opencv ubuntu setup, debian and configuration of library

/* Installing OpenCV 2.3 in Ubuntu.I had a lot of problems initially to install and port the OpenCV library to Python. But after reading a lot of blogs I finally found a solution which I have posted below. */
1)Install all pre requisites
[crayon-6684c3e806eb6701437302/]

2)Install the python development headers
[crayon-6684c3e806ebf797209530-i/]

3)Download the source code:
http://sourceforge.net/projects/opencvlibrary/files/opencv-unix/2.3/

4)Go to the directory where OpenCV is downloaded via the terminal only and then un zip the package:
(note:- It is recommended that you move the downloaded OpenCV package to the home/ directory
[crayon-6684c3e806ec1886778427/]
5)Now make a new directory called build and go in to it
[crayon-6684c3e806ec4945400123-i/]

6)Run Cmake
[crayon-6684c3e806ec6294944754/]
7)Now make

[crayon-6684c3e806ec8653339959-i/]

8) Make it permanent

[crayon-6684c3e806eca715908786-i/]

9)Configuring OpenCV for using shared libraries:

[crayon-6684c3e806ecd289839181-i/]

Add the following line at the end of the file (it may be an empty file, that is ok) and then save it:
[crayon-6684c3e806ecf268359315/]
Close the file and run the following command to configure the library:

[crayon-6684c3e806ed1221506124-i/]

[crayon-6684c3e806ed3904627755-i/]
10)Open your .bashrc file and add the following:
[crayon-6684c3e806ed6941979493/]
[crayon-6684c3e806ed8343765336/]
Save and close the file

11) Reboot the system

http://ubuntuone.com/0qwnXh8VoufDMHGiu5kLIK

Code::Blocks is an GPL based and cross-platform IDE. This is the tutorials using Code::Blocks with OpenCV.

There are 2 different ways to configure OpenCV (shown below), but first you should create a new project:

Create a simple console project.

We can use project wizard to create a simple console project. Here is the steps

Give this project name of “test_opencv”

Then copy some sample code to main.cpp, such as the contents of[crayon-6684c3e806eda724712996-i/]

Configuring Code::Blocks for OpenCV v2.4

Now you need to configure the compiler to find the OpenCV header files and libraries. There are 2 different ways you can configure Code::Blocks for OpenCV v2.4. OpenCV was made of just 4 libraries originally (until v2.1), but now there are many more library files, so you are highly recommended to use the tool "pkg-config" (as mentioned on **CompileOpenCVUsingLinux**), but if you are using MinGW on Windows then you might prefer the manual method.

1. Automatically with the pkg-config tool (easier with Linux or Mac).
2. Manually adding the OpenCV library (easier with MinGW on Windows).

Configuring Code::Blocks for OpenCV using pkg-config

pkg-config is a free command-line tool (available on Windows, Mac and Linux) that should have been automatically setup correctly if you built OpenCV with CMake. Open a command-line and enter **pkg-config opencv –cflags** or **pkg-config opencv –libs**, it will display the compiler and linker arguments to successfully compile your own OpenCV projects without worrying about where OpenCV is installed or worrying about which version you have installed, or how to link to the library in each Operating System, etc. If you get an error that it does not know what pkg-config is, then install pkg-config yourself (Linux or Mac can use the official release, but for Windows (MinGW) you should use the tool pkg-config-liteinstead).

If you were compiling your project on the command line you could type:

[crayon-6684c3e806ede362831715/]
(Note: pkg-config is surrounded by back-tick characters, not quote or apostrophe characters (its usually the same key as the Tilda key ~, next to the 1 and Esc keys).

To setup Code::Blocks to use pkg-config, first you should right-click on your project and open the “Build options …” dialog.

Now you can simply put this into “Linker settings -> Other linker options”:
[crayon-6684c3e806ee5014118739/]
And put this into “Compiler-> Other options”:
[crayon-6684c3e806eeb561404243/]
(Remember to include the back-tick characters, by copy-pasting those lines directly into Code::Blocks).

The beauty of this method is that it should work for Linux, Windows and Mac, and for all versions of OpenCV, whereas the old manual method has different lib filenames for different Operating Systems and different versions of OpenCV.

Troubleshooting

If you have tried the pkg-config method above but Code::Blocks does not find OpenCV headers and libs but you have verified that pkg-config works for OpenCV on a command-line, then Code::Blocks probably doesn’t know where pkg-config is installed. So you should place a link into [crayon-6684c3e806ef3369843894-i/] (Windows) so that it will be found. For Linux or Mac, you can find the path to pkg-config if you type this into a terminal:
[crayon-6684c3e806ef8851095638/]
This might print something like “/opt/local/bin/pkg-config”.

So then type this in a terminal:
[crayon-6684c3e806efe337936139/]
This allows accessing pkg-config from the most common program folder “/usr/bin/” as well as it’s actual location.

Configuring Code::Blocks for OpenCV v2.4 manually

In this tutorial I will be using OpenCV v2.4.2 and Code Blocks v10.05 with GNU compiler (MinGW) on Windows 7. To work on OpenCV with Code Blocks you just need to add some OpenCV libraries and the folder with OpenCV header files. Following are some simple steps that I followed:

1.
 http://sourceforge.net/projects/opencvlibrary/files/opencv-win/,

2. installed OpenCV to “C:\OpenCV”, so adjust the paths if you installed somewhere else)In Code::Blocks, goto the menu “Settings > Compiler and Debugger > Search Directories”. Then goto “Add” and add the directory “C:\OpenCV\include\opencv”. This will let your project find the OpenCV header include files when compiling (before linking):

4. In the Linker tab, add the directory[crayon-6684c3e806f06452121597-i/]

5. Now click on “Linker Settings”. Add all the .lib files from[crayon-6684c3e806f0d077700626-i/] (many files). This will let you project link to OpenCV libraries:

6. That’s it!! Now you are ready to run your first OpenCV program. I tested a sample program “kutayzorlu” in the folder “C:\OpenCV\samples\cpp\kutayzorlu.cpp”.

Alternative method, for OpenCV 2.2

In order to make OpenCV 2.2 (containing C++ code) work under Windows with Code::Blocks and MinGW, Matthew (mazeus12 on the mailing list) did the following:

As mentioned in "OpenCV-2.2.0-win-README.txt", "OpenCV-2.2.0-

win32-vs2010.exe" does not contain binaries for MinGW so they need to be built from the contents in "OpenCV-2.2.0-win.zip".

Steps to build OpenCV 2.2 with Code::Blocks and MinGW:

1. Install Code::Blocks (10.05) with the (MinGW) C++ compiler option. This should among other install the C++ compiler and mingw32-make to"C:\Program Files\CodeBlocks\MinGW\bin" (I also tried to install the latest MinGW using "mingw-get install gcc g++ mingw32-make" from www.mingw.org but I got an error in extracting some files…)

2. Add "C:\Program Files\CodeBlocks\MinGW\bin" to system PATH (at your own judgment: remove any other paths to MinGW (SomehowDevCpp MinGW paths with probable different versions messed up the build process))

3. Install Cmake (2.8)

4. Extract "OpenCV-2.2.0-win.zip" to "C:\OpenCV-2.2.0-win" (It creates a second folder so the final destination looks like that: "C:\OpenCV-2.2.0-win\OpenCV-2.2.0")

5. Run Cmake (cmake-gui)

6. Set the source code: "C:\OpenCV-2.2.0-win\OpenCV-2.2.0"

7. Set where to build the binaries: e.g. "C:\OpenCV2.2MinGW"

8. Press configure

9. Let Cmake create the new folder

10. Specify the generator: MinGW Makefiles

11. Select “Specify native compilers” and click next

12. For C set: C:/Program Files/CodeBlocks/MinGW/bin/gcc.exe

13. For C++ set: C:/Program Files/CodeBlocks/MinGW/bin/g++.exe

14. Click finish

15. In the configuration screen type in “RELEASE” (or “DEBUG” if you want to build a debug version) for “CMAKE_BUILD_TYPE”. Select BUILD_EXAMPLES if you want (I didn’t change anything else here like “WITH_TBB” or “WITH_QT” etc. I’ll try that when time comes to use TBB or Qt)

16. Click configure again

17. Click generate

18. Close cmake

19. Go to the command prompt and inside the folder "C:\OpenCV2.2MinGW" type "mingw32-make" and hit enter (takes some time)

20. Then type "mingw32-make install" and hit enter again

21. Open Code::Blocks and create a new C++ project (Configuration similar to the guide: CodeBlocks).

22. In menu: “Project/Build options/Linker settings/Link libraries”
add "C:\OpenCV2.2MinGW\lib\libopencv_calib3d220.dll.a" and all the other *.dll.a files in this folder

23. In menu: “Project/Build options/Search directories/Compiler”
add "C:\OpenCV2.2MinGW\include" (includes in a new program schould look like this: “#include , #include , #include etc.”)

24. In menu: “Project/Build options/Search directories/Linker” add "C:\OpenCV2.2MinGW\lib" The options in steps 22 – 24 can also be added as global options in menu: Settings/Compiler an Debugger/Global compiler settings/…, so they will apply to any project and opened *.cpp file.

25. If necessary, in menu: Settings/Compiler an Debugger/Global compiler settings/Toolchain executables”

specify"C:\Program Files\CodeBlocks\MinGW" for the compiler’s installation directory.

26. Open a sample file (or import it into the project), e.g. "C:\OpenCV2.2MinGW\samples\cpp\dft.cpp" and built it. (If the *.cpp files do not exist in this folder, you can find them in the initial folder where you extracted "OpenCV-2.2.0-win.zip" i.e. "C:\OpenCV-2.2.0-win\OpenCV-2.2.0\samples\cpp")

27. Add "C:\OpenCV2.2MinGW\bin" to the system path

28. Run the program

Configuring Code::Blocks using old method for OpenCV v1.1

Set the include header file path (OpenCV v1.1):

Set the library path (OpenCV v1.1):

Add the libraries (OpenCV v1.1) directive:

Here is another way you can add the include path and lib path in Code::Blocks

First, you need to add a global variable in Codeblocks, see here: You can open this dialog in:
[crayon-6684c3e806f1b390459949/]
Then you can define the [crayon-6684c3e806f23684678725-i/] path, in my system,

fill the include edit bar with(this is where your opencv include path locates) : $(#cv)\OpenCV-2.1.0\include\opencv

fill the lib path with(this is where your opencv libraries

locate) : $(#cv)\opencv_build\lib

Later, in your Opencv project, you can change the build options like below:

Also, you can add the libraries by using these linker options:
[crayon-6684c3e806f29194688655/]