Using OpenCV in Microsoft Visual C++

This is a short tutorial on how to compile and run a simple OpenCV program using Microsoft Visual C++ 2015. You should already know how to create a simple “Hello world” program in Microsoft Visual C++. If not, please go through the tutorial called “Creating Your First Program in Microsoft Visual C++”.

Setting up Visual C++

Start up Microsoft Visual Studio. You can open the project that you created in the previous tutorial, or start a new project. If you start a new project, follow the same steps that we did in the previous tutorial, to create a project and add a “main.cpp” source file.

Step 1 – Location of include files

We want to change the “solution platform” to be 64 bits. In the drop down menu shown below, change “x86” to “x64.

![Visual Studio Platform Change](image)

Step 2 – Location of include files

Tell the compiler where to find the include files. Go to Project->Properties. In the pop-up window (shown in the figure below), select “Configuration Properties” -> “C/C++” -> “General”. Then enter the location of the include files into “Additional Include Directories”. On the computers in Brown, it is

- `C:\sw\opencv\install\include`
Step 3 – Location of “lib” files

Tell the compiler where to find libraries at linking time. Go to the “Linker” section on the property page. As shown below, go to “General”. Enter the location of the “.lib” files in the slot called “Additional Library Directories”. On the computers in Brown, it is

- C:\sw\opencv\install\x64\vc14\lib

Note – if you click on this arrow, it brings up a popup window that allows you to browse to find the directories you want.
Step 4 – Names of “lib” files

Go to “Input” under “Linker” and enter in “Additional Dependencies”, all the library names, as shown below.
Here is the complete list (the d is for “debug” builds). You can copy and paste the text below. The list is also in a file called “list_of_libs.txt” in the main OpenCV folder.

opencv_aruco310d.lib
opencv_bgsegm310d.lib
opencv_bioinspired310d.lib
opencv_calib3d310d.lib
opencv_ccalib310d.lib
opencv_core310d.lib
opencv_datasets310d.lib
opencv_dnn310d.lib
opencv_dpm310d.lib
opencv_face310d.lib
opencv_features2d310d.lib
opencv_flann310d.lib
opencv_fuzzy310d.lib
opencv_highgui310d.lib
opencv_imgcodecs310d.lib
opencv_imgproc310d.lib
opencv_line_descriptor310d.lib
opencv_ml310d.lib
opencv_objdetect310d.lib
opencv_optflow310d.lib
opencv_photo310d.lib
opencv_plot310d.lib
opencv_reg310d.lib
opencv_rgbd310d.lib
opencv_saliency310d.lib
opencv_shape310d.lib
opencv_stereo310d.lib
opencv_stitching310d.lib
opencv_structured_light310d.lib
opencv_superres310d.lib
opencv_surface_matching310d.lib
opencv_text310d.lib
opencv_tracking310d.lib
opencv_ts310d.lib
opencv_video310d.lib
opencv_videoio310d.lib
opencv_videostab310d.lib
opencv_xfeatures2d310d.lib
opencv_ximgproc310d.lib
opencv_xobjdetect310d.lib
opencv_xphoto310d.lib
opencv_xphoto310d.lib

Then click “OK” to get out of the property pages.
Step 5 – Location of “dll” libraries

Finally, you need to tell the computer where the “dll” libraries are, so that it can grab them at run time. There are several ways to do this. If you have administrator privileges on your computer, just add the directory containing the “dll” files to the PATH environment variable (this is done by going to the Control Panel and selecting “System”). The path to the “dll” libraries on my computer is

C:\OpenCV-3.0.0\install\x64\vc14\bin

The PATH is already set on the computers in Brown to

C:\sw\opencv\install\x64\vc14\bin

If the path name is not set, and you don’t have administrator privileges, you can always just copy the “dll” libraries that you need to your runtime directory. For example if you called your project “ConsoleApplication1”, then you should copy the “dll” files to “ConsoleApplication1\x64\Debug”.
Creating a Simple OpenCV Program

In the main window, enter the following code for `main.cpp`. You will have to change the path name to the image file. (Note – I have found that if you copy and paste this code from a pdf file, you may find that there are some spurious invisible characters that are included. I think a safe way is to first copy the text to a blank MS Word document, then copy that and paste it into Visual C++.)

```cpp
/* First OpenCV program. */
#include <iostream>
#include <opencv2/opencv.hpp>

int main(int argc, char* argv[]) {
    printf("Hello world\n");

    // Read an image.
    cv::Mat image = cv::imread("C:/sw/opencv/sources/opencv/samples/data/lena.jpg");
    if (image.empty()) {
        std::cout << "Hey! Can't read the image!" << std::endl;
        system("PAUSE");
        return EXIT_FAILURE;
    }

    // Create image window named "My Image". (You actually don't have to do this step, but this command allows you to set properties of the window, such as its location, or whether you can resize it.)
    cv::namedWindow("My Image");

    // Show the image in the window
    cv::imshow("My Image", image);

    // Wait for 5000 ms (0 means wait until a keypress)
    cv::waitKey(5000);
    return EXIT_SUCCESS;
}
```

Choose Debug->Start Debugging to compile and run the program. If there are no errors, the program should display an image.

Note - your program may compile ok, but then crash at run-time. This could happen, for example, if the system can’t find a “dll” library file. In such a case, you may get a cryptic popup window like the one below.
To get out of this mode, hit the “Break” button. Then go to the “Debug” menu and select “Stop Debugging”.

**Additional Notes**

**PDB Files**

You may get warning messages when you run the program, that look like this:

'C:\Windows\SysWOW64\ntdll.dll', Cannot find or open the PDB file  
'C:\Windows\SysWOW64\kernel32.dll', Cannot find or open the PDB file  
'C:\Windows\SysWOW64\kernelbase.dll', Cannot find or open the PDB file

A program database (PDB) file holds debugging and project state information. It is needed if you use the debugger to step through your program. You can ignore the messages if you don’t plan to use the debugger. Or, an easy way to fix this problem is to just go to Tools -> Options -> Debugging -> Symbols, and then check the box for “Microsoft Symbol Servers”.

**Additional Projects**

It is tedious to go through all the above steps every time you want to create a new OpenCV project. There is a way to automate this, using a feature of Visual Studio called “property sheets”. However, I have found that the easiest thing to do is to simply copy an existing project (i.e., the whole directory), and just edit the source files.