

**课程名称：软件工程与计算||**

**课程题目：Opencv环境配置**

院（系）：计算机与信息工程学院

专 业：软件工程

班 级：13软件

学 号：20131105752

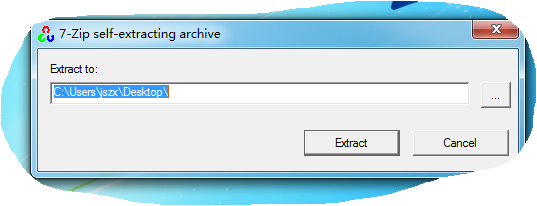
姓 名：杨泽奇

指导教师：朝力萌

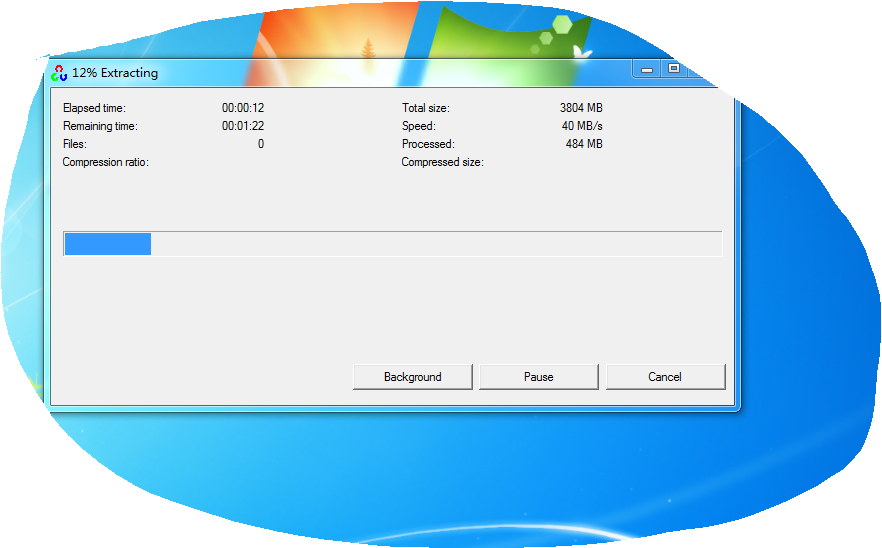
完成日期： 2017-01-03

**1 下载安装OPENCV**

（1）在官网：[http://opencv.org/](http://opencv.org/" \t "_blank)上找到OpenCV windows版下载下来。下载完后得到文件OpenCV 2.4.8，双击后会提示解压到某个地方



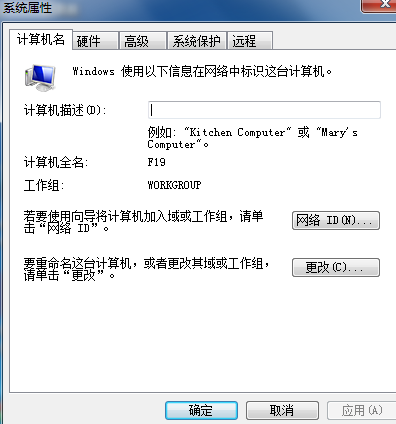
（2）下载完成后，双击文件后会自动提取文件，根据需求自己选择提取文件夹目录;

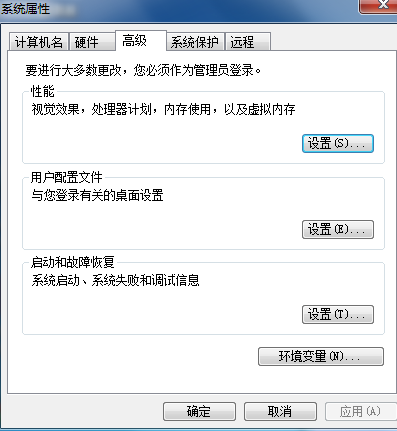


（3）提取完成后，即完成了OpenCV的安装;

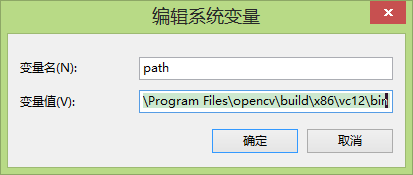
**2 配置环境变量**

找到计算机图标，右键鼠标找到属性，页面的右下角处有更改设置的按钮，点击鼠标左键，找到高级标题这一栏，点击鼠标左键，弹出窗口，找到系统变量，编辑path变量的值，在系统变量中找到Path变量并在最后加上下面语句; C:\opencv\build\x86\vc10\bin

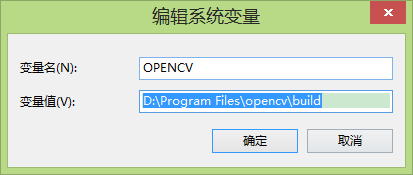




编辑其中的path变量，并添加变量值：;D:\Program Files\opencv\build\x64\vc12\bin;D:\Program Files\opencv\build\x86\vc12\bin（添加两个bin目录是因为有x86和x64位操作系统，为了防止不兼容，才这样做的）



新建一个变量名为OPENCV，变量值为：D:\Program Files\opencv\build

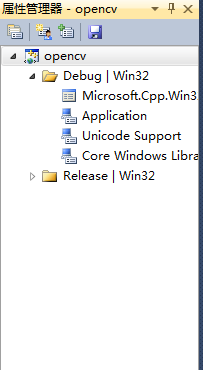


**一 VS2013配置**

**（一）工程包含（include）目录的配置**

1. 打开VS2010，找到<视图>--<属性管理器>；

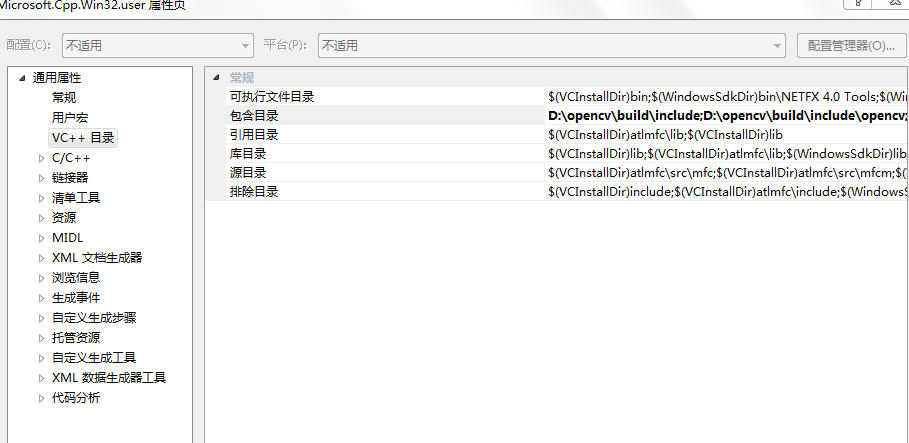
2. 在新出现的“属性管理器”工作区中，点击项目->Debug|Win32->Microsoft.Cpp.Win32.userDirectories即可打开属性页面；

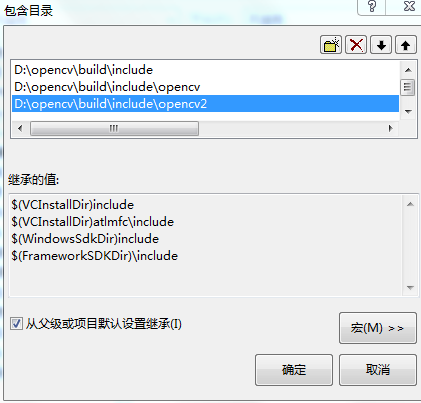


3. 在【通用属性】 ->【VC++目录】 ->【包含目录中】复制路径D:\Program Files\opencv\build\include

D:\Program Files\opencv\build\include\opencv

D:\Program Files\opencv\build\include\opencv2





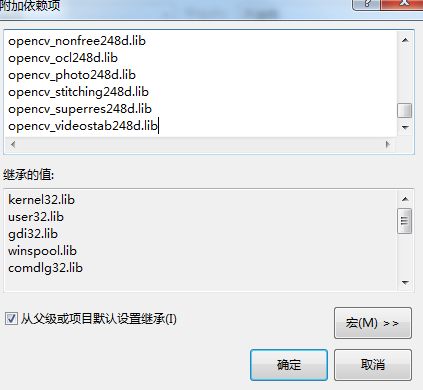
**（二） 工程库（lib）目录的配置**

1. 属性管理器”工作区中，点击项目->Debug|Win32->Microsoft.Cpp.Win32.userDirectories即可打开属性页面。【通用属性】 ->【VC++目录】 ->【库目录】，添加上D:\ProgramFiles\opencv\build\x86\vc10\lib这个路径。

**（三） 链接库的配置**

1. “属性管理器”工作区中，点击项目->Debug|Win32->Microsoft.Cpp.Win32.userDirectories即可打开属性页面，【通用属性】 ->【链接器】->【输入】->【附加的依赖项】。添加如下的内容：

opencv\_calib3d248.lib  
opencv\_contrib248.lib  
opencv\_core248.lib  
opencv\_features2d248.lib  
opencv\_flann248.lib  
opencv\_gpu248.lib  
opencv\_highgui248.lib  
opencv\_imgproc248.lib  
opencv\_legacy248.lib  
opencv\_ml248d.lib  
opencv\_objdetect248.lib  
opencv\_ts248.lib  
opencv\_video248.lib  
opencv\_nonfree248.lib  
opencv\_ocl248.lib  
opencv\_photo248.lib  
opencv\_stitching248.lib  
opencv\_superres248.lib  
opencv\_videostab248.lib  
opencv\_calib3d248d.lib  
opencv\_contrib248d.lib  
opencv\_core248d.lib  
opencv\_features2d248d.lib  
opencv\_flann248d.lib  
opencv\_gpu248d.lib  
opencv\_highgui248d.lib  
opencv\_imgproc248d.lib  
opencv\_legacy248d.lib  
opencv\_objdetect248d.lib  
opencv\_ts248d.lib  
opencv\_video248d.lib  
opencv\_nonfree248d.lib  
opencv\_ocl248d.lib  
opencv\_photo248d.lib  
opencv\_stitching248d.lib  
opencv\_superres248d.lib  
opencv\_videostab248d.lib

****

**（四）程序代码**

**（1）人脸识别按钮**

void COpencvPictuteDlg::OnBnClickedRun()

{

cascade = (CvHaarClassifierCascade\*)cvLoad(cascade\_name, 0, 0, 0);

if (!cascade)

{

MessageBox(L"没有找到人脸训练器文件");

return;

}

storage = cvCreateMemStorage(0);

cvNamedWindow("人脸识别", 1);

IplImage\* image = cvLoadImage((char\*)(\_bstr\_t)m\_strFilePath, 1);

if (image)

{

detect\_and\_draw(image);

cvReleaseImage(&image);

}

cvDestroyWindow("人脸识别");

this->ShowWindow(SW\_SHOW);

}

void COpencvPictuteDlg::detect\_and\_draw(IplImage\* img)

{

double scale = 0.7;

static CvScalar colors[] = {

{ 0, 0, 255 }, { 0, 128, 255 }, { 0, 255, 255 }, { 0, 255, 0 },

{ 255, 128, 0 }, { 255, 255, 0 }, { 255, 0, 0 }, { 255, 0, 255 }

};

//图像准备

IplImage\* gray = cvCreateImage(cvSize(img->width, img->height), 8, 1);

IplImage\* small\_img = cvCreateImage(cvSize(cvRound(img->width / scale), cvRound(img->height / scale)), 8, 1);

cvCvtColor(img, gray, CV\_BGR2GRAY);

cvResize(gray, small\_img, CV\_INTER\_LINEAR);

cvEqualizeHist(small\_img, small\_img);

//检测到对象执行以下操作

cvClearMemStorage(storage);

double t = (double)cvGetTickCount();

CvSeq\* objects = cvHaarDetectObjects(small\_img, cascade, storage, 1.1, 2, 0, cvSize(30, 30));

t = (double)cvGetTickCount() - t;

for (int i = 0; i<(objects ? objects->total : 0); ++i)

{

CvRect\* r = (CvRect\*)cvGetSeqElem(objects, i);

cvRectangle(img, cvPoint(r->x\*scale, r->y\*scale), cvPoint((r->x + r->width)\*scale, (r->y + r->height)\*scale), colors[i % 8]);

}

int i;

CString str;

for (i = 0; i < (objects ? objects->total : 0); i++)

{

CvRect\* r = (CvRect\*)cvGetSeqElem(objects, i);

CvPoint center;

int radius;

center.x = cvRound((r->x + r->width\*0.5)\*scale); //获取圆心X坐标

center.y = cvRound((r->y + r->height\*0.5)\*scale);//获取圆心Y坐标

radius = cvRound((r->width + r->height)\*0.25\*scale); //获取半径

//对img脸部进行圆圈标注。参数为：图像，圆心坐标，圆心半径，线条颜色，线条粗细负数表示填充元，线条类型，圆心坐标点和半径值的小数点位数

cvCircle(img, center, radius, colors[i % 8], 1, 8, 0);

}

str.Format(L"一共检测到%d张人脸,耗时%g毫秒", i, t / ((double)cvGetTickFrequency() \* 1000));

cvShowImage("人脸识别", img); //显示结果窗口

cvReleaseImage(&gray); //显示图像

cvReleaseImage(&small\_img);//显示图像

this->ShowWindow(SW\_HIDE);

MessageBox(str);

}

**（2）轮廓标记：**

void COpencvPictuteDlg::OnBnClickedRun2()

{

src = imread((char\*)(\_bstr\_t)m\_strFilePath);

/// 转成灰度并模糊化降噪

cvtColor(src, src\_gray, CV\_BGR2GRAY);

blur(src\_gray, src\_gray, Size(3, 3));

/// 创建窗体

char\* source\_window = "Source";

namedWindow(source\_window, CV\_WINDOW\_AUTOSIZE);

imshow(source\_window, src);

createTrackbar(" Canny thresh:", "Source", &thresh, max\_thresh, thresh\_callback);

thresh\_callback(0, 0);

waitKey(0);

}

void thresh\_callback(int, void\*)

{

Mat canny\_output;

vector<vector<Point> > contours;

vector<Vec4i> hierarchy;

/// 用Canny算子检测边缘

Canny(src\_gray, canny\_output, thresh, thresh \* 2, 3);

/// 寻找轮廓

findContours(canny\_output, contours, hierarchy, CV\_RETR\_TREE, CV\_CHAIN\_APPROX\_SIMPLE, Point(0, 0));

/// 绘出轮廓

Mat drawing = Mat::zeros(canny\_output.size(), CV\_8UC3);

for (int i = 0; i< contours.size(); i++)

{

Scalar color = Scalar(rng.uniform(0, 255), rng.uniform(0, 255), rng.uniform(0, 255));

drawContours(drawing, contours, i, color, 2, 8, hierarchy, 0, Point());

}

/// 在窗体中显示结果

namedWindow("Contours", CV\_WINDOW\_AUTOSIZE);

imshow("Contours", drawing);

}

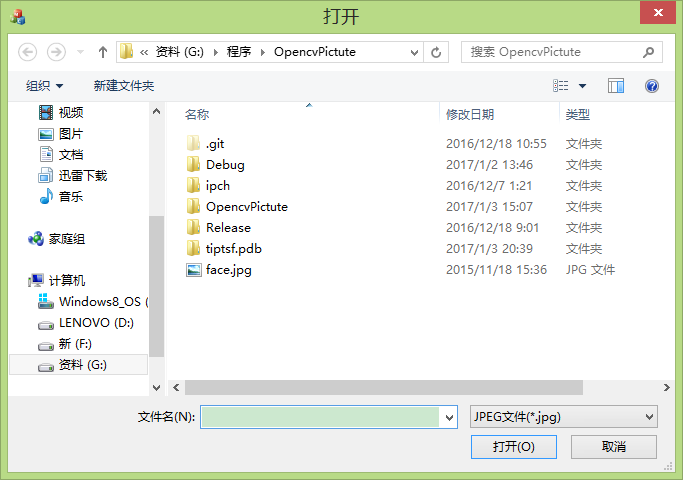
**程序运行展示图：**

**人脸识别：**

**（1）运行程序的第一个界面**



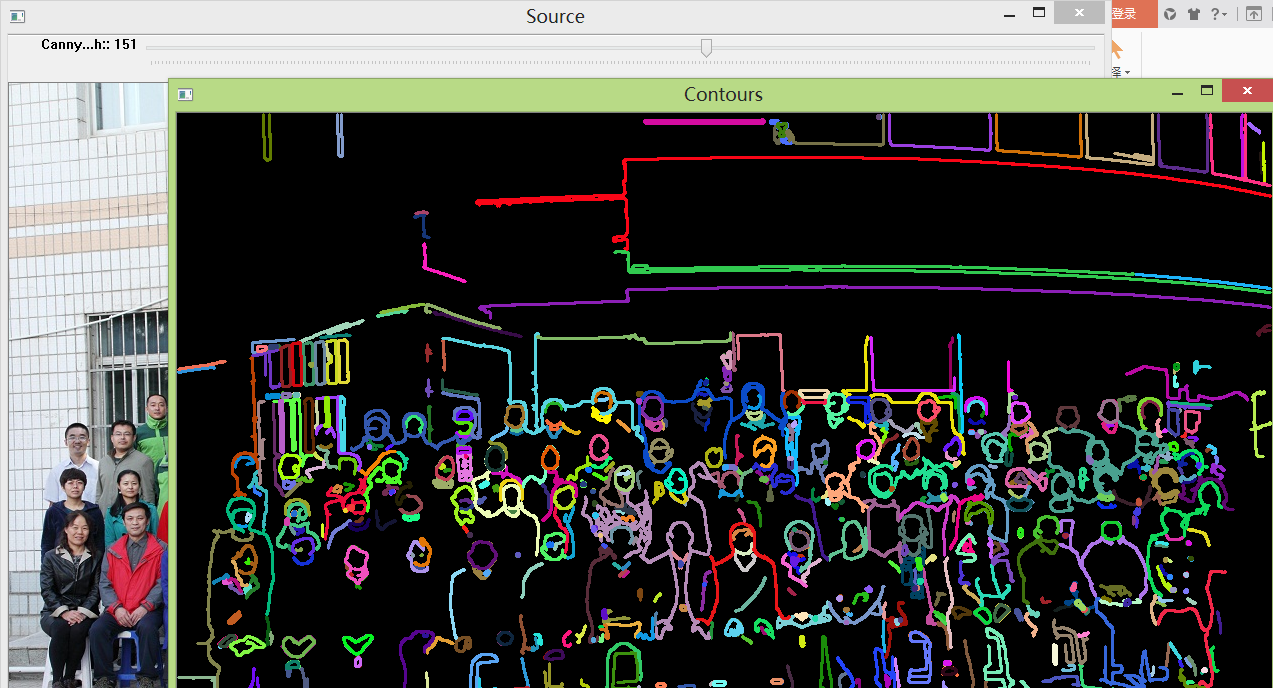
1. **点击浏览按钮，弹出选择文件的对话框**



1. **选择文件后点击打开，并点击第一个对话框中的人脸识别按钮弹出的界面**



**（4）轮廓标记界面**



**（五）可能出现的错误提示：**

**(1)无法解析的外部命令**

解决：编译器是win32，就用x86，编译器是X64，就用X64。不过一般情况下，都是用的win32的X86编译器。所以，无论32还是 64位操作系统，配置文件最好都选择x86版的。

**(2) 应用程序无法正常启动0xc000007b**

解决：lib包含的问题。也许你同时包含了X86和X64的，或者包含出错了。

**(3) 提示指针越界，有未经处理的异常**

解决：工程属性里面关于带d和不带d的lib文件的附加依赖项的问题，就算配置好了每次想debug和release下都运行还得手动在工程属性里面加。当得到这样的错误时，可以把调试方式改一改，debug和release变换：

打开当前工程的属性页，debug或者release哪个报错，就把对应的带d或不带d的lib添加到 属性->链接器->输入->附加的依赖项下即可。

**（六）课程设计小结**

**在实验中发现——分析——总结的问题有：进行编码前应该尽可能详细的把设计先写出来。我们现在虽然配置的是一个比较小的软件，但是由于初始时没有将计算机的设计实现和在运行过程中计算机所处状态的装换想好，而是采取了配置编码边设计的这个严重偏离了软件工程的正确思想的错误行为，导致了在测试中出现比较多的bug。**