Setting up your OpenCV environment

Instructions:

I. Overall goal and software

In this assignment you have to set up your programming environment for use with OpenCV v2.4.latest (http://www.opencv.org) and Boost v1.56.0 (http://www.boost.org/) and GL/OpenGL/GLUT. While OpenCV and GL/OpenGL/GLUT are mandatory in this course, Boost is not, but it can make your life substantially less difficult.

You are allowed to any OS that supports OpenCV (ie.: Windows, Linux, OS X). However, take note of the fact that we can only test your programs on Windows and Linux. Programming for OS X is at your own risk. In the past this has proven to be not much of an issue however, so it is allowed. **Doing the assignments in something different than C/C++ IS also allowed, but as you will only be given C/C++ framework, you may have to implement more than strictly necessary if you choose another language. The reason we allow this, is that some people have a really hard time understanding the implications of the memory storage model for the Matrix structures of OpenCV, which may lead to a world of hurt. OpenCV and C++ are unforgiving in this regard. So be warned that not properly understanding this (which is NOT part of this course), may lead to unpredictable results and failing programs due to pointer exceptions and memory leaks. I leave it to your own judgement if you think C/C++ is suitable for you. Using a programming language with which you cannot talk to OpenCV is strongly discouraged (ie.: not allowed)!**

We strongly urge you to use a decent IDE, such as Visual Studio or Eclipse. You will be provided with code which has been tested in both VS2012 (on Windows 8) and Eclipse CDT (Luna SR1) (on Linux Mint 17). **It is your own responsibility take get the provided code running in your preferred development environment if you choose to use it!**

If you choose to use C#, Python or Java, you are more on your own as far as reading/writing input or output goes. I assume you have the skills for this. As you can talk to the OpenCV API from these languages, questions regarding the OpenCV API are fine. As far as the OpenGL code goes: you should be fine in C# and Java, for Python I honestly have no idea (you’re a bit on your own here).

Here’s some helpful links OpenGL outside of C/C++:
C#: https://www.youtube.com/watch?v=2KEHrB82Z2M
Python: http://www.labri.fr/perso/nrougier/teaching/opengl/
Java: https://solarianprogrammer.com/2014/12/08/getting-started-jogl-java-opengl-eclipse/
From here this tutorial is about the C/C++ environment. Go ahead and download the latest OpenCV library of the 2.4 branch. Beware that v3.0 is also available, but none of the provided code has been tested for it. The API has changed significantly, parts of provided code may not work due to API incompatibilities. For Boost we assume v1.54 (though newer versions exist and are OK).

The following gives detailed instructions for Windows and Linux on how to get the C++ libraries installed on these systems. They are not perfect, and will most likely not work by copying it directly. Try to understand what’s happening as you setup your environment! Again, if you choose to use C# (EmguCV), Python (Native) or Java (through JNI), you are more on your own, though decent installations tutorials are available at http://docs.opencv.org/2.4/doc/tutorials/introduction/table_of_content_introduction/table_of_content_introduction.html

If any of the instructions are completely wrong or completely unclear, please send me an e-mail that has a subject that starts with: “[INFOMCV] … your subject ...”. Send it to: c.j.vangemeren@uu.nl

Assignment #1 will appear online shortly.

Kind regards and good luck!
Coert

Ia. Installation on Windows

- Download and extract the compiled libraries
- Add the following Environment Variables to your system
  - BOOST_DIR and OPENCV_DIR, pointing to where you extracted the libraries. Follow the steps in the image below to get to your Environment Variables
- Add your own PATH variable as well, point to the correct library directories which contain the OpenCV libraries (and Boost libraries). Beware that you point to the correct versions of the libraries (32bit most likely).

- Create a new “Win32 Console Application” project in Visual Studio (Visual C++)
  - Follow the Application Wizard steps, nothing fancy
  - Go to the new project’s “Properties"
- Make sure it all looks something like this in the “VC++ Directories” (terribly sorry about the layout). Please note that we are only setting up a **Win32 Debug** environment (a **64 bit Release** environment is left as a challenge to the reader). Go to “Configuration Properties” → “VC++ Directories”
- Add this to the “Include Directories”

- Add this to the “Library Directories”
- Add this to the “Source Directories” (optional, again you may want to add more)
- Go to “Configuration Properties” → “Linker” → “Input” and add the libraries named below

If you look closely, you see there are 3 groups: OpenCV (opencv_*), GL (OpenGL32.lib and Glu32.lib) and Boost (boost_*). For OpenCV and Boost you may want to add more libraries than just the ones mentioned here, depending on your requirements.

Also note that the OpenCV and Boost libs have a “d” or “gd” suffix before the version number, this means these are the “Debug” libraries. Your project will not compile correctly if you choose the wrong libs for a given mode (Debug or Release)!

You should now be able to compile C++ programs with OpenCV, OpenGL (and Boost) libraries in Windows.
1b Installation in Linux (Ubuntu / Mint) + Eclipse CDT

In Linux we use CMAKE to build projects that are managed by Eclipse CDT (http://www.eclipse.org/cdt/). You may also choose QTCreator (https://qt-project.org/), though no building instructions will be given (though in combination with CMAKE it is not very difficult to build OpenCV projects in QTCreator).

- Install OpenCV and GCC build tools on your machine using apt
  - `sudo apt-get install build-essential eclipse-cdt cmake libopencv-dev libboost-dev`
  - (more detailed info here: http://docs.opencv.org/doc/tutorials/tutorials.html)

- Create a new CDT C++ project: File → new → C++ project
- Give a project name
- Choose “Empty Project”, Cross GCC
- Go to the project’s Properties (right click on the project in the “Project Explorer”)
- Add CMake to the Builders
  - Make sure your Launch Configuration is setup correctly, as shown here:
As you can see 3 fields are important: **Location** (cmake should be in /usr/bin if installed through apt), **Working Directory** (all through Eclipse Variables) and **Arguments** (this will allow you to build as either Debug or Release from Eclipse).

At “C/C++ Build”, disable “Generate Makefiles automatically”:
Add the “build” and “src” directory to your project:

- Right click the project in the Project Explorer
  - “New” → Folder: “build” (as we set that as our Working Directory in the Launch Configuration)
  - “New” → Source Folder: “src” (this is where the sources will go)
- Add a “CMakeLists.txt” file to the root of your project:
  - “New” → File
  - Insert into the file:

```cmake
# Your CMakeLists.txt

cmake_minimum_required(Versions 2.8.4)
SET(CMAKE_VERBOSE_MAKEFILE OFF)
project(OpenCVTest)
if(CMAKE_COMPILER_IS_GNUC)
  add_definitions(-std=c++11)
endif()
find_package(OpenCV 2.4 COMPONENTS core highgui imgproc calib3d contrib REQUIRED)
find_package(Boost 1.54 COMPONENTS filesystem regex system REQUIRED)
find_package(OpenGL 1 REQUIRED)
find_package(GLUT 3 REQUIRED)

add_executable(${CMAKE_PROJECT_NAME} src/main.cpp)

target_link_libraries (${CMAKE_PROJECT_NAME} ${Boost_LIBRARIES})
target_link_libraries (${CMAKE_PROJECT_NAME} ${OpenCV_LIBS})
target_link_libraries (${CMAKE_PROJECT_NAME} ${OPENGL_LIBRARIES})
target_link_libraries (${CMAKE_PROJECT_NAME} ${GLUT_LIBRARIES})
```

- Note the project name line: `project(OpenCVTest)`, it should be the same as the project name in Eclipse
- Also note that you may want to add more libraries to the OpenCV or Boost COMPONENTS in `find_package` as required

You should now be able to compile C++ programs with OpenCV, OpenGL (and Boost) libraries in Linux.

II. Sample program

This zip file provides files for a sample project that is setup for VS2012 (Windows) or Eclipse (Linux). It will probably not work directly, so check all steps provided in I.

http://www.staff.science.uu.nl/~0323217/INFOMCV/OpenCVTest.zip

- The API Reference for OpenCV is here: http://docs.opencv.org/index.html (be sure to check the OpenCV Tutorials).
- The API Reference for Boost is here: http://www.boost.org/doc/