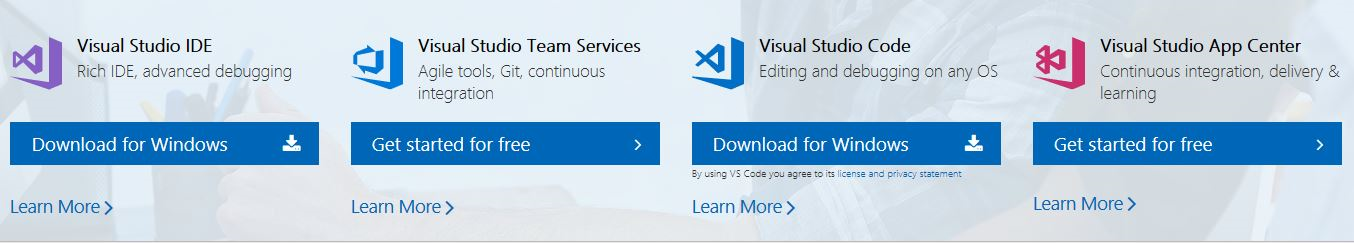
由于最近需要使用OpenCV，本人需要在自己的PC上使用OpenCV，因此最近一直在研究如何使用Visual Studio编译OpenCV源代码并搭建开发环境，折腾了很长时间，查阅了很多相关资料，终于成功。本文记录一下安装的过程以及遇到的问题，仅供参考。开篇吐槽一下博客园的博文【自动保存和恢复】功能，说白了就是可以自动保存，但是几乎不能自动恢复，不要问我为什么这么说，呵呵。

本PC环境的详细配置：Windows 10 16299.192 64-bit，Visual Studio Enterprise 2017 15.5.2，OpenCV-3.4.0-vc14\_vc15，CMake 3.10.1，CUDA 9.0

开始正式介绍安装步骤：

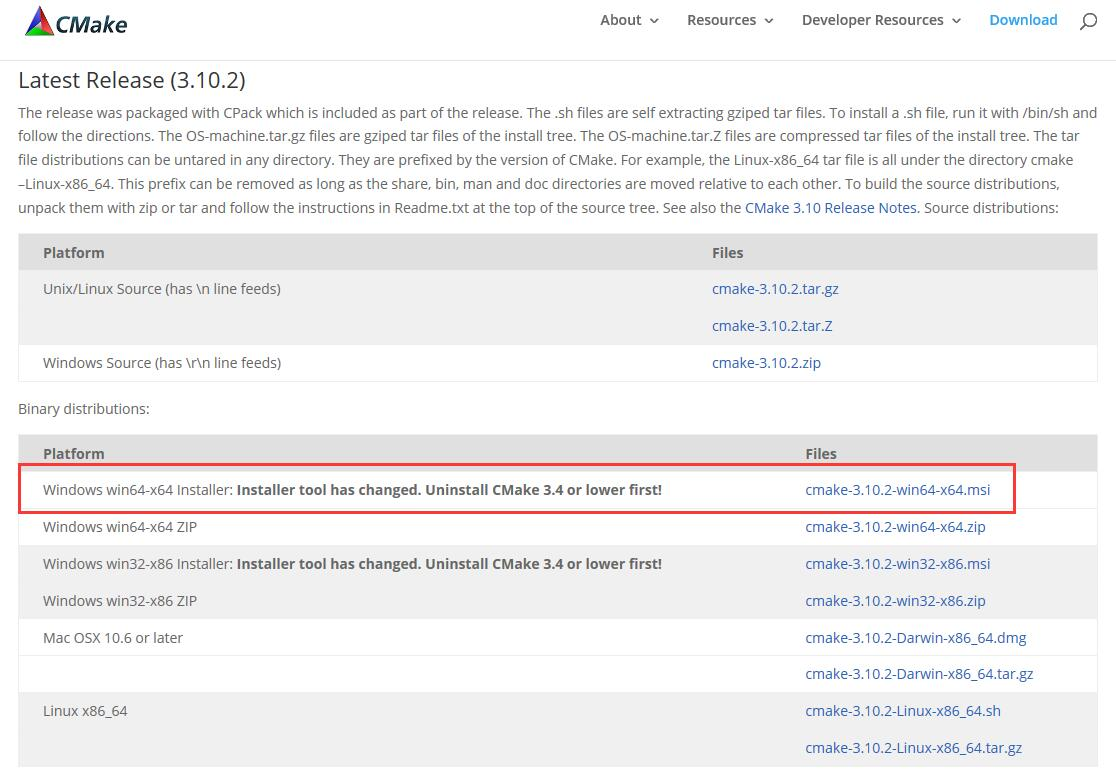
**（1）下载并安装Visual Studio 2017**。由于本PC上安装的是最新的Visual Studio 2017企业版，因此本人主要研究如何在最新版的VS上编译并部署opencv开发环境。如果各位搞不到Visual Studio 2017企业版的激活码，那么使用社区版也是没有问题的。首先到官方网站下载Visual Studio 2017并安装。在Visual Studio Installer中，勾选和C++开发相关的所有组建，并且确保安装过程顺利无误，这一点非常重要，因为必须确保VC++编译环境正确安装。这里提醒一下，注意VS2017的版本号为15，因此对应的VC++编译环境的版本号为vc15，这个很重要，将决定opencv版本的安装。



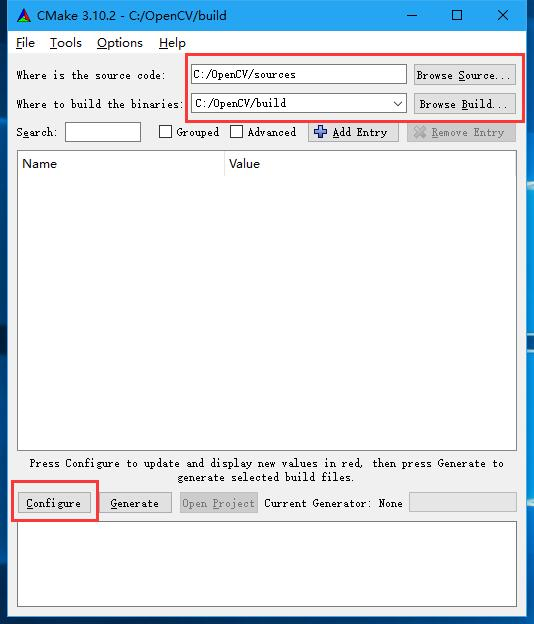
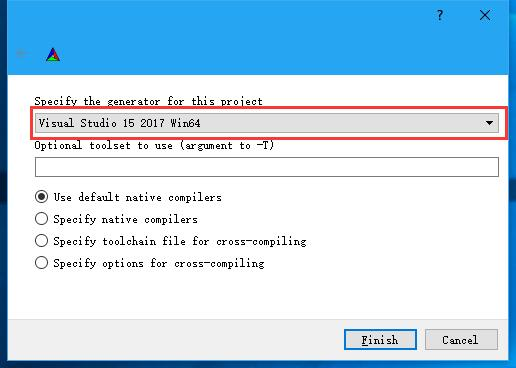
（2）**下载并安装OpenCV**。从官网下载OpenCV，本人使用的是最新版OpenCV-3.4.0-vc14\_vc15，请注意，这里要下载支持vc15的版本，才能很好的兼容Visual Studio 2017，本人曾看一些博客写到用vc14（对应Visual Studio 2015）的版本在Visual Studio 2017中使用，亲测，毛病有不少。安装过程没什么太多好说，比较简单。为了求稳，安装目录设为C:/OpenCV，目录中全英文且不包含任何空格符。PS：这个版本是在12月23日发布的，圣诞礼物哟。



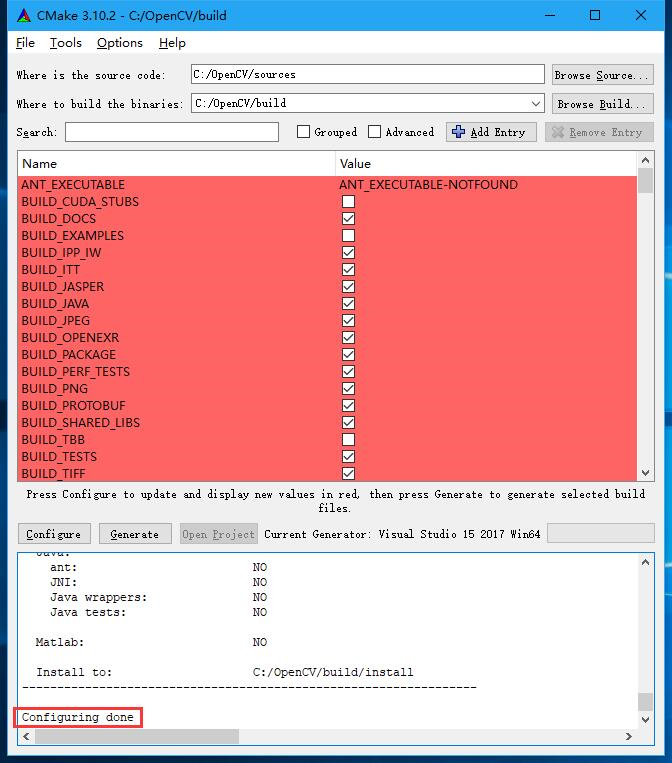
**（3）下载并安装CMake**。从CMake官网上下载最新版的CMake。因为本人使用的环境是比较新的，因此需要使用新的CMake来保证兼容性。



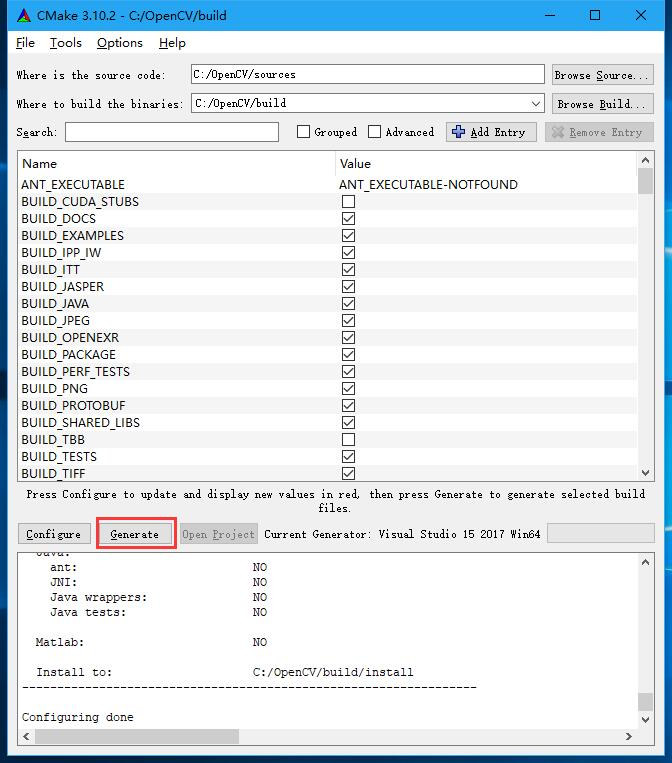
**（4）使用CMake创建OpenCV.sln**。为了实现OpenCV的本地编译，首先要使用CMake创建OpenCV.sln，然后再使用VS2017进行编译，生成库文件。首先启动CMake，选择OpenCV的源代码目录和build目录，然后点击【Configure】,CMake开始执行配置，开始前，要选择正确的编译器，本PC上安装的是Visual Studio 2017 15.5.2，因此选择Visual Studio 15 2017 Win64.此时点击【Finish】，配置过程开始。只要OpenCV和CMake下载安装不出问题，一般来说这个过程是不会报错的，如果出错导致配置失败，则需要查阅日志文件，解决问题。这里说明一点，在Configure过程中，需要联网自动下载三个文件，可能会由于网络的问题导致下载不下来，如果由于该问题导致配置失败，可以通过重新配置的方法解决该问题，实在不行，可以去网上找这三个文件，手动下载。

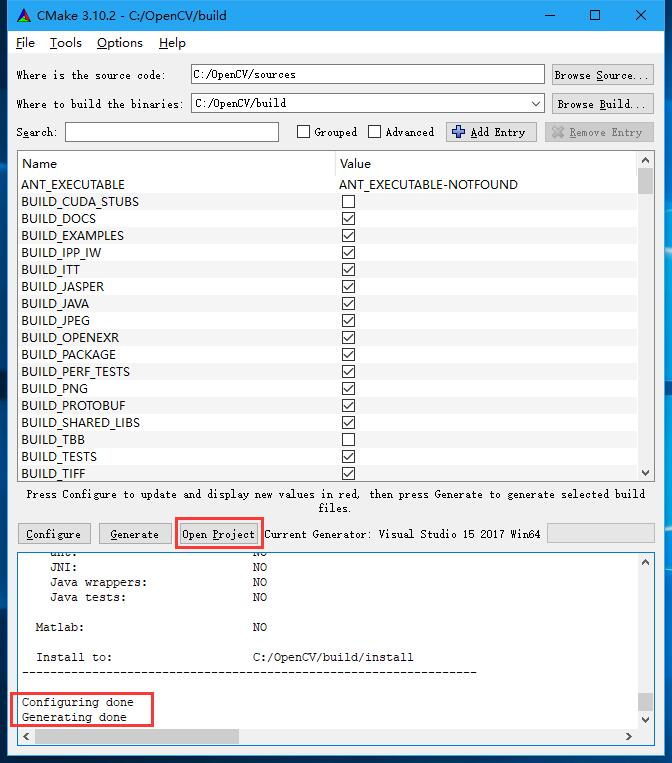
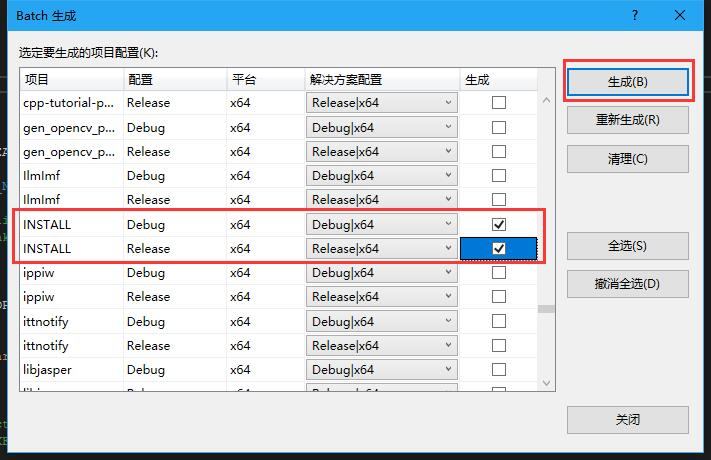
在配置过程结束后，如果成功，则会在日志框里显示【Configuring done】，中间的配置框会变红，显示配置项。在这里，要手动勾选两项【BUILD\_EXAMPLES】和【BUILD\_opencv\_world】，然后再点击【Configure】，再次执行配置。不出意外的话，这个过程会很快完成，配置框变为白色。该步骤完成。本人在这里遇到了一个巨大的坑：本人的PC上安装了CUDA 9.0，不知道是没安装好还是兼容性问题，在成功生成OpenCV工程文件后，使用VS2017编译源代码，总是失败，折腾了两天时间。经过查阅编译日志（这里忘了截图），发现是CUDA编译器的问题，经过摸索，发现将BUILD\_CUDA相关的项目勾选去掉，然后重新生成sln，解决了问题。后来我重装了系统，没有安装CUDA，则正常安装即可，可忽略这个步骤。

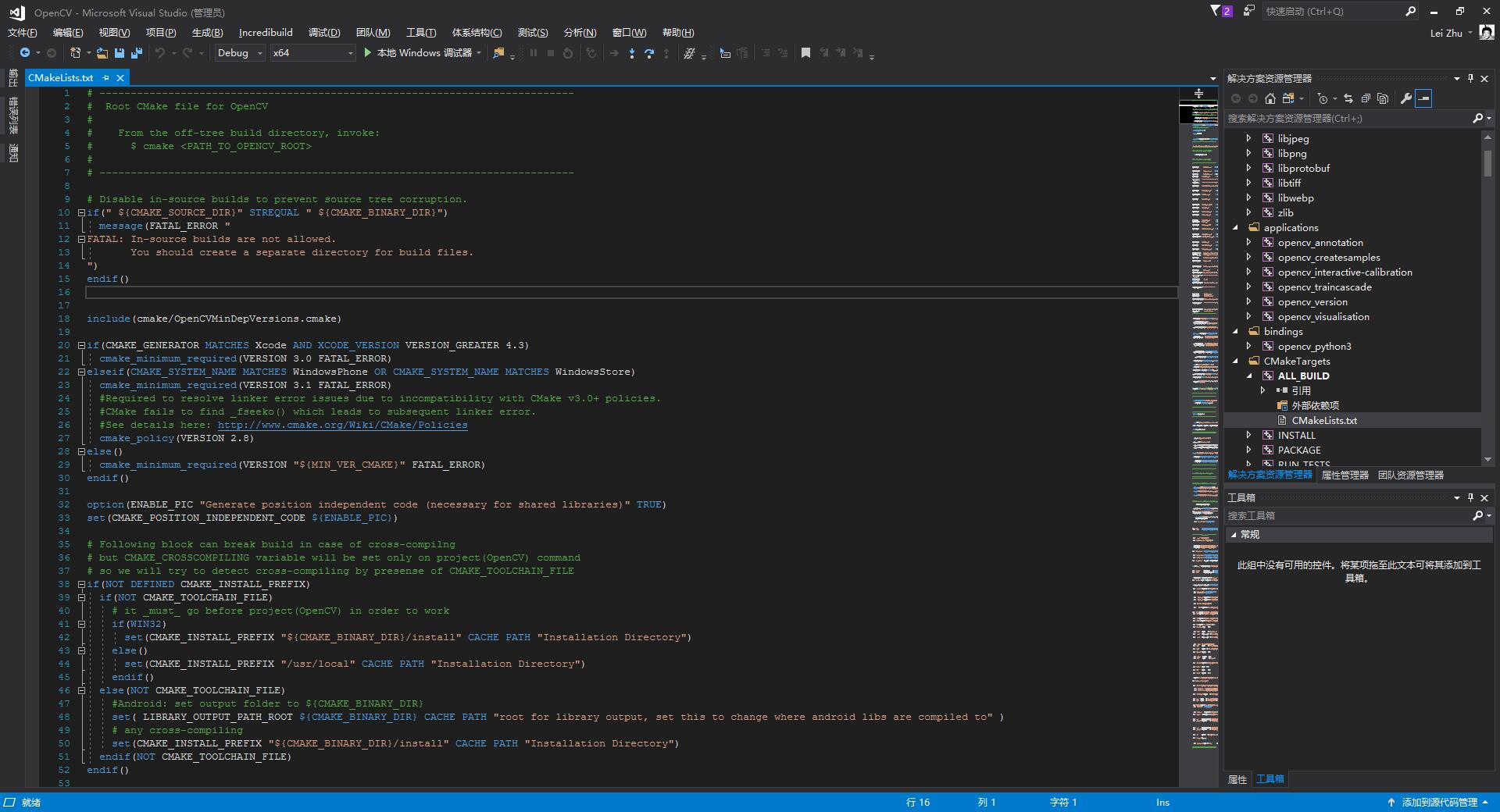


**（5）生成OpenCV.sln**。配置完成后，点击【Generate】，生成VS工程文件。这个过程一般不需要太多时间，只要之前的过程没出错，这里一般不会有问题。



**（6）使用VS2017编译OpenCV源代码**。点击【Open Project】按钮，CMake会启动VS2017并加载OpenCV.sln。加载过程可能要一点时间（相对于一般小规模项目而言），需要耐心等待。加载完成后，点击【生成】，选择【批生成】，然后只勾选INSTALL那两项，其他项目不勾选，然后开始生成。这个过程比较久，需要耐心等待，本PC上跑了50分钟。

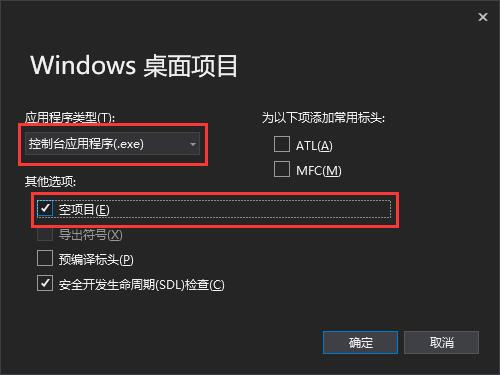
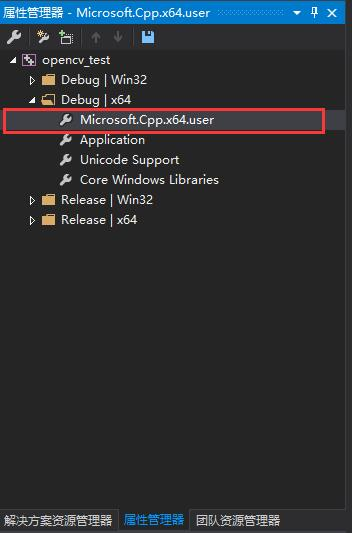
  

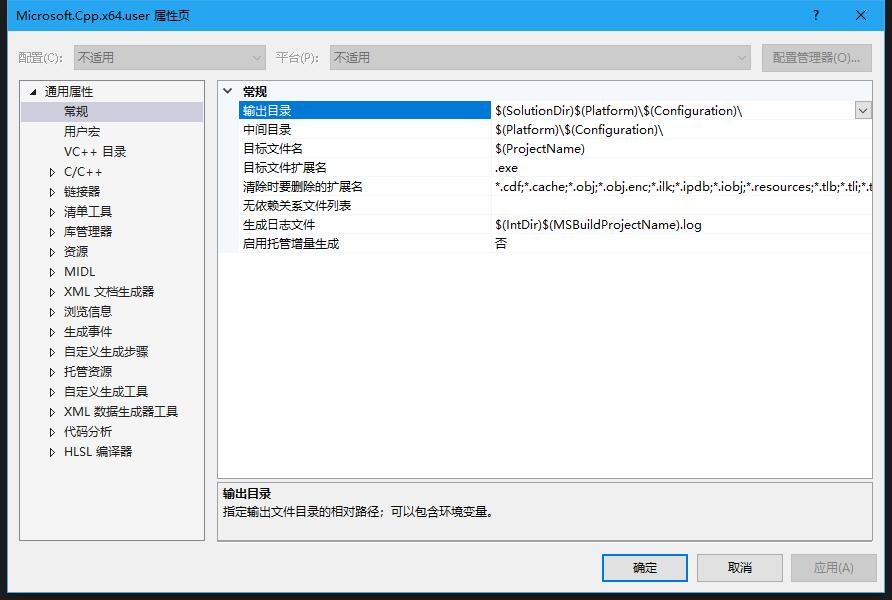


Build成功后，在C:\OpenCV\build\install\x64\vc15\bin目录下会生成相应的动态库。如果有失败的，要查阅输出日志，具体问题具体解决。到此，本地编译OpenCV的工作就全部做完了。

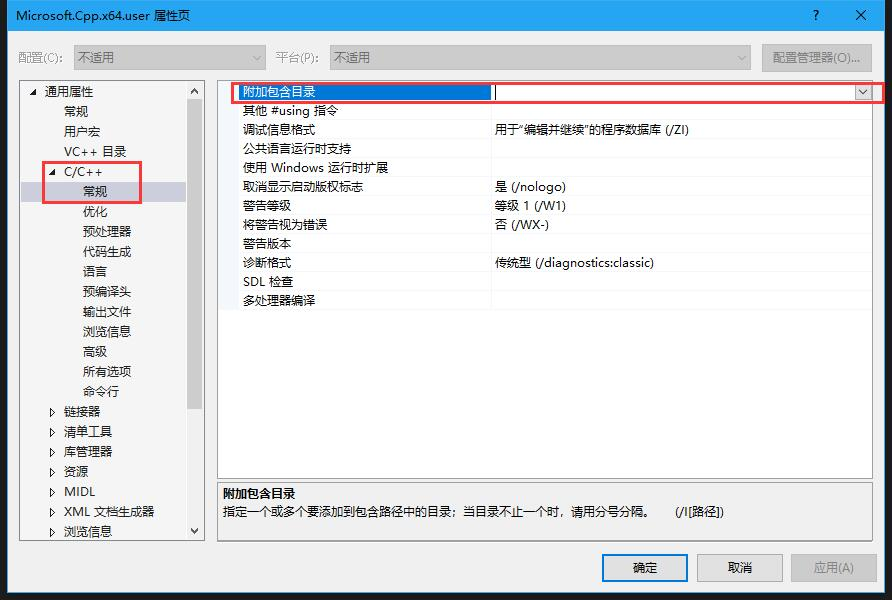
**（7）配置OpenCV环境变量**。右击【此电脑 -> 属性 -> 高级系统设置 -> 环境变量 -> 系统变量 -> 编辑Path -> 新建】，添加环境变量C:\OpenCV\build\install\x64\vc15\bin。这个步骤不难，网上有很多教程。

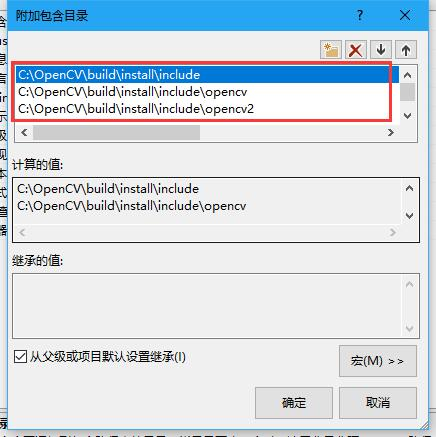
**（8）新建工程并测试**。在VS2017中新建一个空的C++项目。在【属性管理器】中选中【Debug | x64->Microsoft.Cpp.x64.user】并右击弹出菜单，选择【属性】。

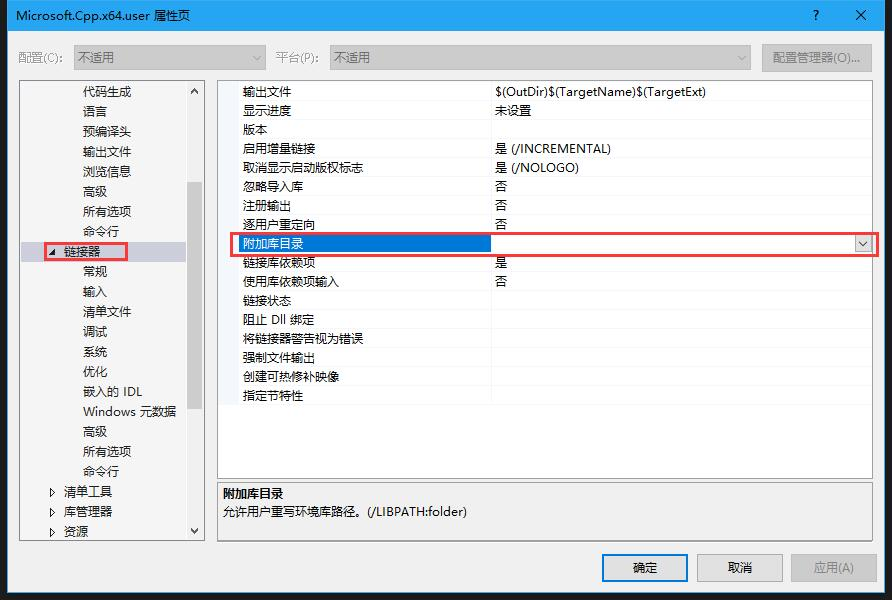


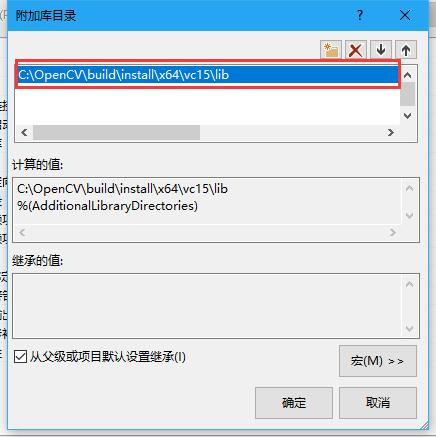
选择【C/C++ -> 常规 -> 附加包含目录】，添加如下三个目录：



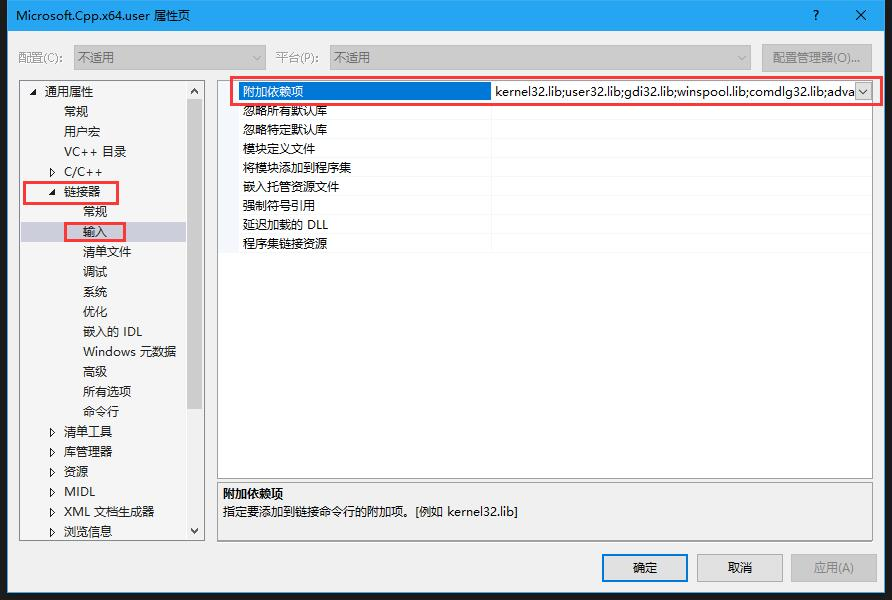


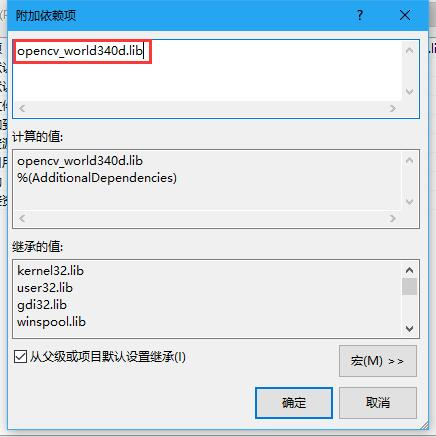
再选择【链接器 -> 附加库目录】，添加如下目录：





再选择【链接器 -> 输入 -> 附加依赖项】，添加如下库文件名：





点击【确定】，【应用】。

创建一个cpp文件，并添加如下测试代码：

[复制代码](javascript:void(0);)

1 #include <opencv2\opencv.hpp>

2

3 using namespace cv;

4

5 int main(int argc, char\*\* argv)

6 {

7 Mat img = imread("C:\\Users\\lzhuc\\source\\repos\\opencv\_test\\x64\\Debug\\lena.jpg");

8 imshow("img", img);

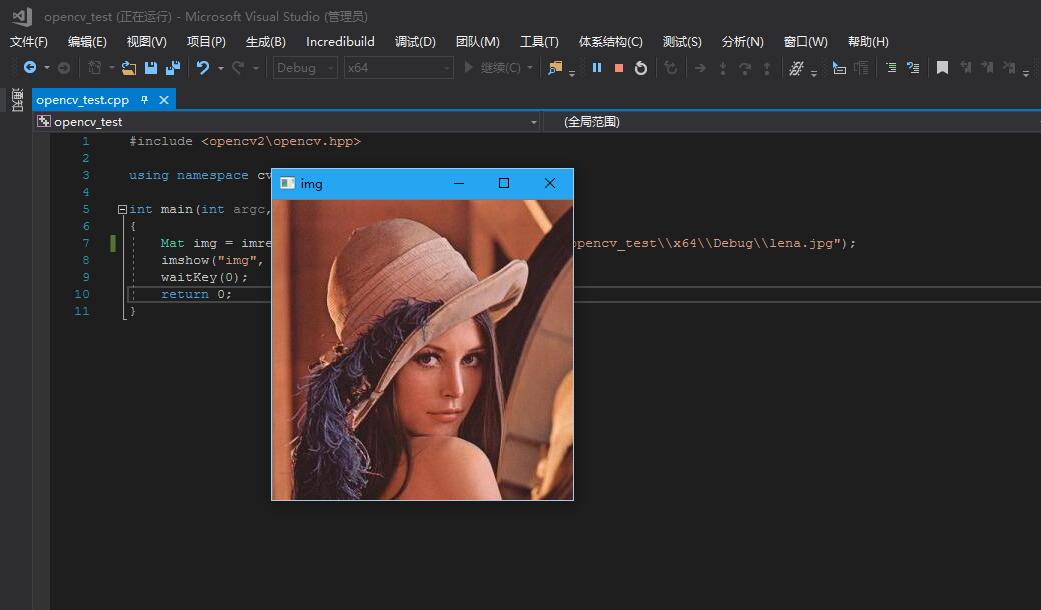
9 waitKey(0);

10 return 0;

11 }

[复制代码](javascript:void(0);)

编译，运行，效果如下：



本人测试发现，如果配好了环境直接运行，会报错，发现img为空，是个0x0的矩阵，重启一下系统，就可以了。

参考资料：

http://blog.csdn.net/rrrfff/article/details/76796261

http://blog.csdn.net/u013832707/article/details/53127710

http://blog.csdn.net/poem\_qianmo/article/details/21974023

http://blog.csdn.net/paulhappy/article/details/52107121