1.4 安装OpenCV

下面介绍如何在不同的操作系统中下载和安装OpenCV。既可以从OpenCV的官方网站下载最新版本以及其他版本的全部源代码，也可以从GitHub上下载各种版本包括最新版本的源代码。OpenCV的一些新的、更高级别的功能通常放在附加模块opencv\_contrib中，可以在GitHub上搜索opencv\_contrib，下载并编译该模块。opencv\_contrib中某些算法的API开发稳定后会逐步移入主模块中。

1.4.1 在Windows系统中安装和配置OpenCV

本节以 Windows10 64 位操作系统为例，介绍在 Windows 系统中如何安装和配置OpenCV4.5.0。

1．安装OpenCV

在Windows环境下，最简单的安装方式是直接从OpenCV官方网站下载预编译好的OpenCV库进行自解压缩并安装，当前最新版本是4.5.0。

双击下载的自解压缩文件opencv-4.5.0-vc14\_vc15.exe，将其解压缩至指定文件夹，如F盘software目录。解压缩完成后会自动创建F:\opencv文件夹。为了与本机安装的其他版本的OpenCV有所区别，也可以将文件夹改名为F:\opencv4.5.0。如果本机未安装其他版本的OpenCV，则可不用更名。

2．配置OpenCV

OpenCV4.5.0的预编译版本支持vc14和vc15两个版本的Visual Studio，对应vs2015和vs2017。本书以vs2015为例说明OpenCV的配置方法。

1）创建用户环境变量

创建用户环境变量是为了后续在项目配置时更加便捷地配置各种目录。

单击“此电脑”→“右键属性”→“高级系统设置”→“系统属性”→“高级”→“环境变量”→“用户变量”→“新建”选项，打开如图1-1所示的“新建用户变量”对话框，填写如下选项。

* 变量名：OPENCV
* 变量值：F:\opencv4.5.0\build

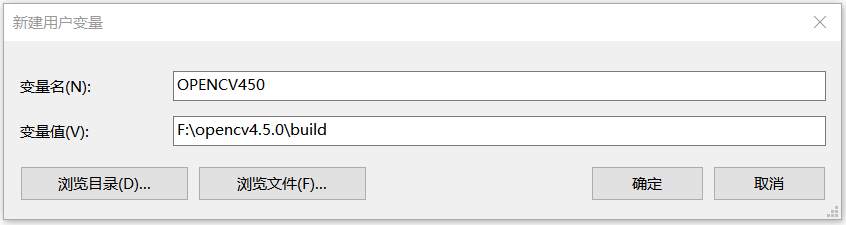


图1-1 “新建用户变量”对话框

如果系统安装了多个版本的OpenCV，刚在上述步骤中为了区分于其他版本，也可以将用户变量名命名为OPENCVxxx，其中xxx为版本号，例如OPENCV450，以便记忆和使用。

2）编辑系统环境变量

编辑系统环境变量Path，以便编译器能在正确路径下找到OpenCV4.5.0的动态链接库。

单击“计算机”→“右键属性”→“高级系统设置”→“系统属性”→“高级”→“环境变量”→“系统变量”→“Path”→“编辑”选项，打开如图1-2所示的“编辑环境变量”对话框，填写如下信息。

* 编辑系统环境变量：Path
* 增加：%OPENCV%\x64\vc15\bin

此处的%OPENCV%就是对上文新建的用户变量“OPENCV450”（D:\opencv4.5.0\build）的引用。需要注意的是，如果是在win10系统中编辑环境变量，则变量之间用英文“;”分隔，vc15是vs2017的版本号。

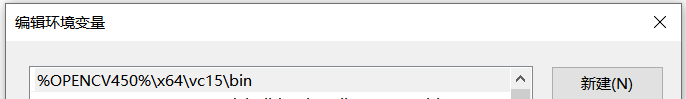


图1-2 “编辑环境变量”对话框

接下来新建一个vs2017项目，在项目中配置OpenCV4.5.0。启动vs2017，新建一个Win32控制台应用程序，名称为ch1\_example\_1\_1，其他选项默认即可，如图1-3所示。

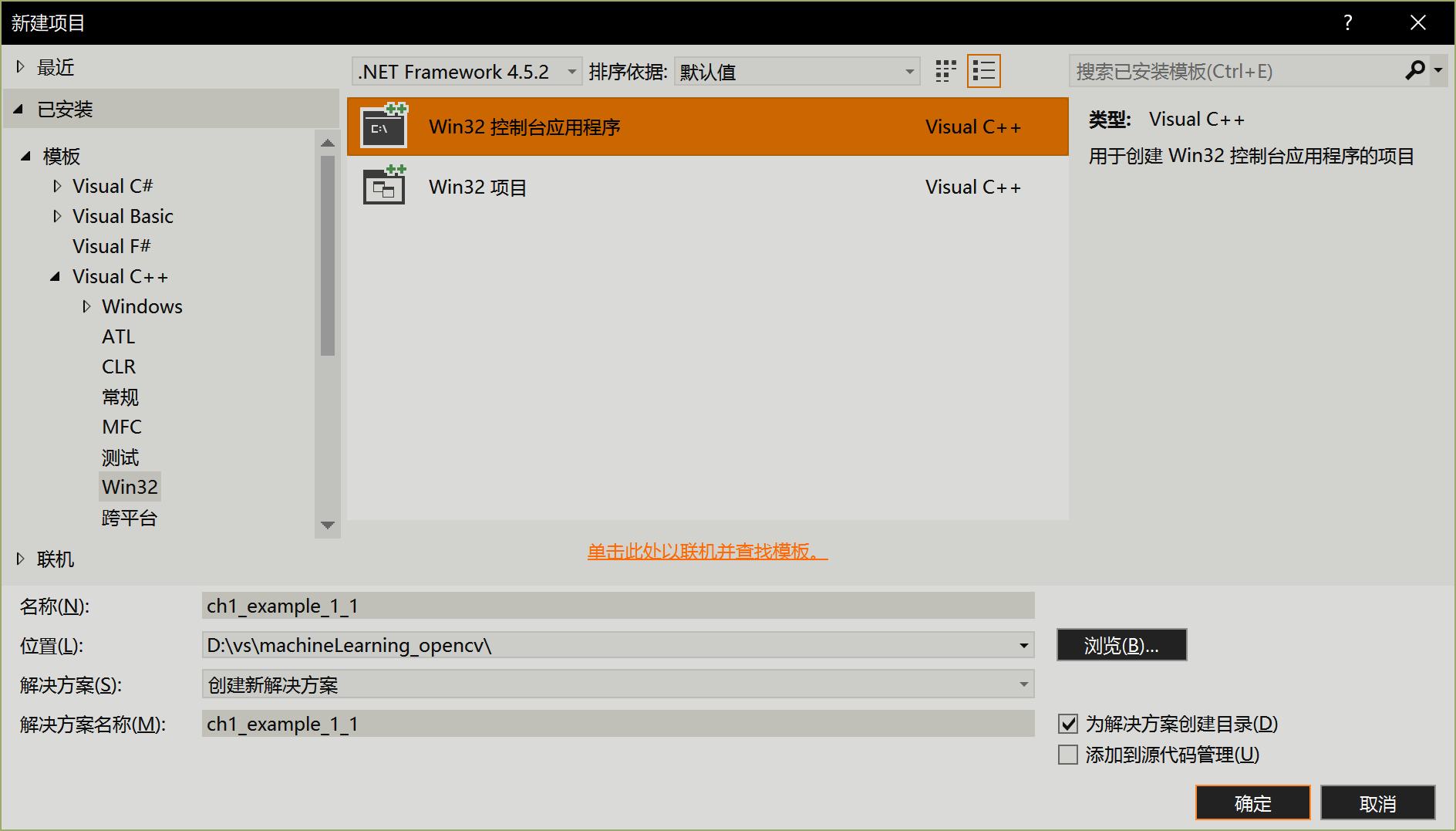


图1-3 “新建项目”对话框

因为OpenCV3之后只支持x64平台，因此先在vs2017的工具栏上将项目的解决方案平台改为x64。如果在下拉列表中没有x64选项，则先通过配置管理器新建x64平台，再选择x64平台。解决方案配置默认Debug，无须修改。配置项目的解决方案与平台如图1-4所示。注意，如果解决方案配置改为Release，则需要按下文所述重新配置一次OpenCV。

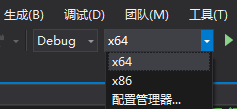


图1-4 配置项目的解决方案与平台

在解决方案资源管理器的项目名称“ch1\_example\_1\_1”上单击右键，在弹出的右键快捷菜单中单击“属性”选项，或者选择菜单栏上的“项目→属性”选项，打开本项目的属性配置页面进行配置。

3）添加包含目录

添加包含目录是为了使程序能正确地找到OpenCV的头文件。

单击“项目”→“属性”→“配置属性”→“VC++目录”选项，打开“包含目录”对话框，添加：

$(OPENCV450)\include\opencv2

$(OPENCV450)\include

“包含目录”对话框如图1-5所示。

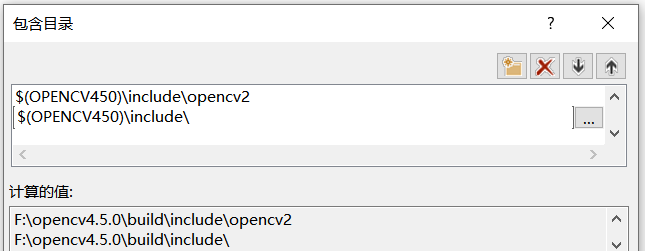


图1-5 “包含目录”对话框

4）添加库目录

添加库目录是为了让程序能正确地找到OpenCV的动态链接库的引入库lib文件。

单击“项目”→“属性”→“配置属性”→“VC++目录”选项，打开“库目录”对话框，添加：

$(OPENCV)\x64\vc14\lib

“库目录”对话框如图1-6所示。

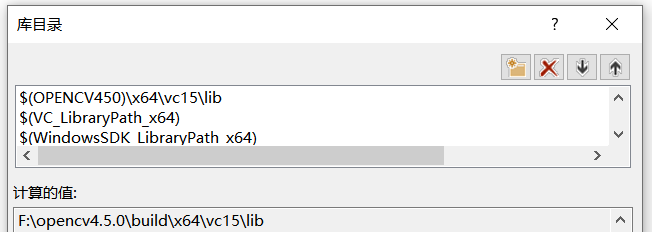


图1-6 “库目录”对话框

在配置上述目录时，用到了前文新建的用户环境变量OPENCV450。这样做的好处是，当把项目复制到其他计算机上运行时，由于OpenCV的安装目录可能不同，只需修改迁移后的计算机户环境变量OPENCV450对应的值即可，不需要逐一在项目属性中修改目录。当然，如果不使用用户环境变量，也可以直接使用相应的绝对路径来配置包含目录与库目录。

5）添加附加依赖项

添加库目录是为了让程序能正确地找到OpenCV的动态链接库的引入库lib文件。

此处我们先打开OpenCV的安装目录..\opencv\build\x64\vc15\lib，查看动态链接库的引入库文件名，例如在笔者的电脑中，lib文件如下：

opencv\_world450.lib

opencv\_world450d.lib

上述两个lib文件分别对应着Release版与Debug版的程序。

打开“项目”→“属性”→“配置属性”→“链接器”→“输入”→“附加依赖项”→“编辑”，添加：

opencv\_world450d.lib

如果配置是Debug模式，则添加opencv\_world450d.lib。如果配置是Release模式，则添加opencv\_world430d.lib。这里添加的是opencv\_world450d.lib。

通过上述步骤我们就成功地配置了OpenCV4.5.0。下面编写一段简单的程序来检验配置的结果。

3．测试程序

在测试项目的ch1\_example\_1\_1.cpp文件中编写如下代码，调试运行后将弹出一个图片对话框，显示E:\images\文件夹中的图片Lena.jpg，如图1-7所示。单击任意按键后退出程序。代码中几个函数的使用方法将在后续章节中介绍。

示例代码1-1 测试OpenCV的安装

#include "opencv.hpp"

int main()

{

cv::Mat img;

img = cv::imread("E:/images/Lena.jpg", 1);

if (img.empty()) return -1;

cv::imshow("Lena", img);

cv::waitKey();

cv::destroyWindow("Lena");

return 0;

｝

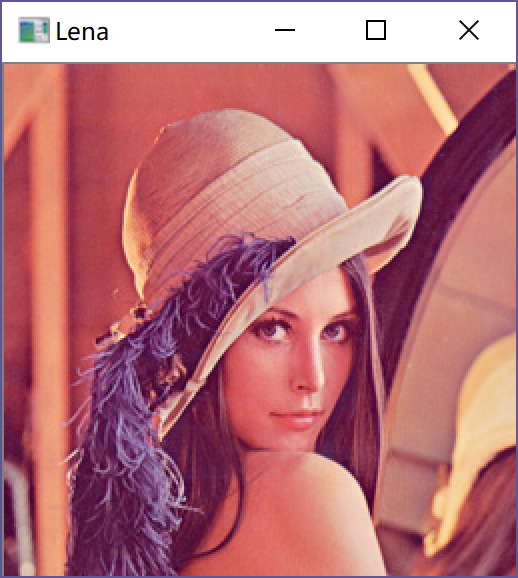


图1-7 测试代码显示图像

从示例代码1-1中可以看出，所有的OpenCV函数都在命名空间cv中，因此需要在每个调用的函数名前面加上cv::，明确告诉编译器函数来自OpenCV函数库。当然，也可以在main函数之前声明使用该命名空间using namespace cv，这样程序将假定引用的函数均来自该命名空间，就不需要为每个函数都添加cv::前缀了，这将在第2章中详细说明。为了提高代码的可读性，建议在编写代码时，在每个OpenCV函数前都使用cv::前缀。

通过示例代码1-1可以发现，需要设置许多属性才能让OpenCV正确工作，下面介绍如何使用配置属性文件的方式简化配置步骤。

单击“视图”→“其他窗口”→“属性管理器”选项，如图1-8所示，右键单击“Debug | x64”选项，在弹出的右键快捷菜单中单击“添加新项目属性表”选项，打开“属性管理器”窗口，将配置另存为属性表文件，并将该文件改名为opencv.props。

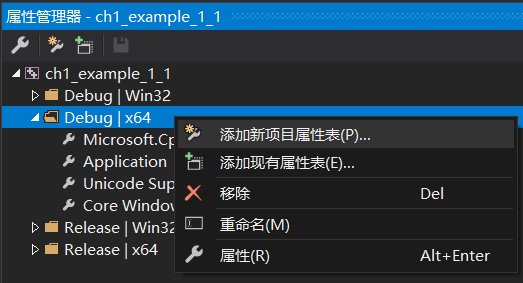
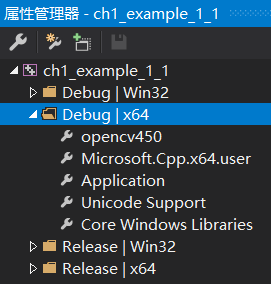
 

图1-8 “属性管理器”窗口

双击“opencv450”属性标签，弹出属性设置页面，根据刚才的配置顺序将opencv450属性重新配置一遍。保存opencv450属性，如图1-9所示。这样在工程目录下即可看到opencv450.props属性配置文件了，后面再新建工程时即可使用该配置文件，简化新建工程时的配置工作。

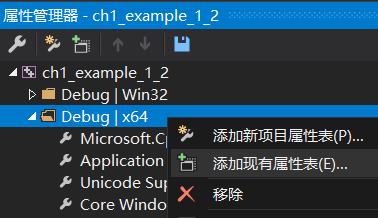
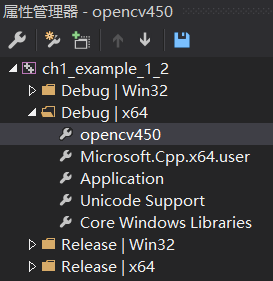
 

图1-9 保存opencv450属性

在新建的工程ch1\_example\_1\_2中，打开“属性管理器”窗口，单击“Debug | x64”→添加新项目属性表选项，将opencv450.props文件添加到项目属性即可，如图1-10所示。

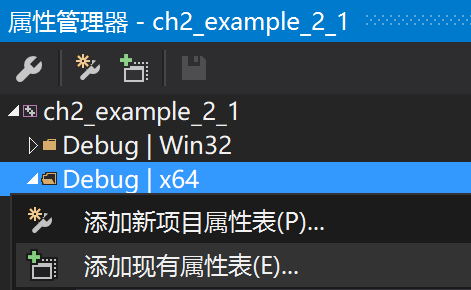
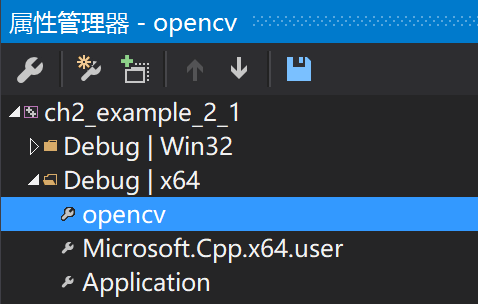
 

图1-10 添加OpenCV属性表

1.4.2 在Linux系统中安装和配置OpenCV

因为Linux系统发布的版本较多，所以没有针对Linux系统预编译好的OpenCV。大多数情况下，Linux系统都包含OpenCV，如果你的Linux系统版本没有提供OpenCV，则需要自己用源代码编译，即到OpenCV官方网站下载对应版本的源代码。

想要编译OpenCV库与测试程序，需要GTK + 2.x或更高版本的GTK（包括头文件）、gcc、Cmake和libtbb，以及可选的zlib、libpng、libjpeg、libtiff和带开发文件的libjasper（包的名称末尾带有-dev）。此外，需要安装Python 2.7或更高版本的Python（包括头文件），同时还需要NumPy、libavcodec和ffmpeg的libavcodec和libav \*库（包括头文件）。

下面以Ubuntu 16.04系统为例，介绍如何在Linux上安装和配置OpenCV。

（1）安装必要的依赖包

使用包管理器运行如下代码，安装必要的依赖包：

$ sudo apt-get -y install libopencv-dev build-essential cmake

libdc1394-22 libdc1394-22-dev libjpeg-dev libpng12-dev libtiff4-dev

libjasper-dev libavcodec-dev libavformat-dev libswscale-dev libxine-dev

libgstreamer0.10-dev libgstreamer-plugins-base0.10-dev libv4l-dev libtbbdev

libqt4-dev libmp3lame-dev libopencore-amrnb-dev libopencore-amrwb-dev

libtheora-dev libvorbis-dev libxvidcore-dev x264 v4l-utils

在安装完依赖包后，就可以下载、编译安装OpenCV了。

2）安装OpenCV

运行如下代码安装OpenCV：

$ wget "https://github.com/Itseez/opencv/archive/4.3.0.zip" -O opencv.zip

$ wget "https://github.com/Itseez/opencv\_contrib/archive/4.5.0.zip" -O

opencv\_contrib.zip

$ unzip opencv.zip -d .

$ unzip opencv\_contrib.zip -d .

$ cd opencv-4.5.0

$ mkdir build

$ cd build

$ cmake -D CMAKE\_BUILD\_TYPE=RELEASE -D CMAKE\_INSTALL\_PREFIX=/full/path/

to/opencv-4.5.0/build -D INSTALL\_C\_EXAMPLES=ON -D BUILD\_EXAMPLES=ON -D

OPENCV\_EXTRA\_MODULES\_PATH=/full/path/to/opencv\_contrib-4.5.0/modules ../

$ make -j4

$ sudo make install

把pkg-config 文件下名为opencv.pc的文件复制到/usr/local/lib/pkgconfig中，并改名为opencv3.pc：

$ cp /full/path/to/opencv-4.5.0/build/lib/pkgconfig/opencv.pc /usr/local/

lib/pkgconfig/opencv3.pc

现在就可以在命令行编译OpenCV程序了。即便本机已经安装了OpenCV2.4.x，也不会有任何冲突。使用下面的代码可检查安装是否正确：

$ cd /full/path/to/opencv-4.5.0/samples/cpp

$ g++ -ggdb 'pkg-config --cflags --libs opencv3' opencv\_version.cpp -o /

tmp/opencv\_version && /tmp/opencv\_version

如果屏幕上打印出Welcome to OpenCV 4.5.0，则表示安装成功。

1.4.3 在Mac OS X系统中安装配置OpenCV

因为在Mac OS X中没有预编译的安装包，所以需要下载源代码进行编译。编译过程与在Linux系统中的非常相似，只是Mac OS X系统自带Xcode开发环境，包括了除CMake外几乎所有需要的东西，不需要GTK+、TBB、libjpeg和ffmpeg，等等。

* 在默认情况下，使用Cocoa代替GTK +；
* 在默认情况下，使用QTKit而不是ffmpeg；
* 使用Grand Dispatch Central（GDC）代替TBB和OpenMP

安装步骤与在Linux系统中的完全相同。传递-G Xcode选项，让CMake为OpenCV（和你的应用程序）生成一个Xcode项目，以便在Xcode中方便地构建和调试代码。