

Opencv 资料

---- By Moco Sun 2010.12.19

1. 简介

OpenCV 的全称是: Open Source Computer Vision Library, Intel 公司支持的开源计算机视觉库, 采用 c/c++编写, 可以运行在 linux/windows/mac 等操作系统上。Opencv 还提供了 python、ruby、matlab 以及其他语言的接口。

其目标是构建一个简单易用的计算机视觉框架, 以帮助开发人员更便捷地设计更复杂的计算机视觉相关应用程序。Opencv 包含的函数有 500 多个, 覆盖了如工厂产品检测、医学成像、信息安全、用户界面、摄像机标定、立体视觉和机器人等, 具体将在下面介绍。

Opencv 使用宽松的 BSD 开源协议, 在遵守协议的情况下, 允许生成商业产品, 不必开发源代码。

Opencv 利用了 IPP (高性能多媒体函数库) 高度手工优化, 且在 intel 处理器上有更高的运行速度。

最新版本: 2.2, 项目网址 <http://sourceforge.net/projects/opencvlibrary/>

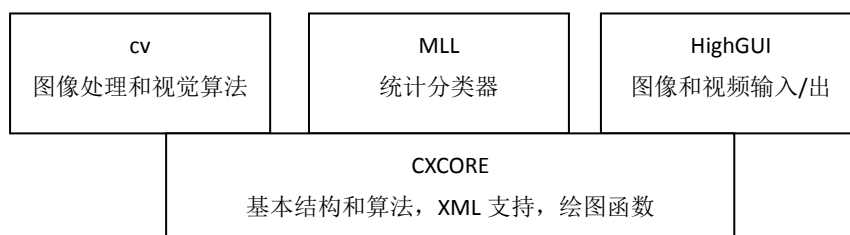
其他版本的 OpenCV:

opencv-extension-library: 扩展, <http://code.google.com/p/opencv-extension-library/>

opencvx, 另一个扩展, <http://code.google.com/p/opencvx/>

emguCV: C#版 opencv, 底层还是 c, <http://www.emgu.com>

opencv2.0 的结构:



2. Opencv 例子

2.1 显示图像

Opencv 可以读取各种类型的图像, 包括 BMP, DIB, JPEG, JPE, PNG, PBM, PGM, PPM, SR, RAS, TIFF 等, 下面程序显示了如何加载一幅图像并在屏幕上显示出来。显示结果如图 1 所示。

程序1: 从文件中读取一幅图像并在屏幕上显示

```
#include "highgui.h"
int main(int argc, char** argv)
{
    if(argc < 2)
        exit(1);
    //读入一张图片
    IplImage* image = cvLoadImage(argv[1]);
    if (NULL == image) //如果读入失败, 退出程序
        exit(1);
    //创建一个窗口, 标题为Example
    cvNamedWindow("Example", CV_WINDOW_AUTOSIZE);
    //在窗口Example中显示图片image
    cvShowImage("Example", image);
    //暂停程序, 等待用户触发一个按键
    cvWaitKey(0);
    //释放图像所分配的内存
    cvReleaseImage(&image);
    //销毁窗口
    cvDestroyWindow("Example");
    return 0;
}
```

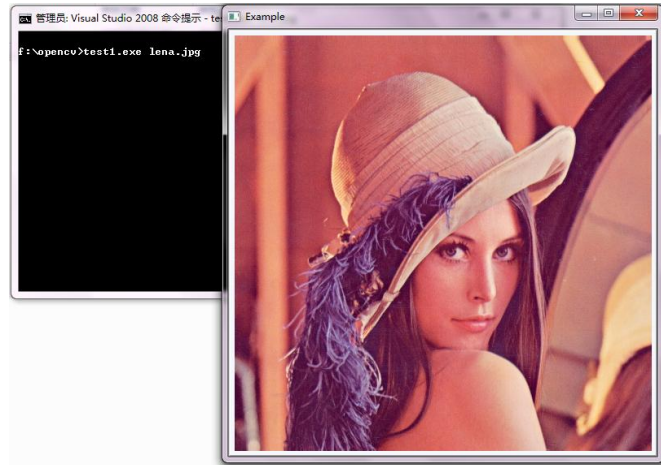


图 1 读入一幅图像并在屏幕上显示

● `IplImage* image = cvLoadImage(argv[1]);`

上面代码声明了一个 `IplImage` 图像的指针，然后根据图像名称的字符串 `argv[1]`，将该幅图像加载到内存。

图像结构体 `IplImage`（前几个字母是 `i` 的大写，小写 `p`，小写的 `L`）：

```
IplImage
|-- int nChannels;    // 颜色通道数目(1,2,3,4)
|-- int depth;       // 像素的位深:
|                   // IPL_DEPTH_8U, IPL_DEPTH_8S,
|                   // IPL_DEPTH_16U, IPL_DEPTH_16S,
|                   // IPL_DEPTH_32S, IPL_DEPTH_32F,
|                   // IPL_DEPTH_64F
|-- int width;       // 图像宽度（像素为单位）
|-- int height;      // 图像高度
|-- char* imageData; // 图像数据指针
|                   // 注意彩色图像按BGR顺序存储数据
|-- int dataOrder;   // 0 - 将像素点不同通道的值交错排在一起，形成单一像素平面
|                   // 1 - 把所有像素同通道值排在一起，形成若干个通道平面，再把平面排列起来
|                   // cvCreateImage 只能创建像素交错排列式的图像
|-- int origin;      // 0 - 像素原点为左上角，
|                   // 1 - 像素原点为左下角(Windows bitmaps style)
|-- int widthStep;   // 相邻行的同列点之间的字节数
|-- int imageSize;   // 图像的大小（字节为单位）= height*widthStep
|-- struct _IplROI* roi; // 图像的感兴趣区域（ROI）。ROI非空时对图像的
|                   // 处理仅限于ROI区域。
|-- char* imageDataOrigin; // 图像数据未对齐时的数据原点指针
|                   // (需要正确地重新分配图像内存)
|                   // (needed for correct image deallocation)
```

小贴士

初始化一个 `IplImage`，一般有以下两种途径：

以一个图像头初始化，通过 `cvCreateImageHeader()` 函数，图像头标志了一副图像的属性，不同深度 `depth`（可看作数据的类型，通常由8位，16位等，可在定义的时候通过 `IPL_DEPTH_8U` 这样的宏来声明）、不同通道的图像头不能混用，不然在操作时会出错。多个图像头可指向同一副图像或同一副图像的不同区域。

（注意，图像头要指向数据 `-> imageData` 后才能进行操作，不然它仅仅相当一个指针存在）

以一个图像创建，自定义方式创建，`cvCreateImage()`；或者通过拷贝创建，`cvCloneImage()`，这时两个图像的属性就自动一样了，或者通过 `cvLoadImage()` 从磁盘读入，或者从视频读入一帧图像 `cvQueryFrame`，后两者也不需人工设定图像属性。应该注意，使用 `cvCreateImage()` 创建空白图像时，记得用 `cvSetZero()` 等函数初始化，不然图像不是全白色的。

● `cvNamedWindow("Example", CV_WINDOW_AUTOSIZE);`

创建一个窗口，用“Example”标识，也是窗口标题栏上显示的名称，第二个参数定义

了窗口的属性。该参数可以省略，默认是 1，宏定义为 CV_WINDOW_AUTOSIZE，自动根据图像的大小调整窗口大小，2.0 目前只支持一个值。

小贴士

这里无法自由指定窗口的大小，因为这涉及了图像的缩放方法，opencv2.0暂时没有支持这个功能，不过可以通过创建另外尺寸的一幅图像，使用cvResize函数，将需要改变的图像变化到该幅图像上，这个函数可以选择实现插值的方法。

也有另外一个函数cvResizeWindow是调整窗口大小，不过它只是简单地截取了一个窗口的一部分，其他部分不会显示。

● cvShowImage("Example",image);

这个函数作用是在窗口“Example”中显示我们读入的 image 这幅图像。第一个参数是 const char* name 型，表示窗口的名称，第二个参数是 const CvArr* image 型。这里的 CvArr 其实上是这样定义的 typedef void CvArr，它的作用是作为一个函数参数，指定了一个函数可以接受多种类型的参数，比如 IplImage，还有矩阵结构体 CvMat，或者点序列 CvSeq。

● cvWaitKey(0);

比较方便的函数，它的功能是使程序暂停，当参数是 0 或负数时，只有当用户触发一个按键时，程序才继续向下运行，当参数是正整数时，表示暂停一段时间，单位是毫秒。

函数返回值是一个 int，是用户按键的 ASCII 码的整数值。

小贴士

cvWaitKey还有另外一个用处是让出CPU的计算时间，在读取视频中我们会经常使用一个循环，在循环里面读取图像，这时如果不使用cvWaitKey函数的话，窗口显示的内容会一直不变的，这跟线程间的调度有关。

● cvReleaseImage(&image); cvDestroyWindow("Example");

最后两句作用是释放图像所占的内存，释放为窗口所分配的所有内存（包括窗口内部的图像内存缓冲区，该缓冲区中保存了与图像指针相关的图像文件像素信息的一个副本）。

2.2 访问图像数据

我们通常需要对图像数据进行迅速而高效的访问，我们可以直接读取 IplImage 结构体的内容，下面这个例子展示了这个过程。图 2 是对图像像素值都增加了 10 后得到的结果。

程序2：访问图像数据，使所有像素值增加10

```
#include "highgui.h"
int main(int argc, char** argv)
{
    //读入一张图片
    IplImage* image = cvLoadImage("lena.jpg");
    if (NULL == image) //如果读入失败，退出程序
        exit(1);
    //创建一个窗口,标题为Example
    cvNamedWindow("Example", CV_WINDOW_AUTOSIZE);

    //用指针指向图像的数据区头部
    uchar *pchar;
    int width = image->width;           //读取图像宽度
    int height = image->height;          //读取图像高度
    int channel = image->nChannels;       //读取图像通道数
    int widthStep = image->widthStep;    //读取图像一行像素所占的字节数
    int i, j;
    for (i=0; i<height; i++)
```

```

{
    pchar = (uchar*)image->imageData + i*widthStep;
    for (j=0; j<width; j++)
    {
        uchar* temp = pchar + j*channel;
        temp[0] += 10; //通道B
        temp[1] += 10; //通道G
        temp[2] += 10; //通道R
    }
}

cvShowImage("Example",image);
cvWaitKey(0);
cvReleaseImage(&image);
cvDestroyWindow("Example");
return 0;
}

```

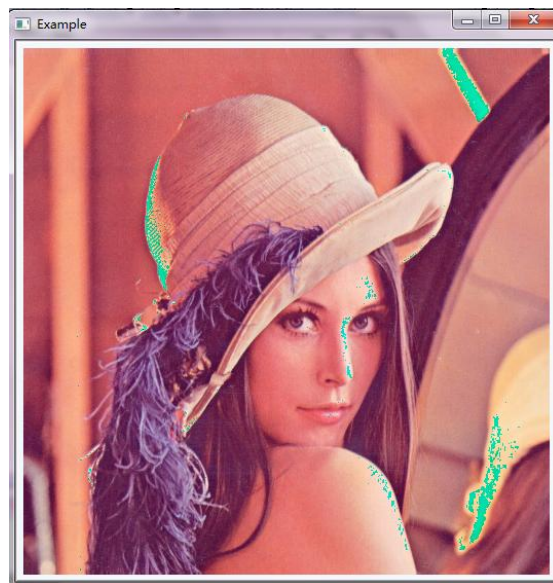


图 2 原图像像素增加 10 后

为了让图像的像素值改变，一个有效的办法就是遍历图像的所有像素，修改其值，`IplImage` 结构的图像数据区头部存放在 `imageData` 中；此外，我们要必须获得图像的数据类型、高度、宽度以及一行像素的字节数量。

其中图像所使用的数据类型决定了我们作为临时变量的指针（如上面的 `pchar`、`temp`）所使用的变量类型。

图像的数据类型：上面程序使用的 `cvLoadImage()` 函数读入的图像像素的类型是 8 位的整数，这也就是 `pchar` 是 `uchar*` 型的原因。

OpenCV 支持的图像数据类型有很多，如果使用 `cvCreateImage` 函数创建图像就可以指定所使用的数据类型，它的函数原型如下

```
IplImage* cvCreateImage(CvSize size,int depth,int channels );
```

CvSize 表示图像的大小，为含两个 `int` 的结构体，定义如下

```
typedef struct CvSize
{
    int width;
    int height;
}
```

CvSize;

小贴士

Cv与cv的区别：以Cv开头的一般是函数，以cv开头的通常是内联数据元素。
CvPoint结构体不支持默认构造函数，但是可以通过inline的cvPoint(注意首字母小写)函数来创建一个无名的CvPoint，这在传递一些函数参数经常使用，同理，CvScalar与cvScalar,CvSize与cvSize等也有这样的用法。

int Depth定义了图像的深度，可以理解为图像的数据类型，Opencv用许多宏来表示不同的图像类型，如下表

表1 Opencv图像类型

宏	图像像素类型
IPL_DEPTH_8U	无符号8位整数（8u）
IPL_DEPTH_8S	有符号8位整数（8s）
IPL_DEPTH_16S	有符号16位整数（16s）
IPL_DEPTH_32S	有符号32为整数（32s）
IPL_DEPTH_32F	32位浮点数单精度（32f）
IPL_DEPTH_64F	64位浮点数双精度（64f）

图像通道 nchannel：Opencv里图像通道可以有4个，即一般的RGB+alpha通道，在创建图像时的nchannels整数参数就是设置通道数的，1通道的图像都称为灰度图像，彩色图像独立查看每个通道，看到的图像也是灰色的。

正如前面介绍的，默认情况下，->imageData 是用交错形式存放图像的各个通道的数值的，如下图所示。

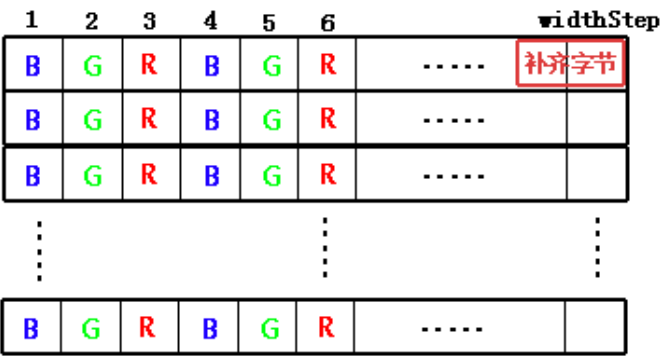


图3 图像像素排列方式

可以通过cvCvtColor()进行图像空间的转换，如RGB或BGR到灰度（反过来有Bayer到彩色），HSV与RGB互相转换等等，比较方便。

图像一行字节数 widthStep：这个容易跟图像宽度width混淆，遍历时如果在计算每一行开始的地址时用width去乘以行序号容易出错，这是因为为了处理过程高效，每行像素都会用固定的字节数来对齐，因此在第i行末和第i+1行开始出可能会有补齐字节（见图3）。

小贴士

除了自行定位像素的位置外，Opencv还提供了一些函数，可以方便地取得或者设置矩阵元

素或者图像中像素的值。如下

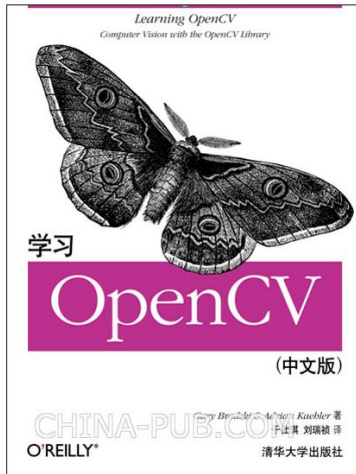
```
Double cvGetReal1D(const CvArr* arr,int idx0)
Double cvGetReal2D(const CvArr* arr,int idx0,int idx1)
void cvSetReal1D(const CvArr* arr,int idx0,double value)
void cvSetReal2D(const CvArr* arr,int idx0,int idx1, double value)
```

```
CvScalar cvGet1D(const CvArr* arr,int idx0)
CvScalar cvGet1D(const CvArr* arr, int idx0,int idx1)
void cvSet1D(const CvArr* arr,int idx0, CvScalar value)
void cvSet2D(const CvArr* arr,int idx0,int idx1, CvScalar value)
```

其中CvScalar是一个三个double组成结构体，表示一个像素。

3. 参考资料

● 参考书籍：



《Learning opencv》，作者：Gary Bradski, Adrian Kaehler，东南大学出版社

《学习 OpenCV 中文版》，译者：于仕琪 刘瑞祯，清华大学出版社

《OpenCV 中文教程》，刘瑞祯 于仕琪，北京航空航天大学出版社

● 互联网资料：

Opencv 中文网站 <http://www.opencv.org.cn>：提供基本的 opencv 参考、例程，还可以互动提问（论坛）

维基百科（英文）：http://en.wikipedia.org/wiki/Main_Page，海纳百川，很多知识的整理。

雅虎 GROUPS: <http://tech.groups.yahoo.com/group/OpenCV/>，信息交流

开源项目网址：<http://sourceforge.net/projects/opencvlibrary/>

附 Opencv 支持的功能描述

图像数据操作（内存分配与释放，图像复制、设定和转换）

图像/视频的输入输出（支持文件或摄像头的输入，图像/视频文件的输出）

矩阵/向量数据操作及线性代数运算（矩阵乘积、矩阵方程求解、特征值、奇异值分解）

支持多种动态数据结构（链表、队列、数据集、树、图）

基本图像处理（去噪、边缘检测、角点检测、采样与插值、色彩变换、形态学处理、直方图、图像金字塔结构）

结构分析（连通域/分支、轮廓处理、距离转换、图像矩、模板匹配、霍夫变换、多项式逼近、曲线拟合、椭圆拟合、狄劳尼三角化）

摄像头定标（寻找和跟踪定标模式、参数定标、基本矩阵估计、单应矩阵估计、立体视觉匹配）

运动分析（光流、动作分割、目标跟踪）

目标识别（特征方法、HMM 模型）

基本的 GUI（显示图像/视频、键盘/鼠标操作、滑动条）

图像标注（直线、曲线、多边形、文本标注）