计算机视觉 第二期

Instructor: Julyedu 金

Image by
kirkh.deviantart.com

Syllabus of CV1

- 图像处理基础
- 图像的特征提取
- 机器视觉中的几何学:
  - 坐标变换与视觉测量
  - 3D计算机视觉
- 机器视觉中的机器学习方法与数据处理
  - 图像识别
  - 图像搜索

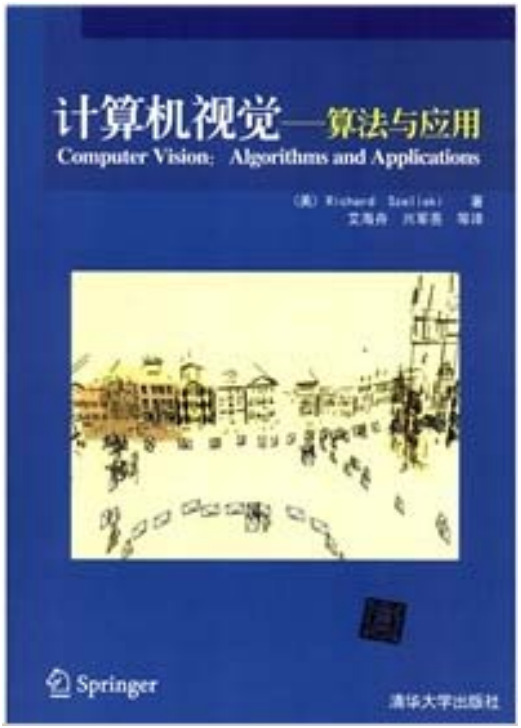
Syllabus of CV2

- 像素的角度理解图像
- 像素与像素之间的关系
- 机器视觉中的几何学：
  - 坐标变换与视觉测量
  - ~~3D计算机视觉~~
- 机器视觉中的机器学习方法与数据处理
  - 图像识别
  - 图像搜索
  - 目标检测、语义分割….

Text

- Computer Vision: Algorithms and Applications

Richard Szeliski



Lecture 1&2
Fundamental of Computer Vision

“工欲善其事必先利其器”

七月在线 金老师

2018年9月22日

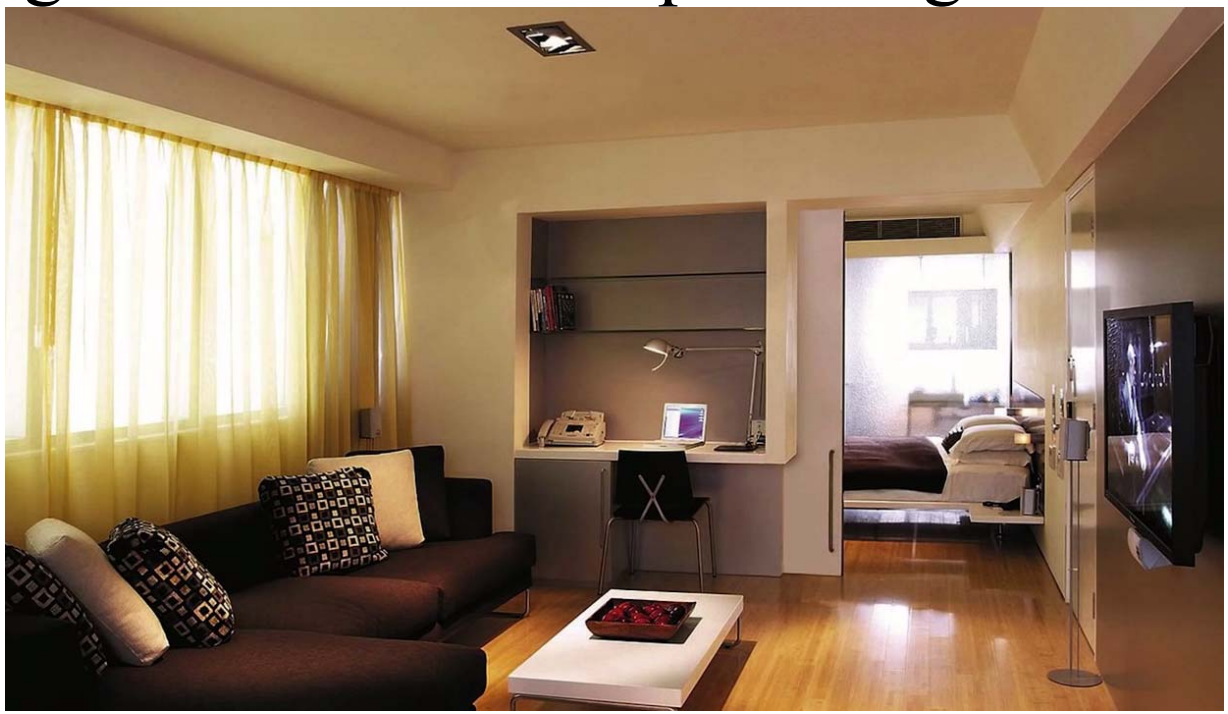
Outline

- 1. CV背景介绍
- 2. OpenCV完全解析
- 3. 图像的基本操作：遍历图像，ROI选取等
- 4. 机器学习在CV中的应用：Kmeans与KNN
- 5. 项目：图像拼接与测量

What is Computer Vision?

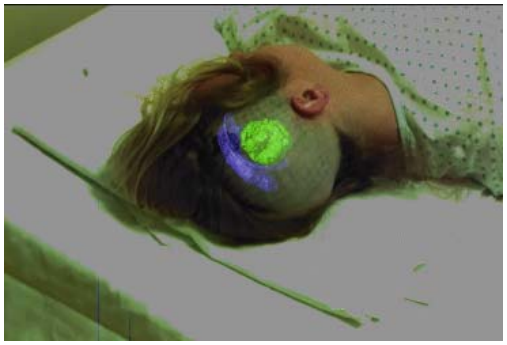
What is Computer Vision?

- Goal of computer vision is to write computer programs that can interpret images



Why computer vision matters

Safety

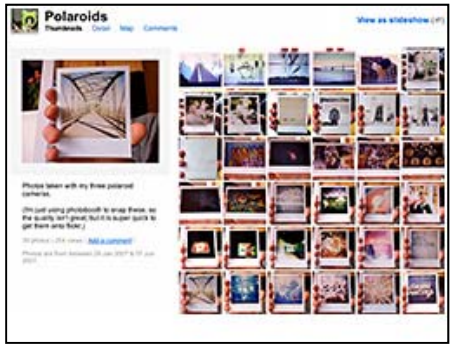


Health

Security

Comfort

Fun

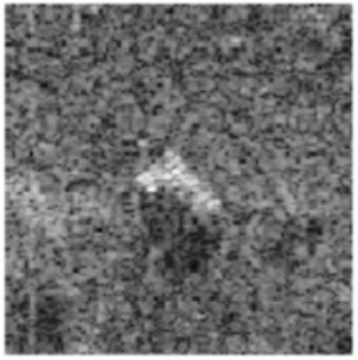


Access

Examples of application areas

- Target recognition
  - Find enemy vehicles (which are trying not to be found!)

FLIR image from sdvision.kaist.ac.kr/



SAR image from web.mit.edu/6.003/courseware/

- Human Interfaces
  - Detect faces and identify people
  - Recognize gestures, activities

Examples of application areas

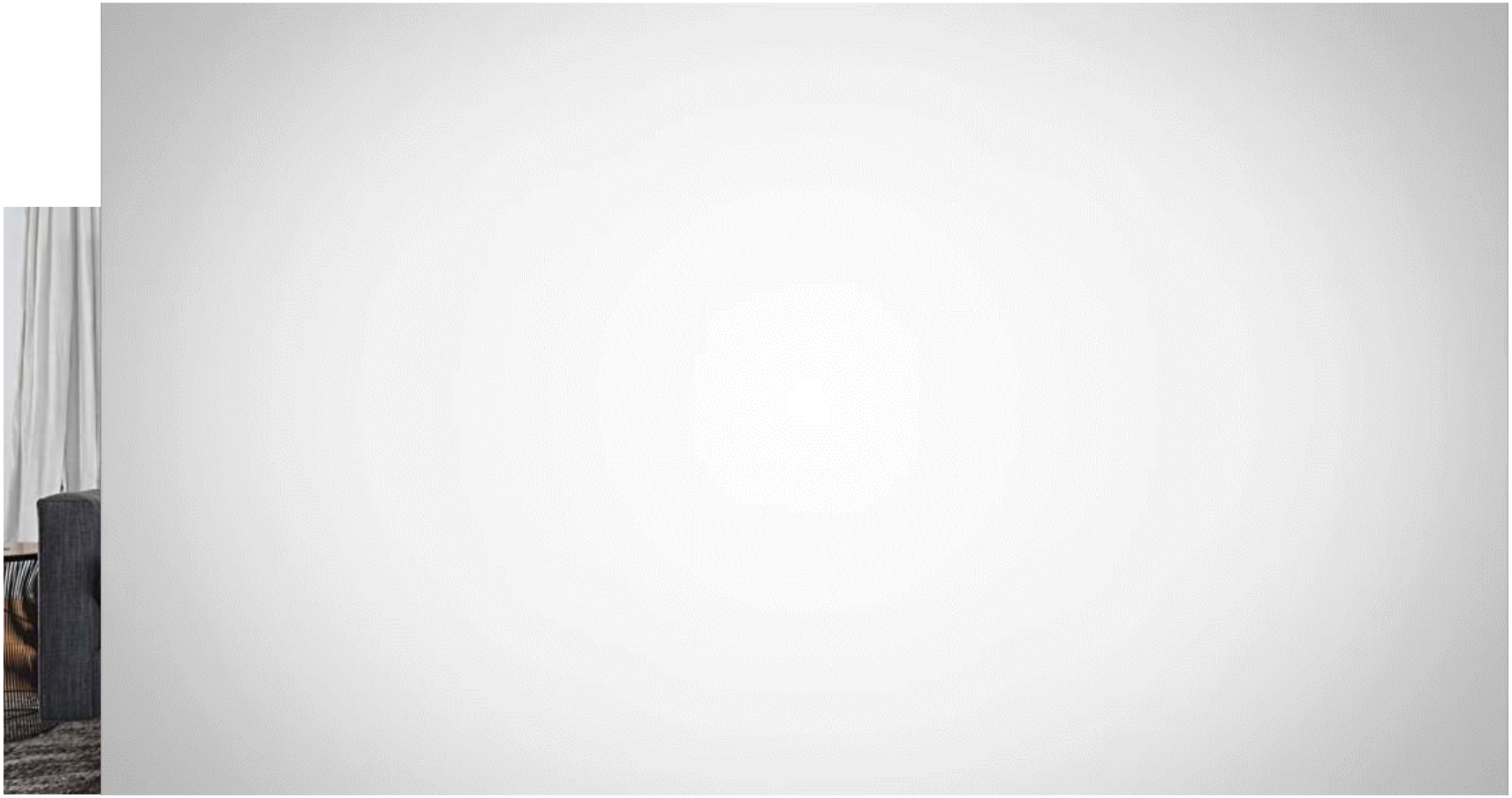
- Augmented reality
  - Augment the real world with virtual objects

- Robotics
  - Recognize objects
  - Estimate motion and position

http://www.robocup.org/

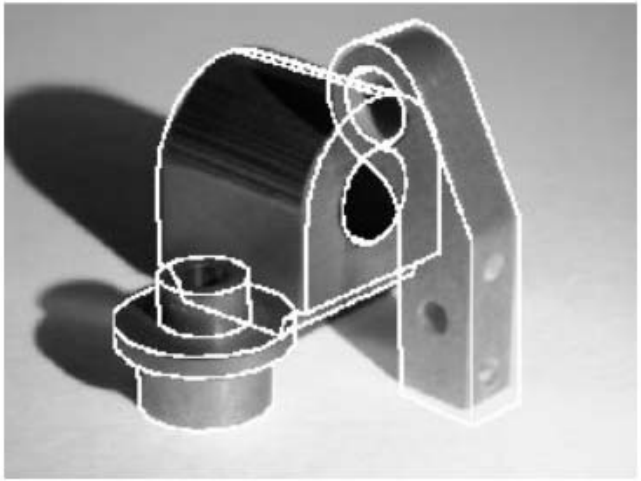
https://ubtrobot.com/

Examples of application areas



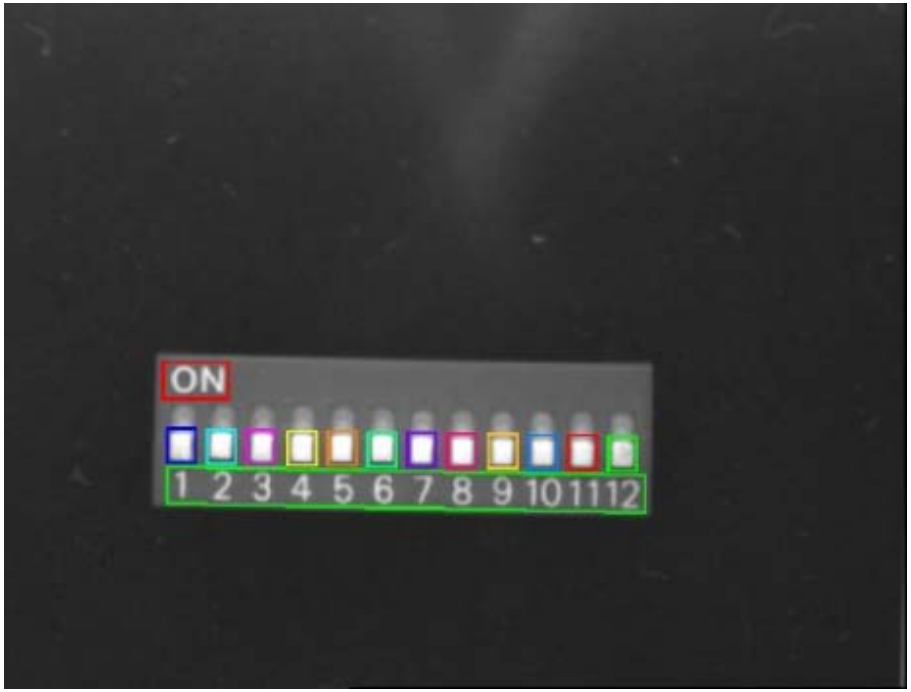
Examples of application areas

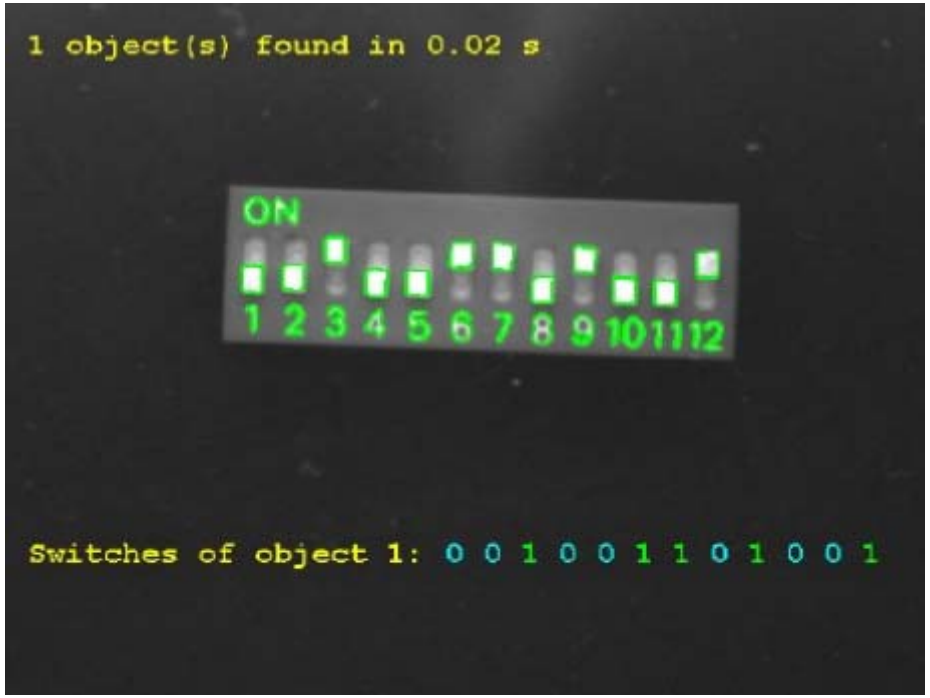
- Industrial inspection
  - Find known objects in the scene
  - Measure dimensions, verify features



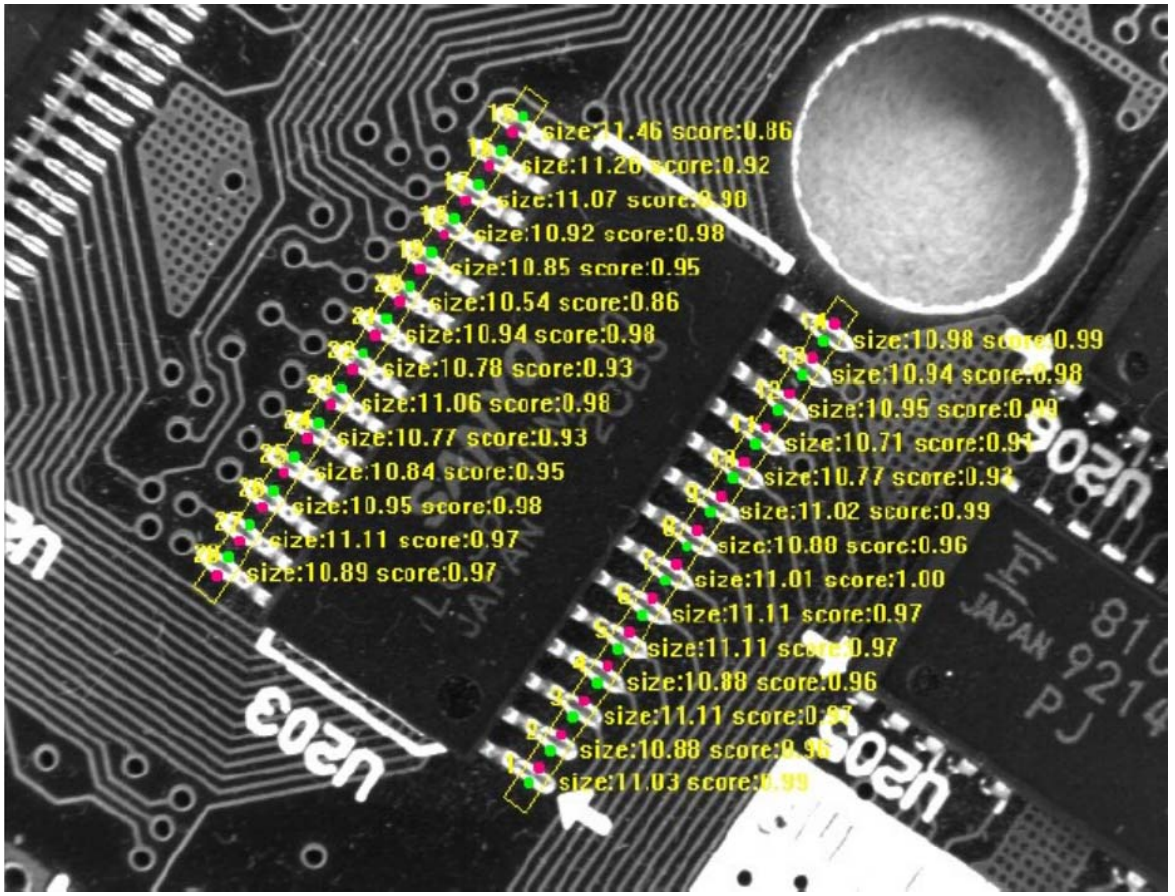
- Optical character recognition
  - Processing scanned text pages
  - Detect and identify characters

Examples of application areas

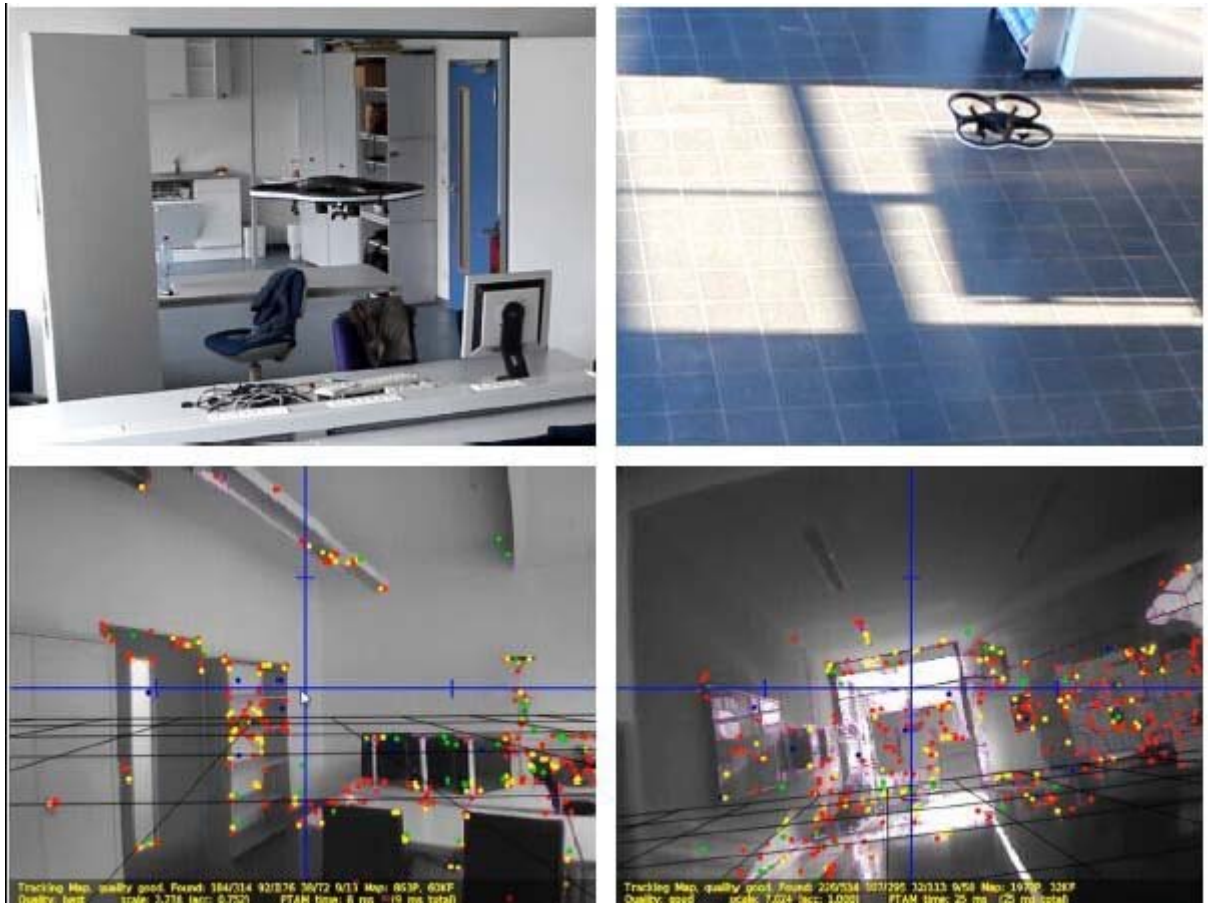




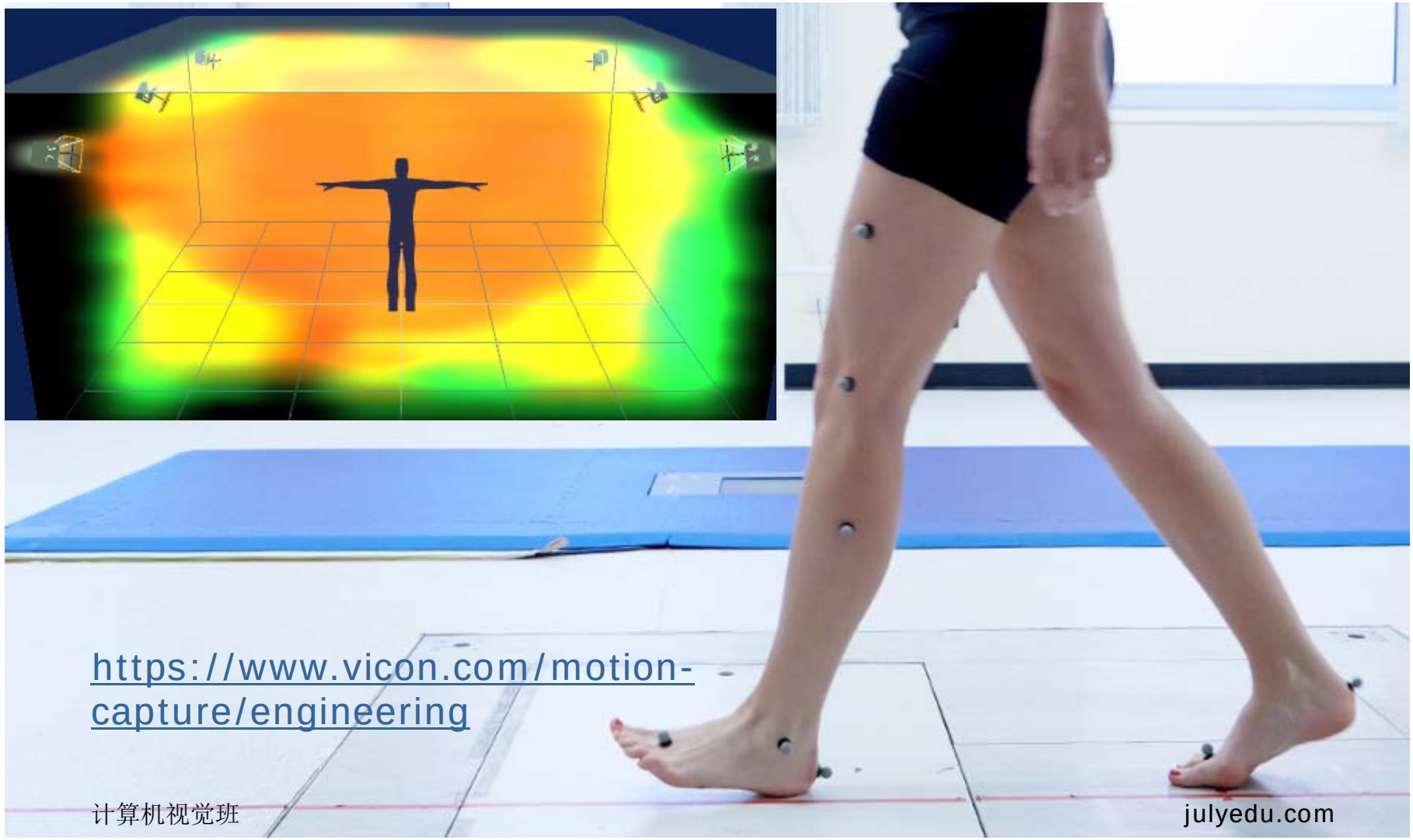
Examples of application areas



Examples of application areas



Vicon



https://www.vicon.com/motion-capture/engineering

MobileEye


Announcing the Market's 1st Turnkey Level 4/5 Automated Driving Solution
Mobileye and Delphi partner to produce the "Central Sensing Localization and Planning" (CSLP) platform to accelerate the time to market for a complete automated driving solution.
1 2 3 4 5 6
Pedestrian
Car
Crosswalk
Road
Car
MOBILEYE
Camera
LiDAR
Radar
RACam
Camera
LiDAR
Radar
Delphi's Multi-Domain Controller
LiDAR
Camera
Radar
LiDAR
Camera
Radar
DELPHI
Technology
2.5m
OEM
Fleet and Consumers

Open Libraries/projects

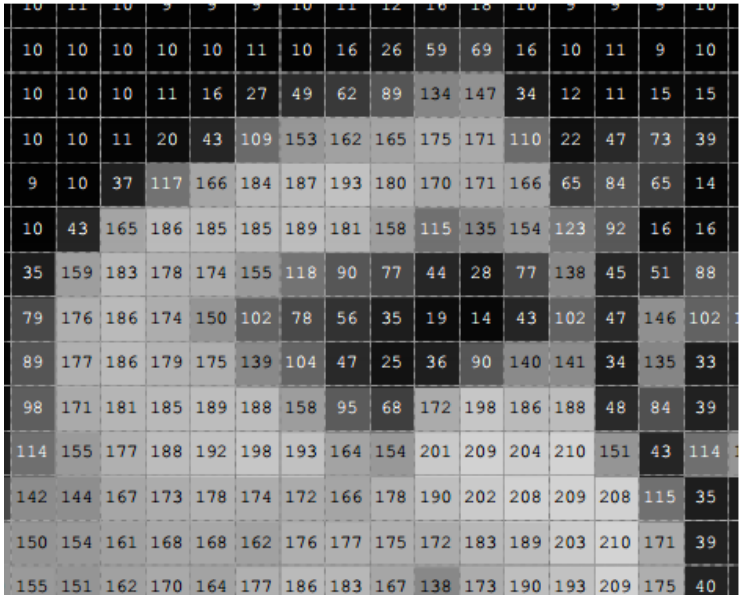
- OpenCV http://opencv.org/
  - C++, C, Python and Java interfaces
  - Windows, Linux, Mac OS, iOS and Android

CV之旅从这里开始

有图未必有真相

认识图像

- 二值图像
- 灰度图像
- 彩色图像（RGB，Lab，HSV）
- 多种颜色空间



digital images

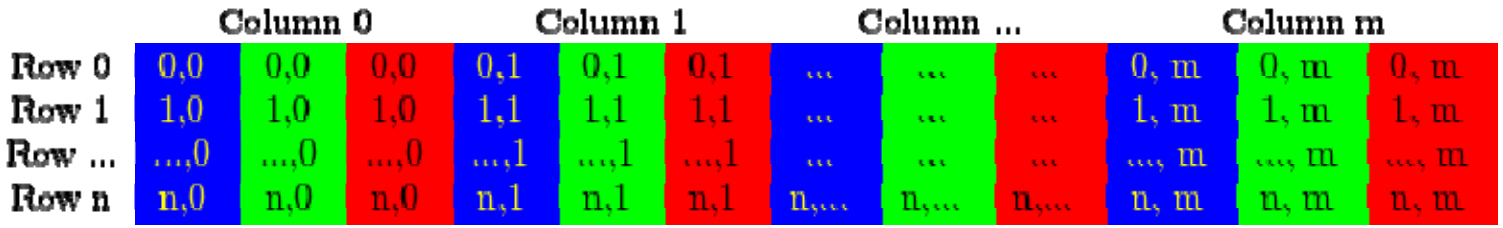
== arrays of numbers

== matrix

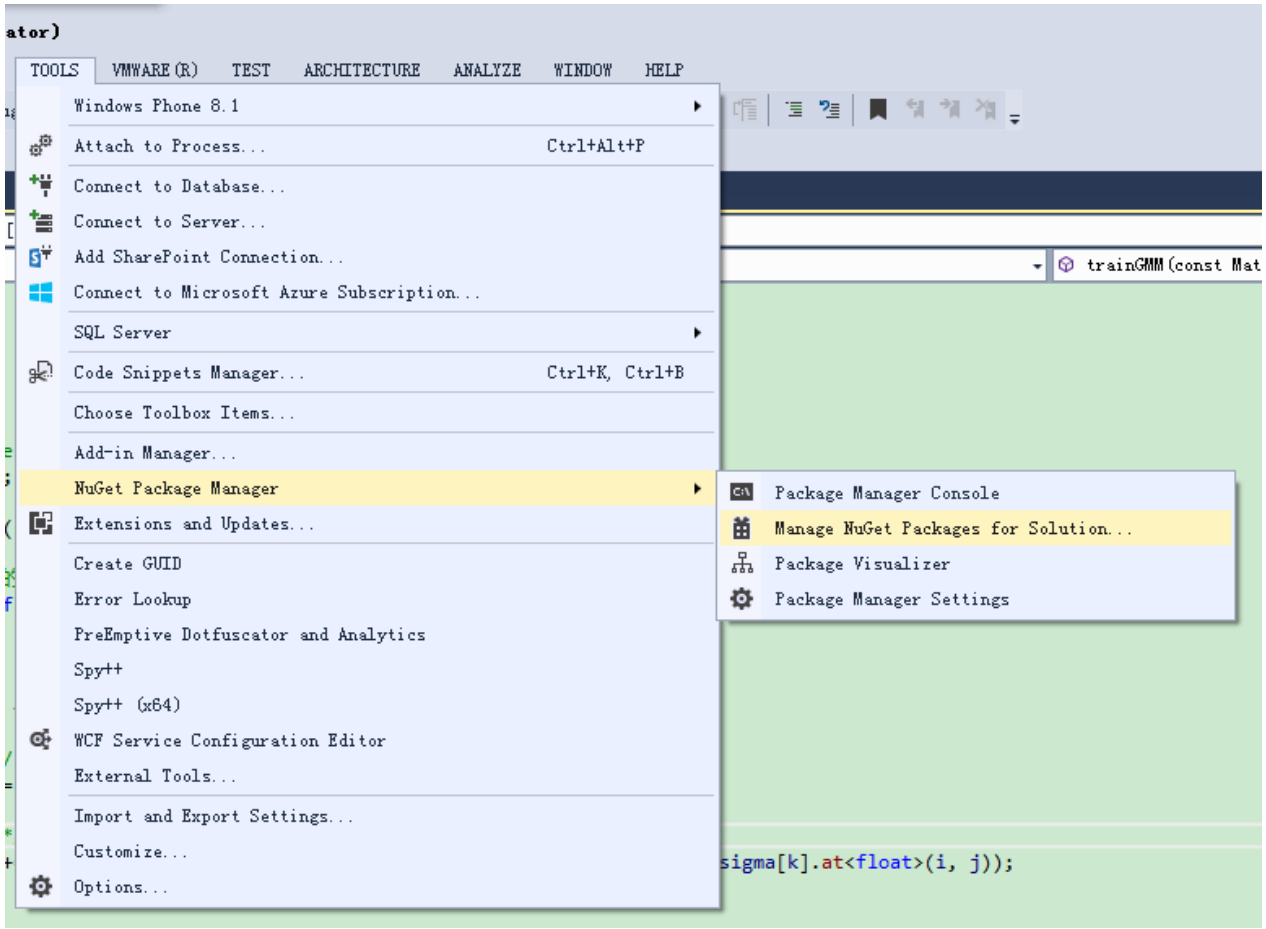
Data in memory

表 3-1 灰度图像的存储示意图

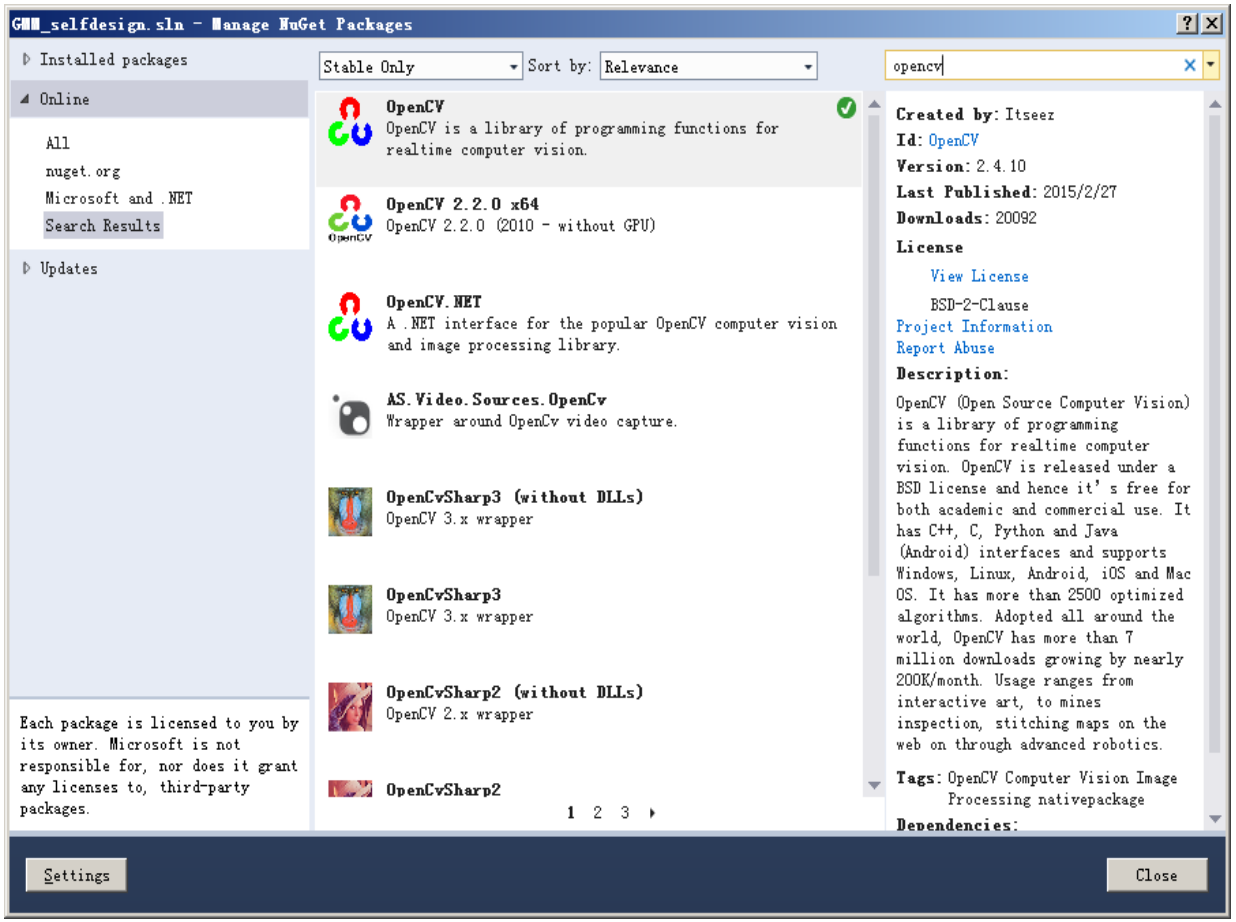
$I_{0\,0}$	$I_{0\,1}$	...	$I_{0\,N-1}$
$I_{1\,0}$	$I_{1\,1}$	...	$I_{1\,N-1}$
...	...	...	...
$I_{M-1\,0}$	$I_{M-1\,1}$	...	$I_{M-1\,N-1}$



免安装方法


ator)
TOOLS
VMWARE (R)
TEST
ARCHITECTURE
ANALYZE
WINDOW
HELP
Windows Phone 8.1
Attach to Process...
Ctrl+Alt+P
Connect to Database...
Connect to Server...
Add SharePoint Connection...
Connect to Microsoft Azure Subscription...
SQL Server
Code Snippets Manager...
Ctrl+K, Ctrl+B
Choose Toolbox Items...
Add-in Manager...
NuGet Package Manager
Extensions and Updates...
Create GUID
Error Lookup
PreEmptive Dotfuscator and Analytics
Spy++
Spy++ (x64)
WCF Service Configuration Editor
External Tools...
Import and Export Settings...
Customize...
Options...
Package Manager Console
Manage NuGet Packages for Solution...
Package Visualizer
Package Manager Settings
trainGMM(const Mat
sigma[k].at<float>(i, j));

免安装方法


GUI_selfdesign.sln - Manage NuGet Packages
Installed packages
Online
All
nuget.org
Microsoft and .NET
Search Results
Updates
Each package is licensed to you by its owner. Microsoft is not responsible for, nor does it grant any licenses to, third-party packages.
Stable Only
Sort by: Relevance
opencv
OpenCV
OpenCV is a library of programming functions for realtime computer vision.
OpenCV 2.2.0 x64
OpenCV 2.2.0 (2010 - without GPU)
OpenCV.NET
A .NET interface for the popular OpenCV computer vision and image processing library.
AS.Video.Sources.OpenCv
Wrapper around OpenCv video capture.
OpenCvSharp3 (without DLLs)
OpenCV 3.x wrapper
OpenCvSharp3
OpenCV 3.x wrapper
OpenCvSharp2 (without DLLs)
OpenCV 2.x wrapper
OpenCvSharp2
1 2 3
Created by: Itseez
Id: OpenCV
Version: 2.4.10
Last Published: 2015/2/27
Downloads: 20092
License
View License
BSD-2-Clause
Project Information
Report Abuse
Description:
OpenCV (Open Source Computer Vision) is a library of programming functions for realtime 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 (Android) interfaces and supports Windows, Linux, Android, iOS and Mac OS. It has more than 2500 optimized algorithms. Adopted all around the world, OpenCV has more than 7 million downloads growing by nearly 200K/month. Usage ranges from interactive art, to mines inspection, stitching maps on the web on through advanced robotics.
Tags: OpenCV Computer Vision Image Processing nativepackage
Dependencies:
Settings
Close

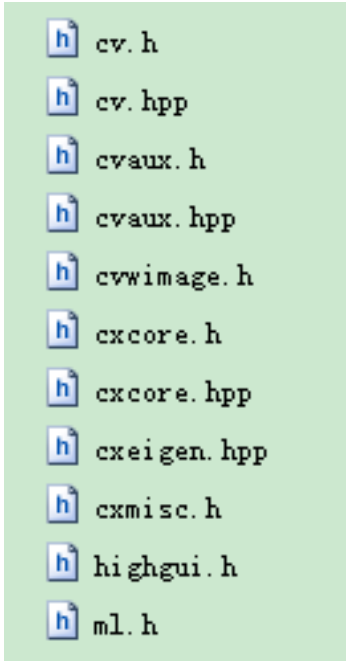
Opencv安装配置（opencv2.4+&3+）

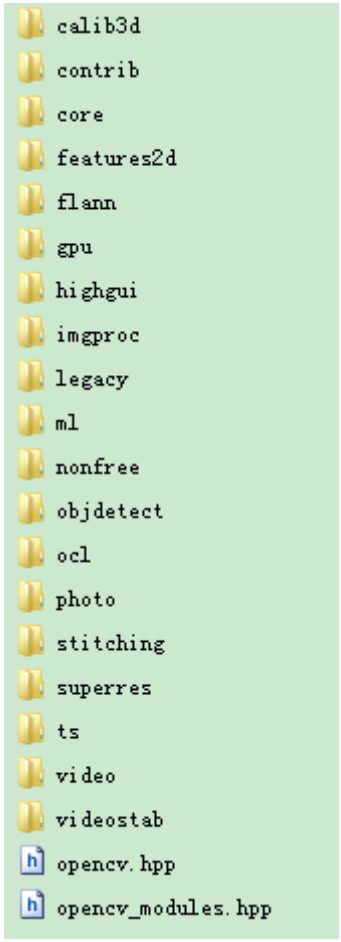
- 下载:
  - http://opencv.org/（最新发布）
  - https://sourceforge.net/projects/opencvlibrary/

  - https://github.com/opencv

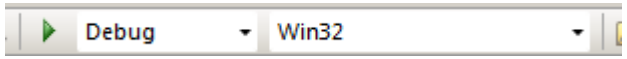
OpenCV文件


doc
include
java
python
share
x64
x86


cv.h
cv.hpp
cvaux.h
cvaux.hpp
cvwimage.h
cxcore.h
cxcore.hpp
cxeigen.hpp
cxmisc.h
highgui.h
ml.h


calib3d
contrib
core
features2d
flann
gpu
highgui
imgproc
legacy
ml
nonfree
objdetect
ocl
photo
stitching
superres
ts
video
videostab
opencv.hpp
opencv_modules.hpp

Opencv安装配置（opencv2.4&3.0）

- 环境变量 PATH
  - opencv¥build¥x86¥vc10¥bin 

  - opencv¥build¥x64¥vc10¥bin（64位编译）

```
%OPENCVROOT%\cmakeOpenCV\X64VS12CV3\install\x64\vc12\bin
%OPENCVROOT%\cmakeOpenCV\X86VS12CV300\install\x86\vc12\bin
%OPENCVROOT%\opencv2.4.10\build\x64\vc10\bin
%OPENCVROOT%\opencv2.4.10\build\x86\vc10\bin
```

```
OpenCV\opencv2411\build\x86\vc12\bin
OpenCV\opencv2411\build\x86\vc10\bin
```

Opencv安装配置（opencv2.4&3.0）

- VS工程“属性管理器”中添加：
  - 【通用属性】->【VC++目录】->【包含目录】
    - build¥include
    - build¥include¥opencv
    - build¥include¥opencv2
  - 【通用属性】->【VC++目录】->【库目录】
    - opencv¥build¥x86¥vc12¥lib
  - 【通用属性】->【链接器】->【输入】->【附加的依赖项】
    - opencv_nonfree2411d.lib opencv_nonfree2411.lib….
    - opencv_ts300d.lib opencv_world300d.lib

Image Watch

- 1、Image Watch 的下载链接。
- 2、OpenCV关于Image Watch的介绍页面链接。
- 3、OpenCV2.4 在线文档关于Image Watch的介绍文档。
- 4、更详细的信息参见Image Watch的官方网站。

view → other windows → Image watch

程序模板

```
#include <iostream>
#include "opencv2/opencv.hpp"
using namespace std;
using namespace cv;
```

学习OpenCV

- Model:
  - Core →opencv_core.lib
  - Imgproc → opencv_imgproc.lib
  - Highgui →opencv_highgui.lib

```
class CV_EXPORTS Mat
{
public:
    //一系列函数
    ...
    /* flag 参数中包含许多关于矩阵的信息，如：
        -Mat 的标识
        -数据是否连续
        -深度
        -通道数目
     */
    int flags;
    //矩阵的维数，取值应该大于或等于 2
    int dims;
    //矩阵的行数和列数，如果矩阵超过 2 维，这两个变量的值都为-1
    int rows, cols;
    //指向数据的指针
    uchar* data;

    //指向引用计数的指针
    //如果数据是由用户分配的，则为 NULL
    int* refcount;

  //其他成员变量和成员函数
  ...
};
```

```
Mat M(3,2, CV_8UC3, Scalar(0,0,255));
cout << "M = " << endl << " " << M << endl;
```

```
M =
  [0,  0,  255,  0,  0,  255;
   0,  0,  255,  0,  0,  255;
   0,  0,  255,  0,  0,  255]
```

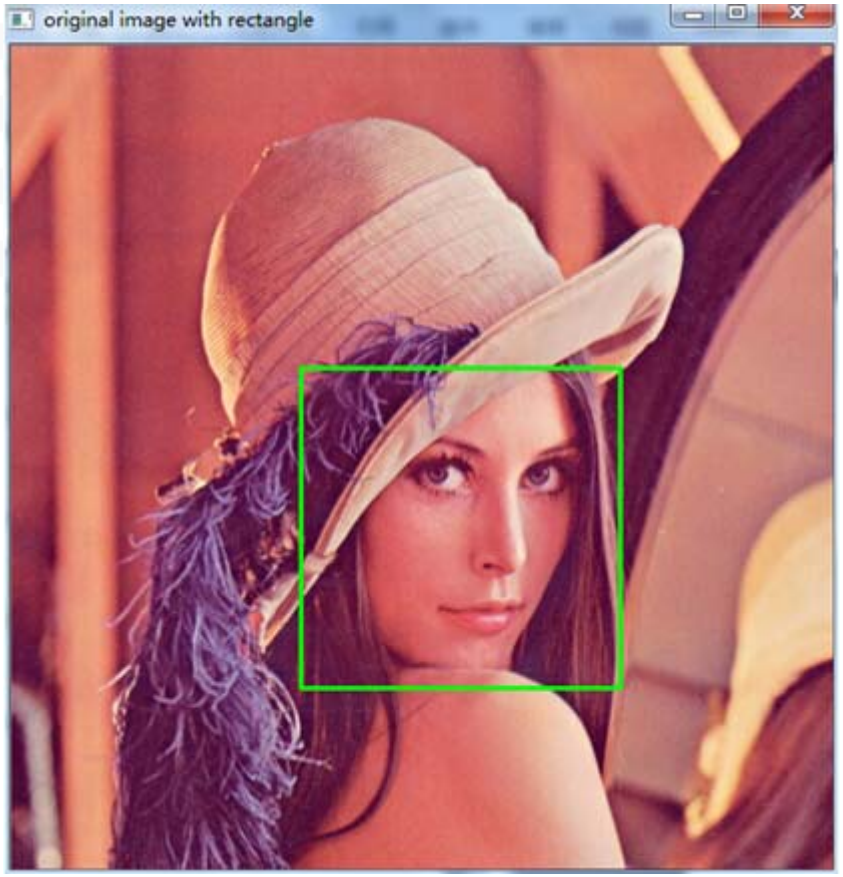
常用构造函数

- Mat::Mat()
- Mat::Mat(int rows, int cols, int type)
- Mat::Mat(Size size, int type)
- Mat::Mat(int rows, int cols, int type, const Scalar& s)
- Mat::Mat(Size size, int type, const Scalar& s)
- Mat::Mat(const Mat& m)
- Mat::Mat(int rows, int cols, int type, void* data, size_t step=AUTO_STEP)
- Mat::Mat(Size size, int type, void* data, size_t step=AUTO_STEP)
- Mat::Mat(const Mat& m, const Range& rowRange, const Range& colRange)
- Mat::Mat(const Mat& m, const Rect& roi)

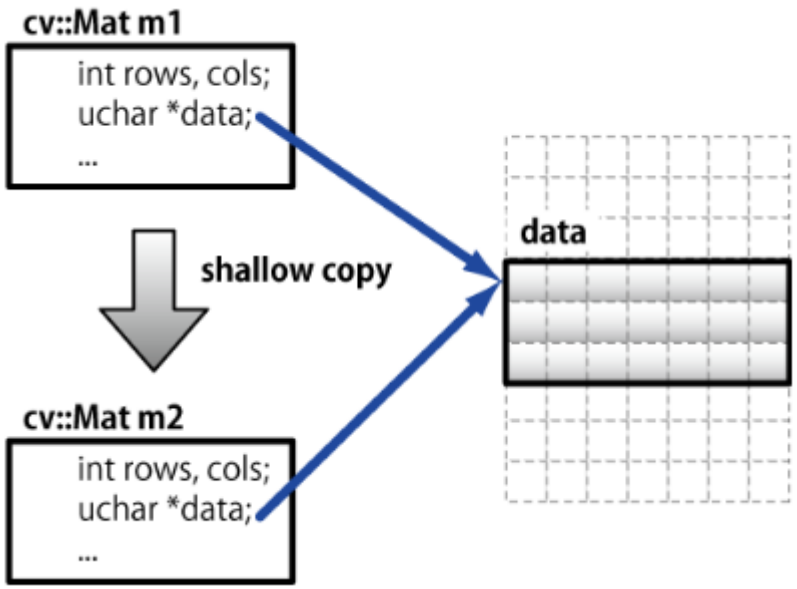
```
Mat M(2,2, CV_8UC3);//构造函数创建图像
M.create(3,2, CV_8UC2);//释放内存重新创建图像
```

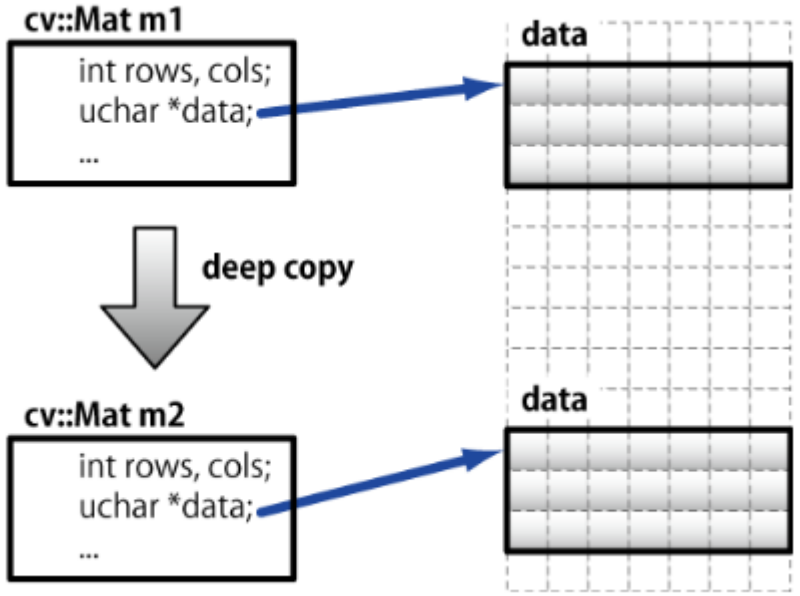
ROI

```
// Rect
    cv::Mat pImg = imread("Lena.jpg",1);
    cv::Rect  rect(180,200,200,200);//(x,y)=(180,200),w=200,height=200
    cv::Mat  roi = cv::Mat(pImg, rect);
    cv::Mat  pImgRect = pImg.clone();
    cv::rectangle(pImgRect,rect,cv::Scalar(0,255,0),2);
    cv::imshow("original image with rectangle",pImgRect);
    cv::imshow("roi",roi);
    cv::waitKey();
```



Mat的赋值和拷贝问题





Similar to Matlab

- Mat Z = Mat::zeros(2,3, CV_8UC1);
  cout << "Z = " << endl << " " << Z << endl;
  Mat O = Mat::ones(2, 3, CV_32F);
  cout << "O = " << endl << " " << O << endl;
  Mat E = Mat::eye(2, 3, CV_64F);
  cout << "E = " << endl << " " << E << endl;

- E = [?]

像素值的读写 1

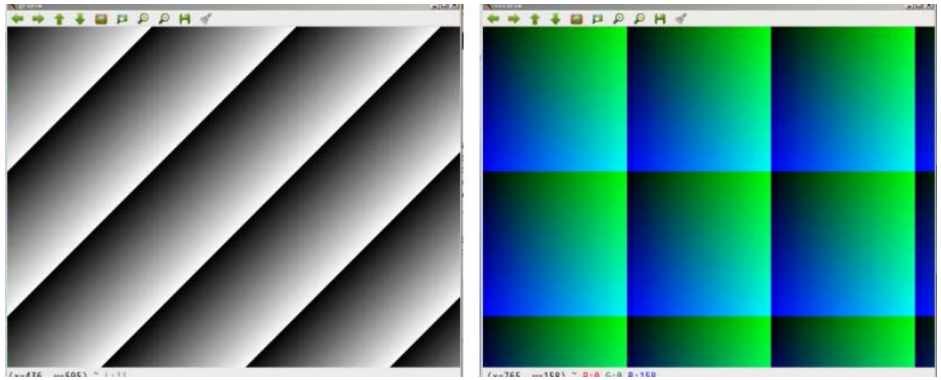
- uchar value = grayim.at<uchar>(i,j);
  for(int i = 0; i < grayim.rows; ++i)
  for(int j = 0; j < grayim.cols; ++j)
  grayim.at<uchar>(i,j) = (i+j)%255;

```
for( int i = 0; i < colorim.rows; ++i)
    for( int j = 0; j < colorim.cols; ++j )
    {
    Vec3b pixel;
    pixel[0] = i%255; //Blue
    pixel[1] = j%255; //Green
    pixel[2] = 0; //Red
    colorim.at<Vec3b>(i,j) = pixel;
    }
```

像素值的读写 1

- uchar value = grayim.at<uchar>(i,j);
  for(int i = 0; i < grayim.rows; ++i)
  for(int j = 0; j < grayim.cols; ++j)
  grayim.at<uchar>(i,j) = (i+j)%255;

```
for( int i = 0; i < colorim.rows; ++i)
    for( int j = 0; j < colorim.cols; ++j )
    {
    Vec3b pixel;
    pixel[0] = i%255; //Blue
    pixel[1] = j%255; //Green
    pixel[2] = 0; //Red
    colorim.at<Vec3b>(i,j) = pixel;
    }
```



像素值的读写 2

- cv::Mat Iterator_<uchar> grayit, grayend;
  for(grayit = grayim.begin<uchar>(), grayend = grayim.end<uchar>(); grayit != grayend; ++grayit)

  *grayit = rand()%255;

```
MatIterator_<Vec3b> colorit, colorend;
for( colorit = colorim.begin<Vec3b>(), colorend = colorim.end<Vec3b>(); colorit != colorend; ++colorit)
{
    (*colorit)[0] = rand()%255; //Blue
    (*colorit)[1] = rand()%255; //Green
    (*colorit)[2] = rand()%255; //Red
}
```

像素值的读写 3

```
for( int i = 0; i < grayim.rows; ++i)
{
    //获取第 i 行首像素指针
    uchar * p = grayim.ptr<uchar>(i);
    //对第 i 行的每个像素(byte)操作
    for( int j = 0; j < grayim.cols; ++j )
    p[j] = (i+j)%255;

}
```

像素值的读写 2

- cv::Mat Iterator_<uchar> grayit, grayend;
  for(grayit = grayim.begin<uchar>(), grayend = grayim.end<uchar>(); grayit != grayend; ++grayit)

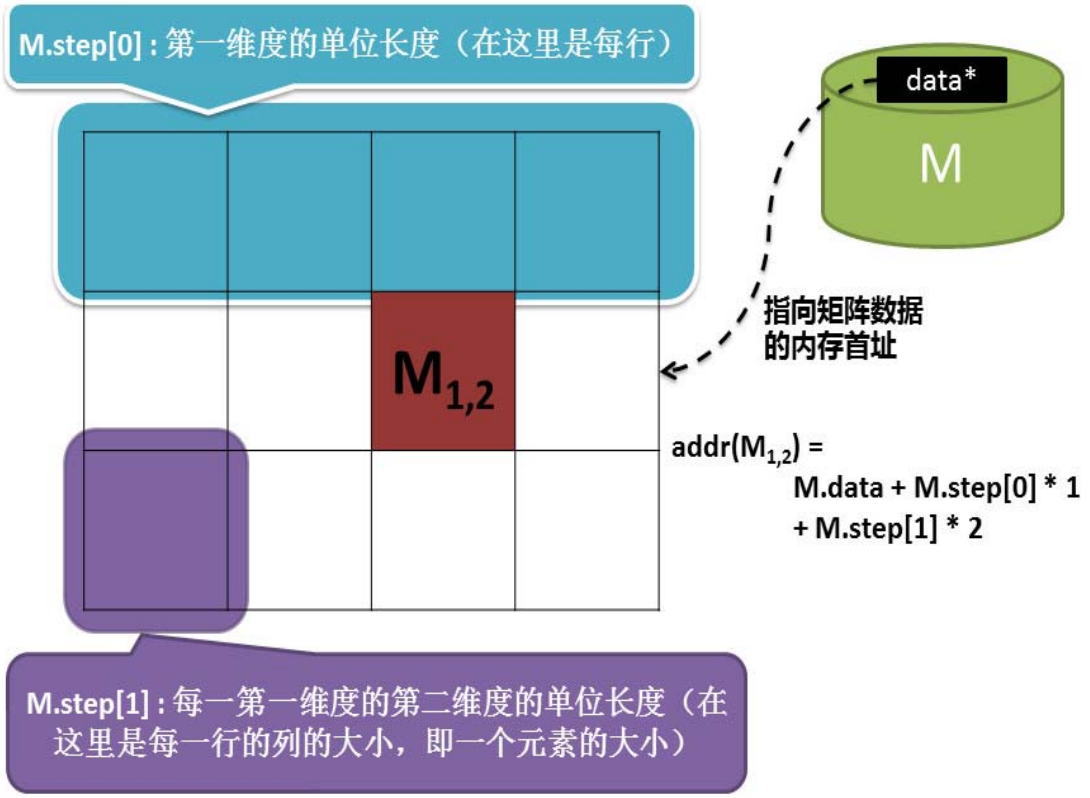
  *grayit = rand()%255;

```
MatIterator_<Vec3b> colorit, colorend;
for( colorit = colorim.begin<Vec3b>(), colorend = colorim.end<Vec3b>(); colorit != colorend; ++colorit)
{
    (*colorit)[0] = rand()%255; //Blue
    (*colorit)[1] = rand()%255; //Green
    (*colorit)[2] = rand()%255; //Red
}
```

像素值的读写 3

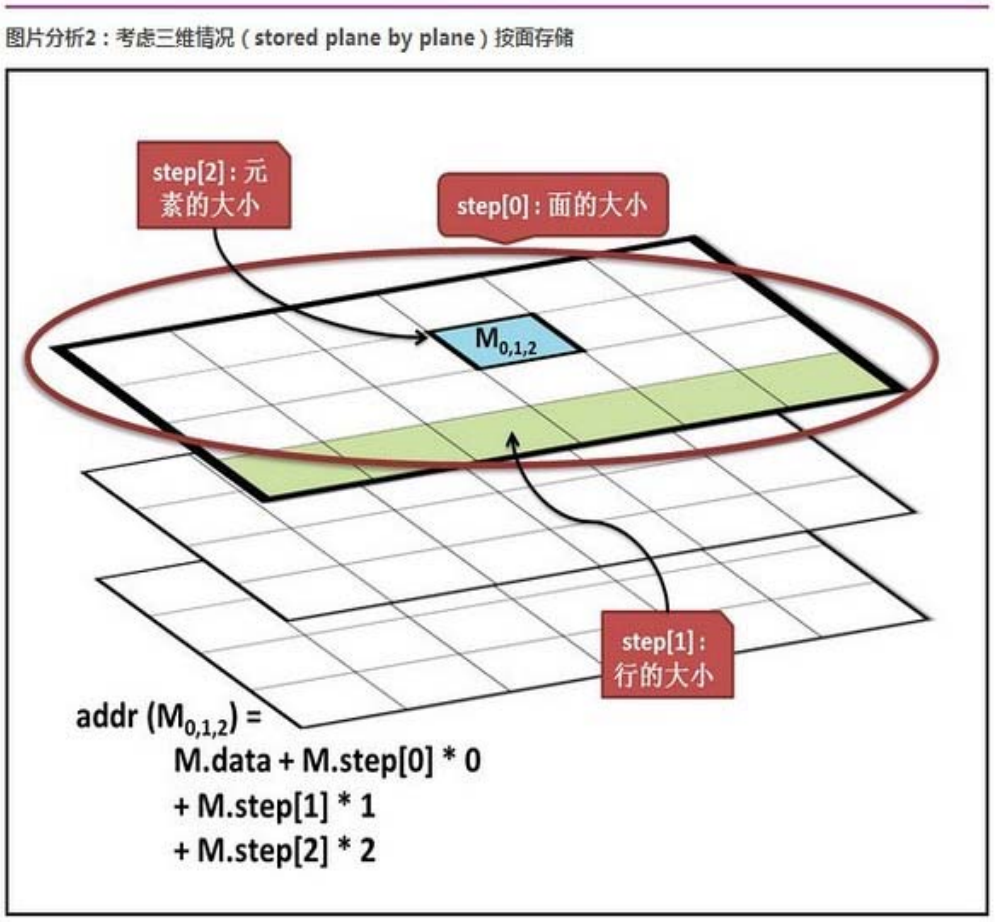
```
int channels = I.channels();
int nRows = I.rows ;
int nCols = I.cols* channels;
if (I.isContinuous())
{
    nCols *= nRows;
    nRows = 1;
}
int i,j;
uchar* p;
for( i = 0; i < nRows; ++i)
{
    p = I.ptr<uchar>(i);
    for ( j = 0; j < nCols; ++j)
    {
        p[j] = table[p[j]];
    }
}
```

像素值的读写 4



addr(Mi0,i1,…im-1) = M.data + M.step[0] * i0 + M.step[1] * i1 + … + M.step[m-1] * im-1 (其中 m = M.dims M的维度)

像素值的读写 4


图片分析2：考虑三维情况（stored plane by plane）按面存储
step[2]：元素的大小
step[0]：面的大小
$M_{0,1,2}$
step[1]：行的大小
addr ($M_{0,1,2}$) =
M.data + M.step[0] * 0
+ M.step[1] * 1
+ M.step[2] * 2

```
int main()
{
    //新建一个uchar类型的单通道矩阵（grayscale image 灰度图）
    Mat m(400, 400, CV_8U, Scalar(0));
    for (int col = 0; col < 400; col++)
    {
        for (int row = 195; row < 205; row++)
        {
            cout << (int)(*(m.data + m.step[0] * row + m.step[1] * col)) << "  ==> ";
            //获取第[row,col]个像素点的地址并用 * 符号解析
            *(m.data + m.step[0] * row + m.step[1] * col) = 255;
            cout << (int)(*(m.data + m.step[0] * row + m.step[1] * col)) << endl;
        }
    }
    imshow("canvas", m);
    cvWaitKey();
    return 0;
}
```

```
int main()
{
	//读入一个彩色图像
	Mat m = imread("../data/HappyFish.jpg");
	int *p_address;
	Vec3i color;
	for (int col = 20; col < 40; col++)
	{
		for (int row = 2; row < 20; row++)
		{
			color[0] = (int)(*(m.data + m.step[0] * row + m.step[1] * col));
			color[1] = (int)(*(m.data + m.step[0] * row + m.step[1] * col + m.elemSize1()));
			color[2] = (int)(*(m.data + m.step[0] * row + m.step[1] * col + m.elemSize1()*2));
			//获取第[row,col]个像素点的地址并用 * 符号解析
			cout << color[0]<<","<<color[1]<<","<<color[2] << "  ==> ";
			color[0] = 255;
			color[1] = 0;
			color[2] = 0;
			*(m.data + m.step[0] * row + m.step[1] * col) = color[0];
			*(m.data + m.step[0] * row + m.step[1] * col + m.elemSize1()) = color[1];
			*(m.data + m.step[0] * row + m.step[1] * col + m.elemSize1()*2) = color[2];
			cout << (int)*(m.data + m.step[0] * row + m.step[1] * col) << (int)*(m.data +
m.step[0] * row + m.step[1] * col + 1) << (int)*(m.data + m.step[0] * row + m.step[1] * col + 2) <<
endl;
		}
	}
	imshow("canvas", m); cvWaitKey();
	return 0;
}
```

像素值的读写 5 Mat_类

```
Mat M(600, 800, CV_8UC1);

for( int i = 0; i < M.rows; ++i)
{

    uchar * p = M.ptr<uchar>(i);
    for( int j = 0; j < M.cols; ++j )
    {

            double d1 = (double) ((i+j)%255);
            M.at<uchar>(i,j) = d1;
            double d2 = M.at<double>(i,j);
     }

}
```

Mat_类

```
Mat_<uchar> M1 = (Mat_<uchar>&)M;
for( int i = 0; i < M1.rows; ++i)
{

    uchar * p = M1.ptr(i);
    for( int j = 0; j < M1.cols; ++j )
    {

        double d1 = (double) ((i+j)%255);
        M1(i,j) = d1;
        double d2 = M1(i,j);

    }

}
```

像素值的读写 5

```
m = imread("../data/HappyFish.jpg");
Mat_<Vec3b> m2 = m;

// for 循环画一个红色的实心圆
for (int y = 21; y < 42; y++)
{
	for (int x = 2; x < 21; x++)
	{
		if (pow(double(x - 11), 2) + pow(double(y - 31), 2) - 64.0 < 0.00000000001)
		{
			// Mat_ 模板类实现了对()的重载，可以定位到一个像素
			m2(x, y) = Vec3b(0, 0, 255);
		}
	}
}

imshow("CircleImage", m2);
```

像素值的读写 6

- **int** divideWith=10;
- uchar table[256];
- **for** (**int** i = 0; i < 256; ++i)
- table[i] = divideWith* (i/divideWith);

- Mat lookUpTable(1, 256, CV_8U);
- uchar* p = lookUpTable.data;
- for(int i = 0; i < 256; ++i)
- p[i] = table[i];
- LUT(I, lookUpTable, Out);

Mat 与 IplImage 和 CvMat 的转换

- void mycvOldFunc(IplImage * p, ...);

- Mat img(Size(320, 240), CV_8UC3);

```
...
IplImage iplimg = img;
//CvMat cvimg = img;
mycvOldFunc( & iplimg, ...);
```

IplImage 和 CvMat 格式转为 Mat

Mat::Mat(const CvMat* m, bool copyData=false)
Mat::Mat(const IplImage* img, bool copyData=false)

```
IplImage * iplimg = cvLoadImage("lena.jpg");
Mat im(iplimg, true);
```

数据获取与存储

- imread()

Mat imread(const string& filename, int flags=1)

Windows 位图文件 - BMP, DIB；

JPEG 文件 - JPEG, JPG, JPE；

便携式网络图片 - PNG；

便携式图像格式 - PBM，PGM，PPM；

Sun rasters - SR，RAS；

TIFF 文件 - TIFF，TIF;

OpenEXR HDR 图片 - EXR;

JPEG 2000 图片- jp2。

imwrite()

bool imwrite(const string& filename, InputArray image, const vector<int>& params=vector<int>())

Video读写类

- VideoCapture cap(0);
- VideoCapture cap("video.short.raw.avi");

```
if(!cap.isOpened())
{
  cerr << "Can not open a camera or file." << endl;
  return -1;
}
Mat edges;
namedWindow("edges",1);
for(;;)
{
  Mat frame;
  cap >> frame;
  if(frame.empty())
     break;
  ....
 if(waitKey(30) >= 0)
  break;
}
```

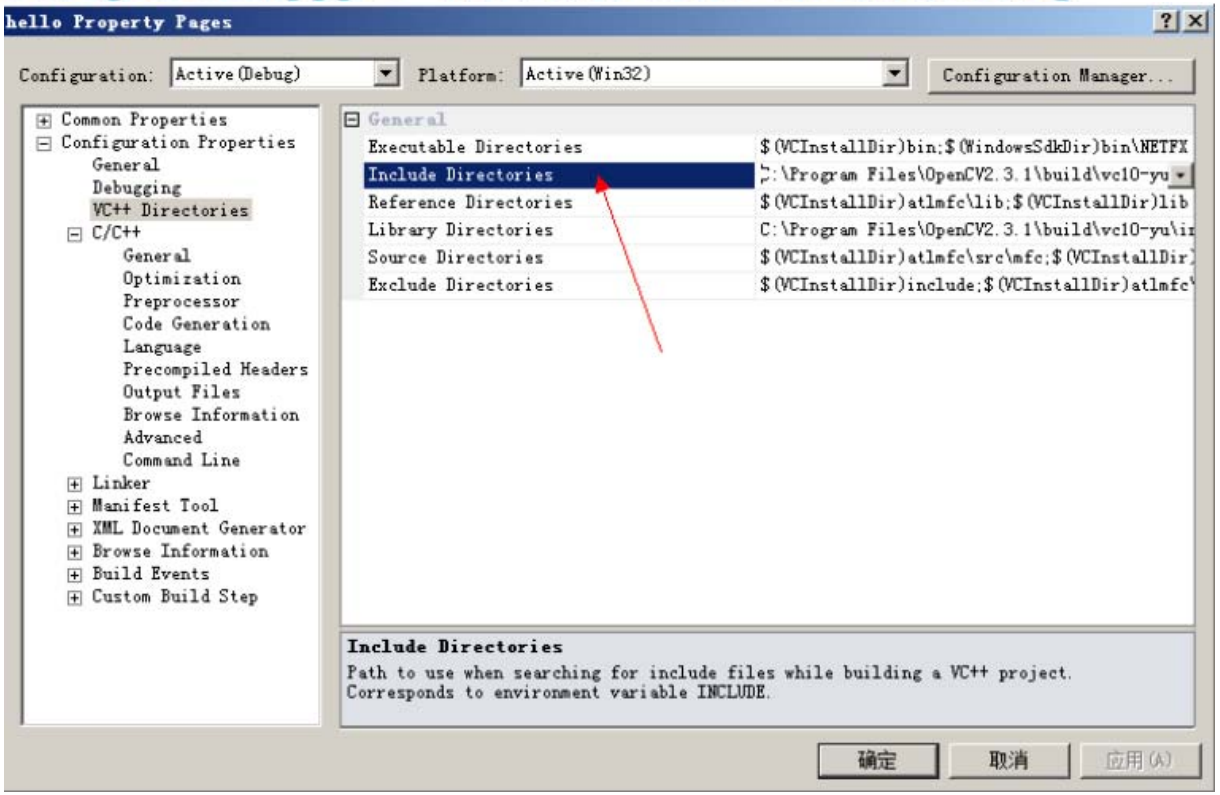
http://www.fourcc.org/codecs.php

Video读写类

```
Size s(320, 240);
VideoWriter writer = VideoWriter(“myvideo.avi”,
CV_FOURCC(‘M’,‘J’,‘P’,‘G’), 25, s);
if(!writer.isOpened())
{
   cerr << “Can not create video file.¥n” << endl;
   return -1;
}
//视频帧
Mat frame(s, CV_8UC3);
for(int i = 0; i < 100; i++)
{
   writer << frame;
}
```

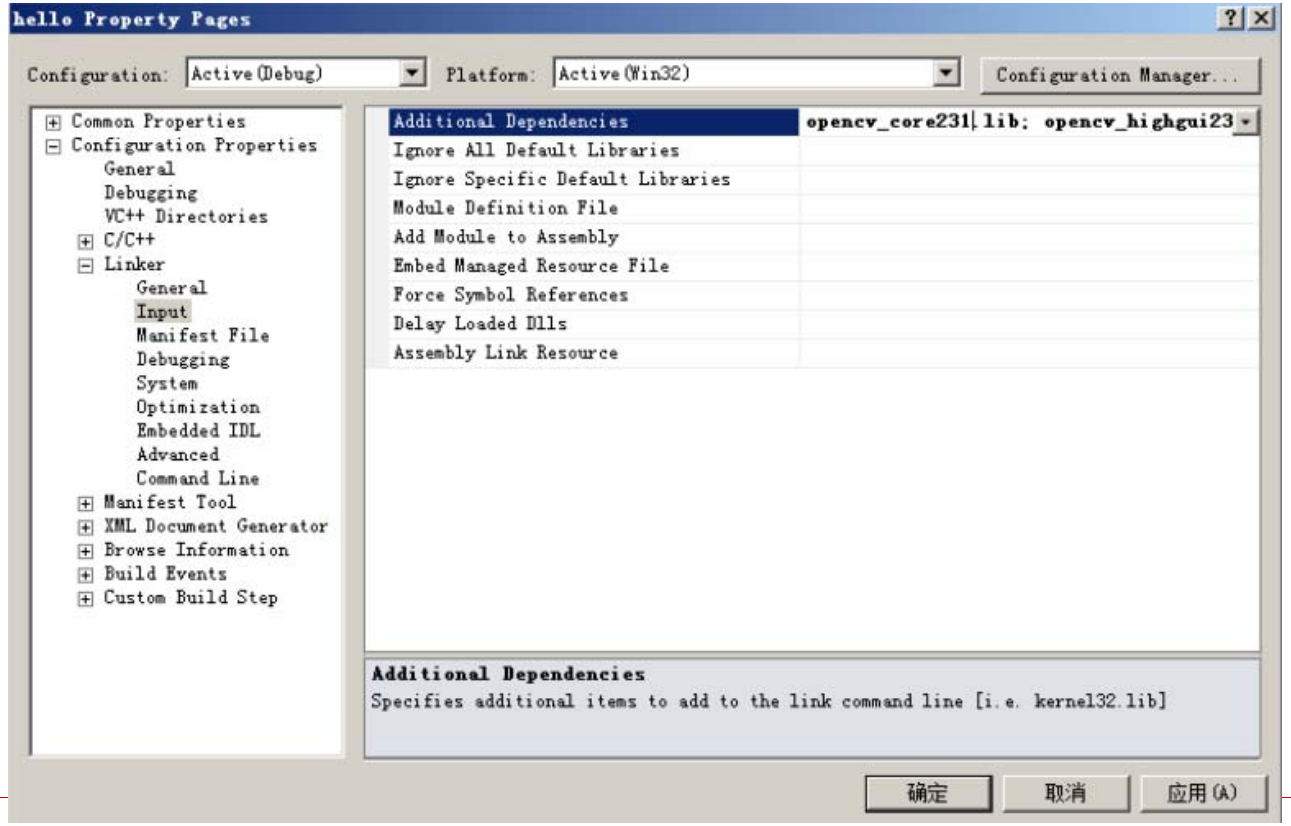
Common mistakes

hello.cpp(2): fatal error C1083: Cannot open include file: 'opencv2/opencv.hppp': No such file or directory

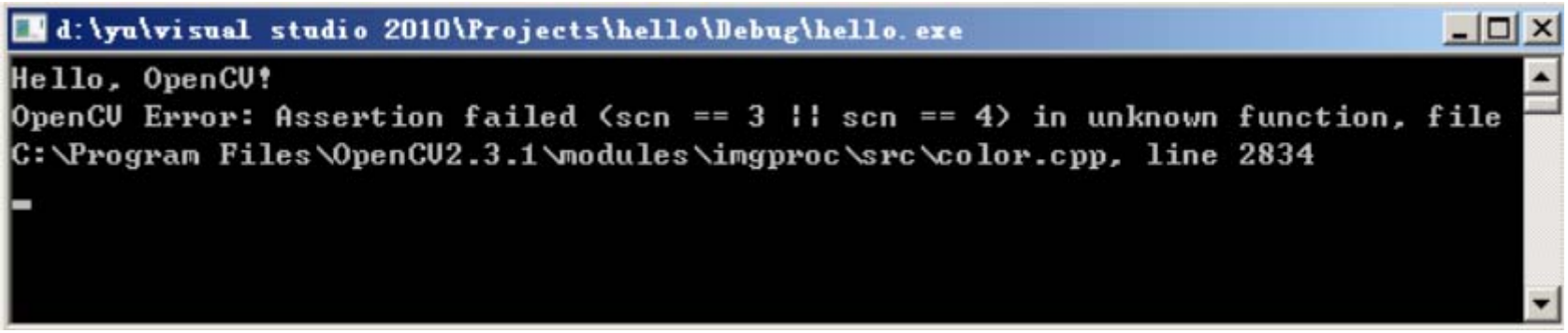


Common mistakes

```
1>hello.obj : error LNK2019: unresolved external symbol "class
cv::Mat __cdecl cv::imread(class std::basic_string<char,struct
std::char_traits<char>,class std::allocator<char> > const &,int)"
(?imread@cv@@YA?AVMat@1@ABV?$basic_string@DU?$char_traits@D@std@@
V?$allocator@D@2@@std@@H@Z) referenced in function _main
```

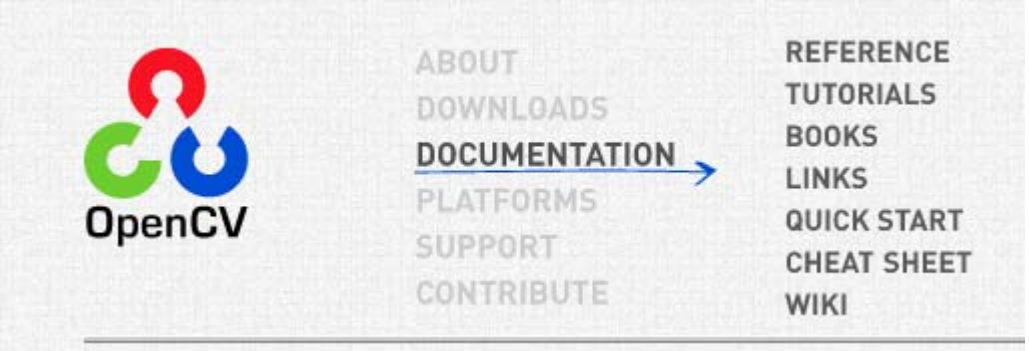


Common mistakes



推荐资料

- 手册：http://docs.opencv.org/
- 教程：http://docs.opencv.org/doc/tutorials/tutorials.html



- 进阶：https://github.com/opencv/opencv/wiki

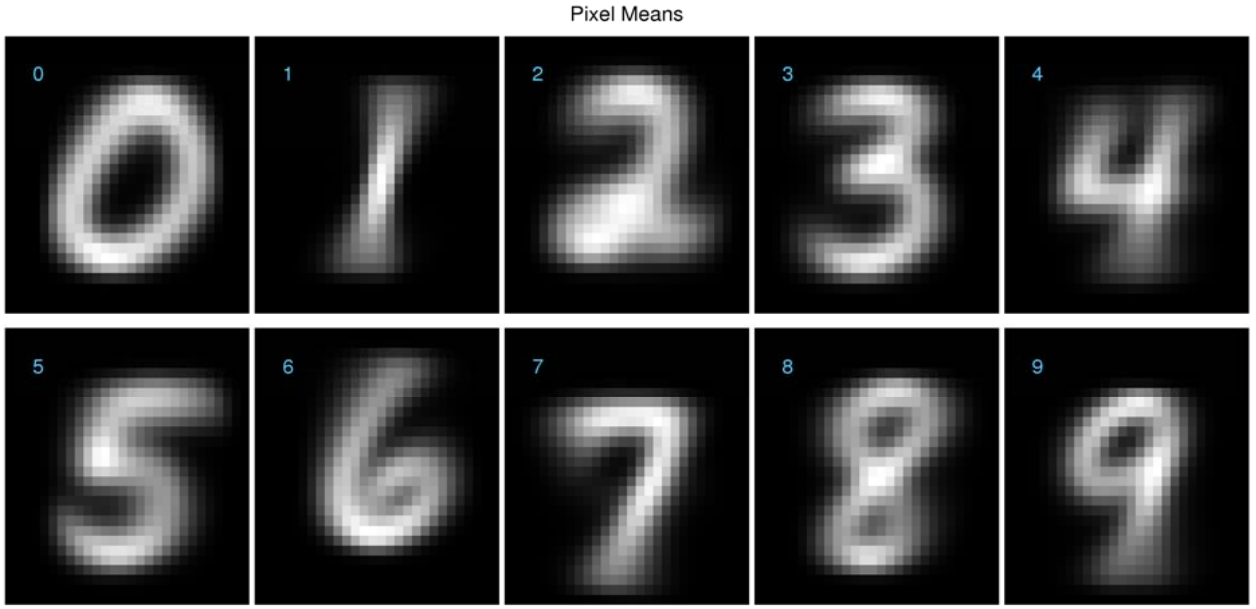
Machine Learning

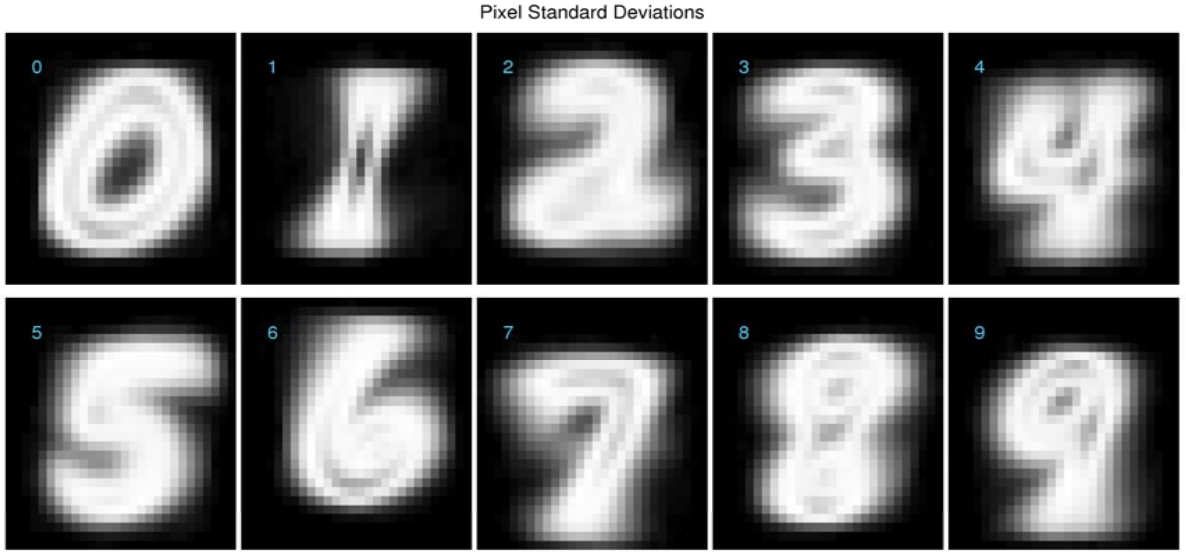
- Machine learning is about learning some properties of a data set and applying them to new data
  - supervised learning
  - unsupervised learning

手写字符识别 MNIST database of handwritten digits

- http://yann.lecun.com/exdb/mnist/
- 60,000 training
- 10,000 test

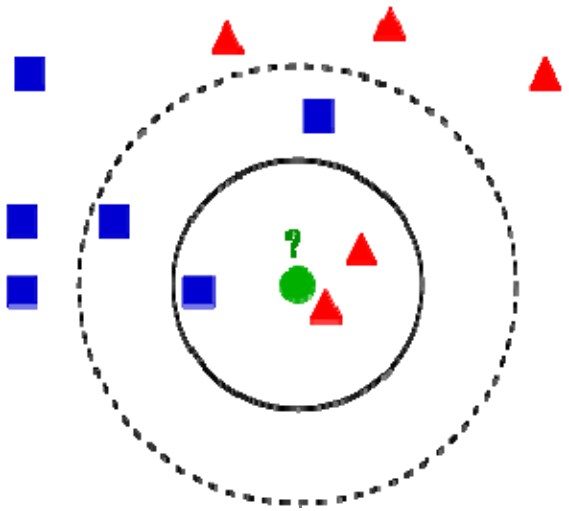
Example in memory


Pixel Means
0
1
2
3
4
5
6
7
8
9


Pixel Standard Deviations
0
1
2
3
4
5
6
7
8
9

Knn – Grassroots Democracy

- 目标:分类未知类别案例。
- 输入:待分类未知类别案例项目。已知类别案例集合D，其中包含 j个已知类别的案例。
- 输出:项目可能的类别。



Knn in OpenCV3

```
Ptr<ml::KNearest> knn = ml::KNearest::create();
Ptr<ml::TrainData> trainData =
ml::TrainData::create(train_features,ml::SampleTypes::ROW_SAMP
LE, labels);
knn->train(trainData);
```

```
Mat predictedLabels;
knn->findNearest(sample, K,
predictedLabels);

float prediction = predictedLabels.at<float>(0,0);
```

Knn in OpenCV 2.4

- int size = numRows*numCols;
- Mat trainingVectors(numImages, size, CV_32FC1);
- Mat trainingLabels(numImages, 1, CV_32FC1);

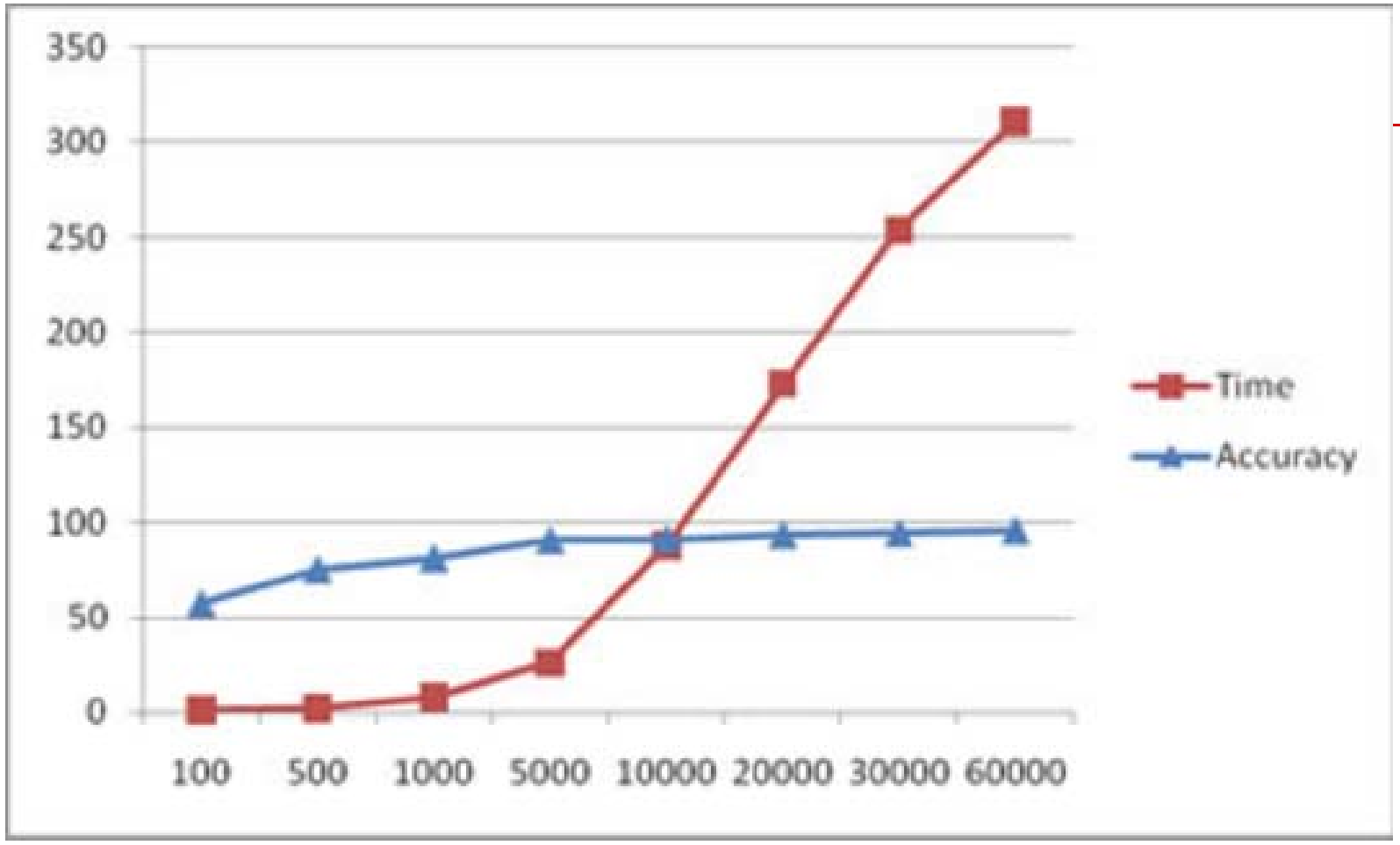
- KNearest knn(trainingVectors, trainingLabels);

- CvMat *currentTest = cvCreateMat(1, size, CV_32FC1);
- CvMat *currentLabel = cvCreateMat(1, 1, CV_32FC1);
- knn.find_nearest(currentTest, 5, currentLabel);

Results

- 60000 train + 10000 test 96.88%
- K = 5

Training samples	Time to test 1000 samples (sec)	Memory used (KB)	Accuracy (%)
100	1 s	4,328K	57.70 %
500	2 s	6,800K	75.50 %
1000	8 s	9,868K	81.50 %
5000	26 s	34,480 K	91.00 %
10000	88 s	65, 220 K	91.60 %
20000	173 s	128, 608 K	93.70 %
30000	254 s	188, 192 K	94.70 %
60000	311 s	372, 648 K	96.10 %


350
300
250
200
150
100
50
0
100
500
1000
5000
10000
20000
30000
60000
Time
Accuracy

Summary of KNN

- 优缺点：
  - （1）优点：

    算法简单，易于实现，不需要参数估计，不需要事先练。
  - （2）缺点：

    kNN计算量特别大，而且训练样本必须存储在本地，存开销也特别大。
- **K的取值：**
- 参数k的取值一般通常不大于20