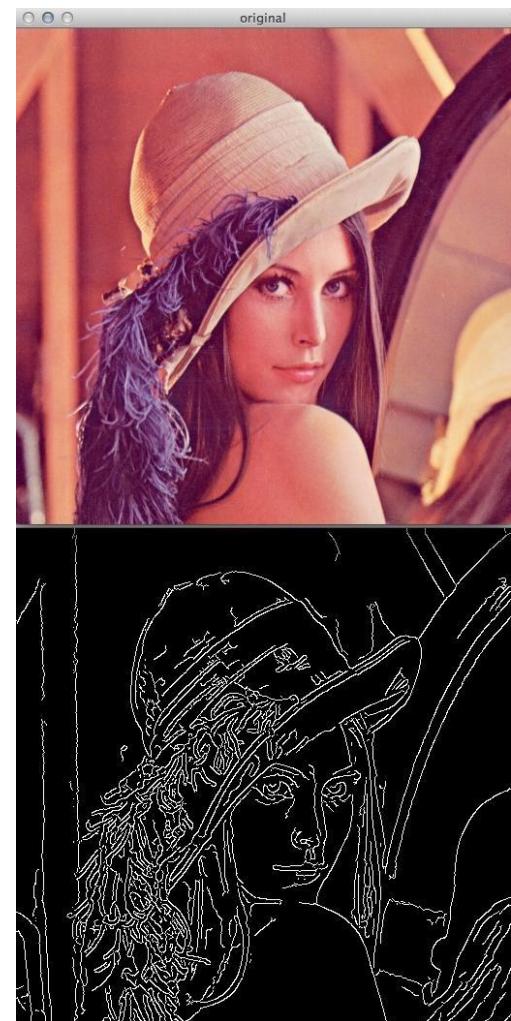
3. Создаем каталог с проектом и кладем туда следующий CMakeList.txt:

```
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.

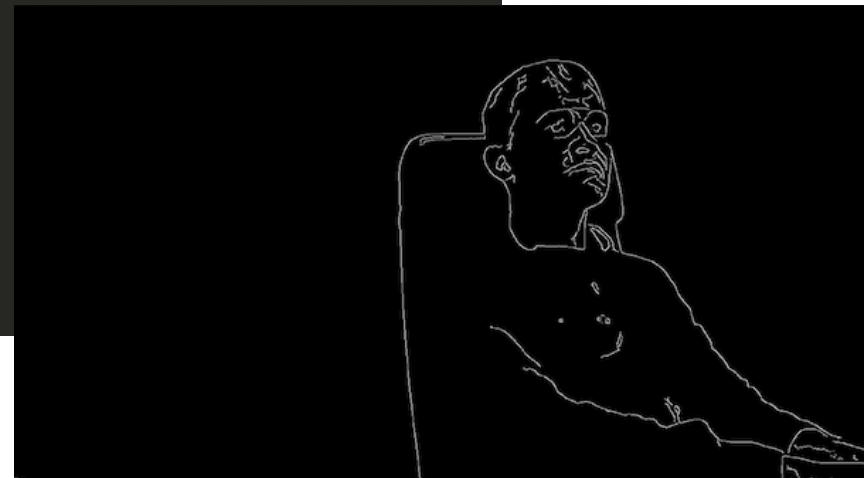
Первая программа: находим ребра

```
1 #include "opencv2/opencv.hpp"
2
3 using namespace cv;
4
5 int main(int argc, char** argv)
6 {
7     Mat img, gray, edges;
8     img = imread(argv[1], 1);
9     imshow("original", img);
10
11    cvtColor(img, gray, COLOR_BGR2GRAY);
12    GaussianBlur(gray, gray, Size(7, 7), 1.5);
13    Canny(gray, edges, 0, 50);
14
15    imshow("edges", edges);
16    imwrite("result.png", edges);
17    waitKey();
18
19    return 0;
20 }
```



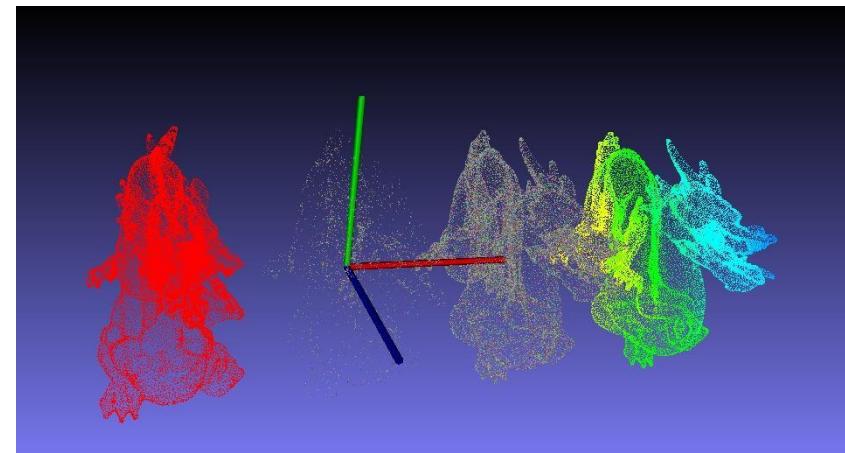
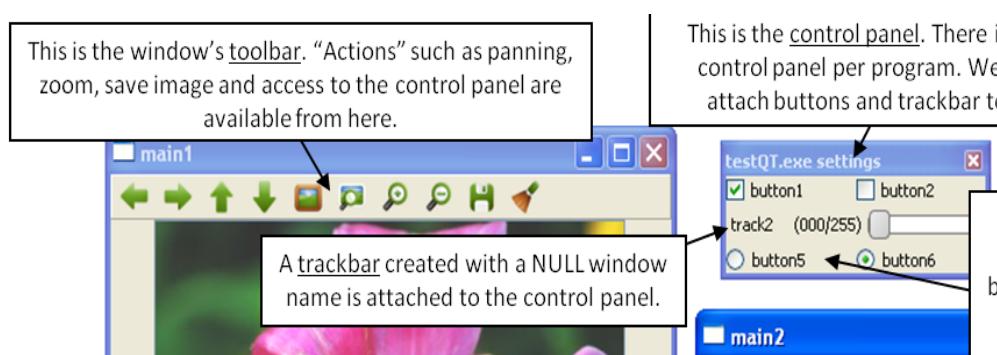
Вторая программа: добавляем видео

```
1 #include "opencv2/opencv.hpp"
2
3 using namespace cv;
4
5 int main(int argc, char** argv)
6 {
7     Mat img, gray, edges;
8     // Use "VideoCapture cap(argv[1])" to grab from video file
9     VideoCapture cap(0);
10
11    for(;;)
12    {
13        cap >> img;
14        if(img.empty()) break;
15
16        cvtColor(img, gray, COLOR_BGR2GRAY);
17        GaussianBlur(gray, gray, Size(7, 7), 1.5);
18        Canny(gray, edges, 0, 50);
19
20        imshow("edges", edges);
21        if(waitKey(30) >= 0) break;
22    }
23
24    return 0;
25 }
```



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/>



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

Элементы/Пиксели

Расположение в памяти матрицы С

RGBRGBRGBRGBB*****RGBRGBRGBRGBB*****

cv::Mat и std::vector

Массив из N точек

cv::Mat a(v);

- Размеры ($N \times 1$), step(=12)
 - Счетчик ссылок
(отсутствует)
 - Указатель на данные



XYZ
XYZ
...
XYZ

Nx1 3-канальное изображение

Работаем с матрицами

```
1 // Создаем 8-битное одноканальное изображение 640x480, заполненное нулями
2 Mat M = Mat::zeros(480, 640, CV_8UC1);
3
4 Rect roi(100, 200, 20, 20); // Определяем ROI
5 Mat subM = M(roi); // выделяем ROI в отдельную матрицу без копирования
6 subM.at<uchar>(y,x) = 255; // изменяем пиксель в строке y и столбце x ROI
7 // т.е. (x+100, y+200) в исходном изображении
8
9 float a = CV_PI/3;
10 // задаем матрицу 2x2 по элементам
11 Mat R = (Mat<float>(2, 2) << cos(a), -sin(a), sin(a), cos(a));
12
13 // альтернативный способ (без копирования данных)
14 float iRdata[] = {cos(a), sin(a), -sin(a), cos(a)};
15 Mat iR = Mat(2, 2, CV_32F, iRdata);
16 CV_Assert(norm(R*iR, Mat::eye(3, 3, CV_32F), NORM_L2) <= 0.01);
17
```

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 структур

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

Функциональность основного OpenCV

Базовая функциональность

$A + B$
 $Ax = B$
 $\text{FFT}(A)$
 $<?xml>...$



Фильтрация

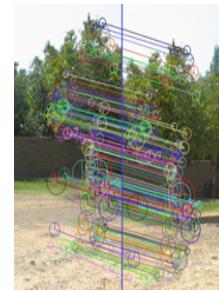


Трансформации

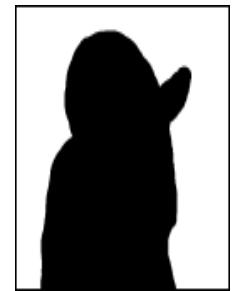
Обработка изображений



Ребра,
контурный
анализ

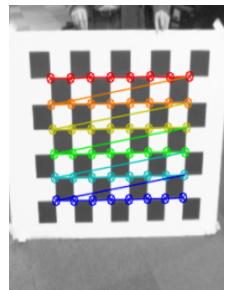


Особые точки

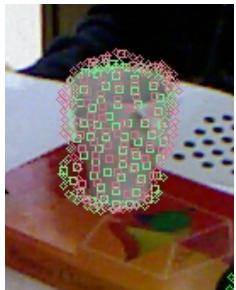


Сегментация

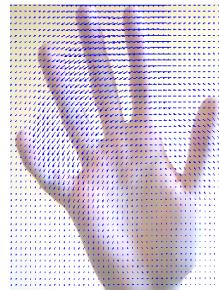
Видео, Стерео, 3D



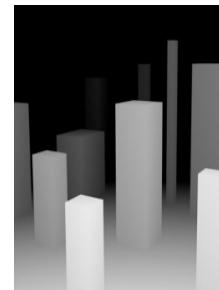
Калибровка
камер



Вычисление
положения в
пространстве



Оптический
поток

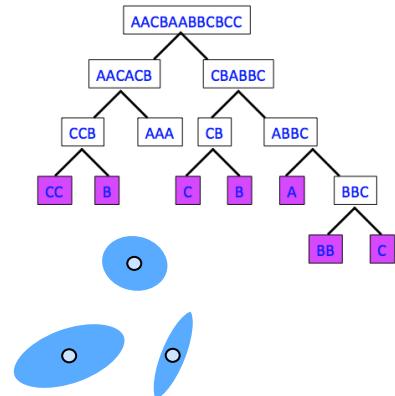


Построение
карты глубины



Нахождение
объектов

Машинное обучение

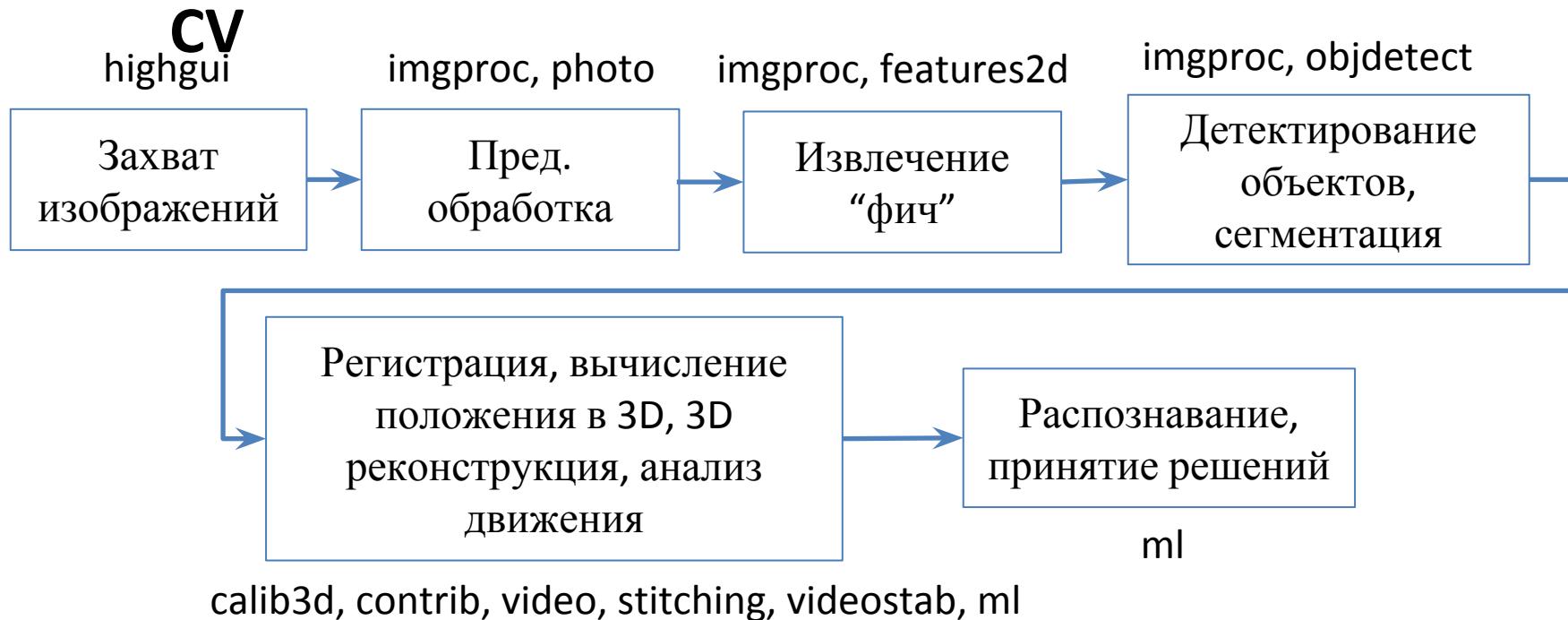


OpenCV в приложениях компьютерного зрения

OpenCV – базовая, в целом низкоуровневая библиотека.

Мы создаем строительные блоки, кирпичики для приложений.
Сами приложения предстоит построить пользователям

Общая схема типичного приложения



Некоторые из основных задач компьютерного зрения

1. Сопоставить 2+ изображений или видеопоследовательностей.
2. Найти заданный объект или объекты заданного класса на изображении
3. Найти похожее изображение в базе
4. Определить 3D позу объекта и его частей, 3D структуру всей сцены
5. “Улучшить” изображение
6. Проанализировать движение объектов/частей объектов на изображении, движение камеры
7. Выделить на изображении/видео отдельные элементы, проанализировать - что изображено, что происходит.

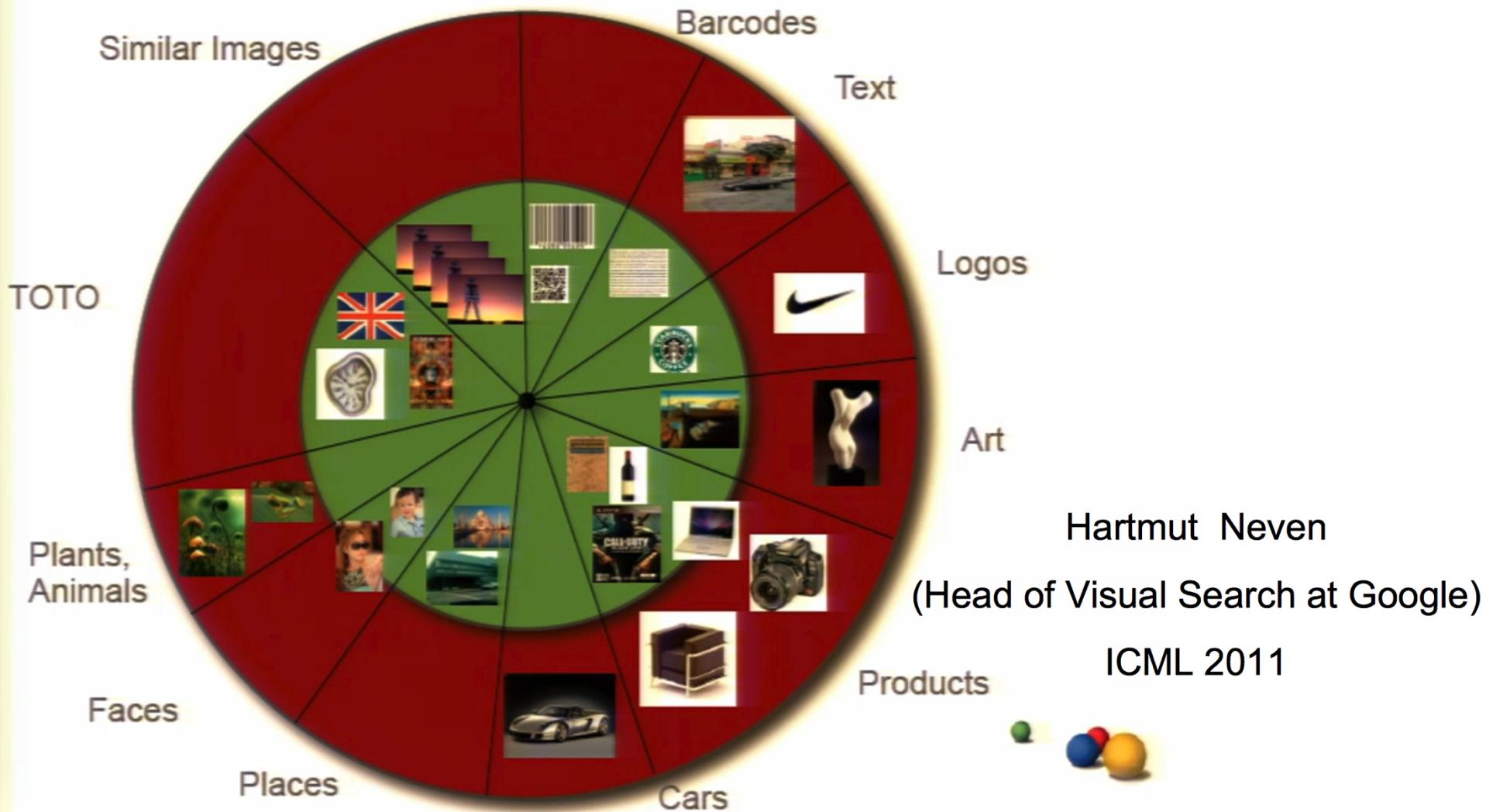
Каждая из задач имеет десятки вариантов, в зависимости от внешних условий, ограничений, требований по точности и скорости и т.д. Задачи могут комбинироваться, вкладываться друг в друга. Задачи могут решаться специальным железом.

См. также http://www.frontiersincomputervision.com/slides/FCV_Taxo_Zisserman.pdf

<http://szeliski.org/Book/>

Задача детектирования/распознавания

Recognition disciplines that work and do not work

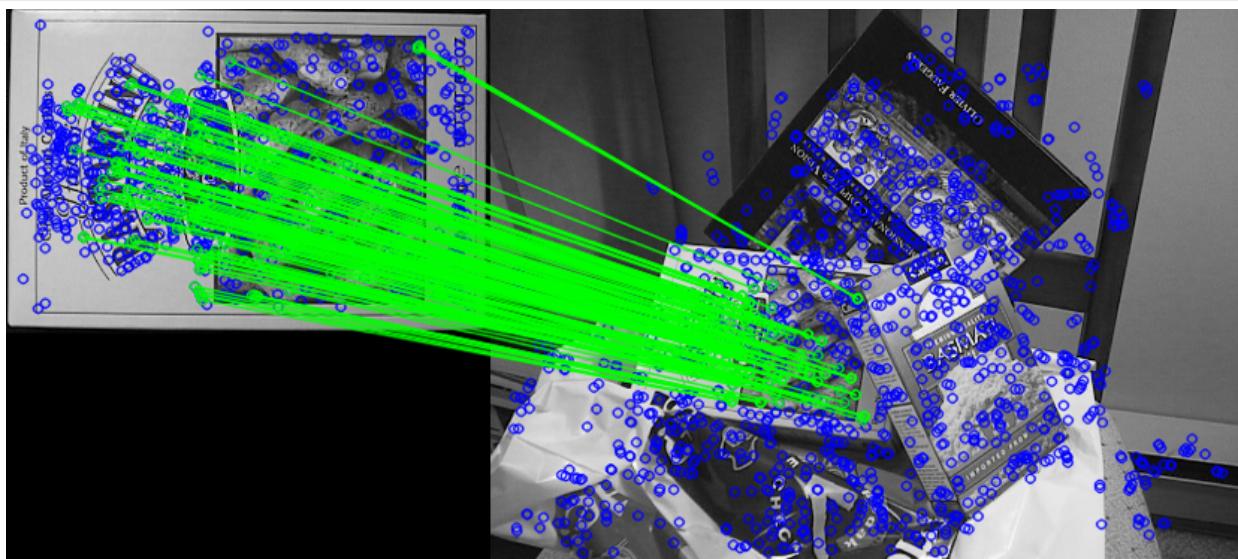


Примеры решения основных задач с использованием OpenCV



Поиск плоского объекта: features2d+ calib3d

```
Mat img[2], desc[2]; vector<KeyPoint> kpt[2];
Ptr<AKAZE> f2d = AKAZE::create();
for( int k = 0; k < 2; k++ ) {
    img[k] = imread(format("image%d.png", k+1), 0);
    f2d->detectAndCompute( img[k], Mat(), kpt[k], desc[k], false );
}
vector< DMatch > matches;
BFMatcher(NORM_HAMMING,true).match( desc[0], desc[1], matches );
vector<Point2f> points[2];
for( size_t i = 0; i < matches.size(); i++ ) {
    points[0].push_back(kpt[matches[i].queryIdx].pt);
    points[1].push_back(kpt[matches[i].trainIdx].pt);
}
Mat H, inliers;
H = findHomography( points[0], points[1], RHO, 1.0, inliers );
```



Детектирование текста

Модуль opencv_contrib/text. Алгоритм Ньюмана-Матаса + Tesseract.

```
// form channels
Mat image = imread(...), gray, group_img; vector<Mat> channels;
cvtColor(image, gray, COLOR_BGR2GRAY);
channels.push_back(gray); channels.push_back(255-gray);
// create and apply ER filters, or you can also use MSER
Ptr<ERFilter> er_filter1 = createERFilterNM1(loadClassifierNM1(
    "trained_classifierNM1.xml"), 8, 0.00015f, 0.13f, 0.2f, true, 0.1f);
Ptr<ERFilter> er_filter2 = createERFilterNM2(loadClassifierNM2(
    "trained_classifierNM2.xml"), 0.5); // create filters out of the loop!
vector<vector<ERStat>> regions(channels.size());
for( size_t c = 0; c < channels.size(); c++ ) {
    er_filter1->run(channels[c], regions[c]);
    er_filter2->run(channels[c], regions[c]);
}
// group regions
vector<vector<Vec2i>> nm_region_groups;
vector<Rect> nm_boxes;
erGrouping(image, channels, regions, nm_region_groups,
            nm_boxes, ERGROUPING_ORIENTATION_HORIZ);
// recognize the text
Ptr<OCRTesseract> ocr = OCRTesseract::create();
vector<string> words;
for (size_t i=0; i < nm_boxes.size(); i++) {
    gray(nm_boxes[i]).copyTo(group_img);
    copyMakeBorder(group_img, group_img, 15, 15, 15, 15, BORDER_CONSTANT, Scalar(0));
    string ocr_out; ocr->run(group_img, ocr_out); words.push_back(ocr_out);
}
```



HDR + вычитание фона используя opencv_contrib/bioinspired

Input video



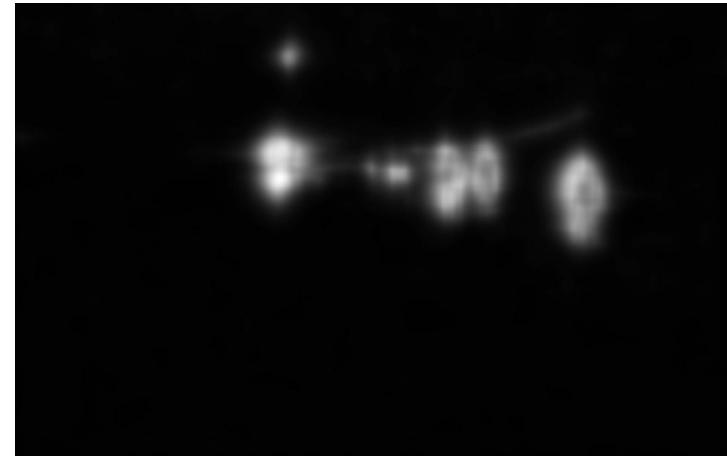
Parvo



```
#include "opencv2/bioinspired.hpp"

VideoCapture cap(...);
Ptr<bioinspired::Retina> retina;
Mat frame, parvo, magno;
for(;;) {
    cap >> frame;
    if(!retina)
        retina = bioinspired::createRetina(
            frame.size());
    retina->run(frame);
    retina->getParvo(parvo);
    retina->getMagno(magno);
}
```

Magno



Вопросы?