Overview

This file is an overview of what is on the course web page for the course CS4102A, Introduction to Computer Vision for the winter of 2017. Please look on CuLearn for the latest assignments and lectures. The up-to-date contents of the PowerPoint slides on CuLearn are what you will be responsible for on the midterm and final exam.

Course Notes

If you go to http://service.scs.carleton.ca/course-outline?field_course_term_value=Winter&field_course_year_value=2017 you will see the course outline and the link for the course web page which is http://people.scs.carleton.ca/~roth/comp4102a-17/ This course web page just contains a some software and data files that you should download. As opposed to previous years, in this year I will only keep CuLearn up-to-date, I will not modify the course web page as the term goes on.

Books

The directory Trucco&Verri on this course web page has the scanned chapters of this book that is a reference in the course. This book is no longer in print, but I have decided to use it for a while longer, which is why I have scanned the appropriate chapters. There is a scan for each of the relevant chapters, and another scan for the entire book (in a slightly different format). In CuLearn I will list which sections of each chapter in the book that you should read for a given lecture. There will be pdf versions of the PowerPoint slides for each lecture on CuLearn, and you are responsible for the content of these PowerPoint slides. The book just makes the PowerPoint slides easier to understand, as does attending my lectures. There will be various small changes in the lecture slides as the course goes on but the latest copy of these lectures are always on CuLearn. The lectures themselves will have a version number on the first page so that you can check if a new version has been produced for a particular lecture. I also plan to record the lectures as video files and link to them on CuLearn.

I might also use some material from a new book, which is on the course web page in pdf form as the file SzeliskiBook_20100903_draft.pdf (the web site with for this book is http://szeliski.org/Book/).
This book is a good resource if you have problems understanding my slides or the Trucco&Verri book. In the directory PythonComputerVision on the DVD is the pdf file, and some code for the book Programming Computer Vision with Python. This is useful for those who wish to use Python for the programming parts of the assignments. This book has a good chapter on the Numpy package, which is the way that images are manipulated in Python. The web site for this book is http://programmingcomputervision.com/

As I said the books that I mention above are simply to make it easier for you to understand the slides and lectures. You are not responsible for what is in the books; you are responsible for the contents of the PowerPoint slides in CuLearn.

OpenCV

This is the software that is used in the course and for programming the assignments. It is a complete computer vision library, including all source code, and is written in C/C++. It runs on Windows, Linux and even on Android and IOS operating systems. The web site for opencv is http://opencv.org/ The latest version of OpenCV is 3.2, and you can download OpenCV for different types of operating systems at http://opencv.org/downloads.html OpenCV for Windows, and Linux/Mac are the ones that matter to you, we do not use the Android or IOS versions of OpenCV. For windows you download OpenCV3.0 if you are using Visual Studio 13 or OpenCV 3.2 if you are using Visual Studio 15. For Windows the OpenCV download is an executable that will extract the opencv software into a directory, while for the Linux/MAC the zip file contains the source for opencv software which you must unzip.

Note that all the versions from 3.0 upwards are compatible with each other, and all the versions from 2.13 downwards are also compatible with each other. By compatible I mean that a program written for a 3.0 will run on any 3.X version, and similarly a program written for 2.13 will run on any 2.X version. But 3.X OpenCV programs will not necessary compile for 2.X OpenCV and vice-versa. The assignments will have some source code that you must change, but I will make sure that there are versions of that code that work with both types of OpenCV (3.X or 2.X). OpenCV 3.X does not have an C bindings, only C++ or Python. If you want to use C, then you must use OpenCV 2.X. The documentation for each version of OpenCV can be accessed from http://opencv.org/documentation.html and for each version there are tutorials, along with a reference manual. The tutorials cover almost all aspects of OpenCV. For OpenCV3.0 the tutorials are http://docs.opencv.org/3.0.0/d9/df8/tutorial_root.html The tutorial code is written in C++ and this code is included in the OpenCV installation on your computer. There is also support for Python in OpenCV, and the Python tutorials for OpenCV are in http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_tutorials.html along with more information on using Python with OpenCV in http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_setup/py_intro/py_intro.html

Almost all of the Python tutorials will work on Python 2.X, even though the documentation says that they are written for Python 3.X. My Python installation instructions below are for Python 2.7 as is the source code for the assignments. You can use whatever version of Python that you like, but version 2.7 is the easiest.

Once you install OpenCV on your computer there are other examples in both C++ and Python besides these tutorials in the sources/samples directory. When installing OpenCV please follow my instructions below and do not use the instructions that are in these tutorials or anywhere else on the web. If you have problems installing OpenCV please send me e-mail or come to my office during consulting hours.

Installing OpenCV for Windows – C++ or Linux

On windows the easiest approach is to use Visual Studio to develop C++ OpenCV programs, but as I mentioned you can also use Python. If you are an experienced C/C++ programmer then using Visual Studio is a good idea, but if you have very little C/C++ experience then Python is a good alternative; the choice is yours. For Visual Studio I have included on the web site an example OpenCV project that you can use directly for your programming assignments. If you use this project then you can simply cut/paste your own C++ source code into this project and thereby avoid having to know enough to properly setup a new VisualStudio project on your own. This is because for your convenience my sample projects on the web site have already been setup to use the appropriate OpenCV directories during compilation and linking. Therefore if you follow my instructions below once you install Visual Studio Professional 2013/2015 along with OpenCV 3.0/3.2 then you should be able to directly compile and run the appropriate example project.

As computer science students you have access to various free copies of Visual Studio from Microsoft at https://secure.scs.carleton.ca:4430/scs_authentication/dreamspark-form.php?department=scs Note that after you login and select the appropriate software then you will be sent a key to unlock the software. You can download either and ISO file and burn a DVD, or download a web installer for the software, both work. I suggest programming OpenCV with Visual Studio 13 (the 32 bit version) because it is the easiest version to use for most people. However, I am not sure there is a web installer for Visual Studio 13, so you can use Visual Studio 15 instead, which I am certain has a web installer. So you use either Visual Studio 13 with OpenCV3.0, or Visual Studio 15 with OpenCV3.2. There are two example projects on the web site, one for VisualStudio 13 and another for VisualStudio 15.

However, you can use any development environment that you prefer (including other versions of Visual Studio), since you will only have to submit source code and program output for your assignments. Please remember that if you use a different development system then you are responsible for setting it up on your machine.

So below is the simplest and easiest way to create and run an OpenCV program using Visual Studio 13.

a) You should first copy the file opencv-3.0.0.exe (or opencv-3. 2.0-vc14.exe) to any directory and then execute it, and choose the location for the extraction as the current directory. Then rename the extracted directory called opencv to OpenCV3.0 (or OpenCV3.2) and move it to the C: drive so that is now C:\OpenCV3.0 (or OpenCV3. 2).
b) Install Visual Studio on your machine, Version 13 is the easiest. Restart your machine.
c) Find out if your machine is a 32 or 64 bit machine. Do this by selecting computer in the startup menu, then right button, then properties. Look in the field called system type; it will say either 64 bit operating system or 32 bit operating system.
d) Now again, startup – computer – right button properties – advanced system settings – environment variables – path variable edit. On Windows 10 - file explorer - this PC - right button advanced system settings – environment variables – path variable edit.
e) At the end of this path variable add either "C:\OpenCV3.0\build\x64\vc12\bin" (or "C:\OpenCV3.2\build\x64\vc14\bin" for a 64 bit machine. For a 32 bit machine this will be "C:\OpenCV3.0\build\x86\vc12\bin" or "C:\OpenCV3. 2\build\x86\vc14\bin". Note that it says vc12 (vc14), which really means Visual Studio 13(15) for reasons known only to Microsoft. Restart your system.
f) Copy the folder OpenCVExampleVC13-3.0 or OpenCVExampleVC15-3.2 onto your machine.

g) Go to that directory and open the file OpenCVExample of type MicroSoft Visual Studio Solution. This will start Visual Studio and open this project.
h) In the top select either x64 or w32 depending on your type of machine and also choose release, not debug. You should not need to set up the x64 configuration mode, but if necessary this can is done by following http://msdn.microsoft.com/en-us/library/9yb4317s.aspx
i) Now select build, rebuild all followed by Start without Debugging. It should also work with if you select the debug configuration and rebuild, you should try both debug and release mode.

When installing Visual Studio make sure to also install the appropriate Visual C++ Tools as described in http://stackoverflow.com/questions/36269673/no-console-application-in-visual-studio-2015

The folder OpenCVExampleVC13-3.0 or OpenCVExampleVC13-3.2 contains an example project that has the appropriate settings to use the version of OpenCV you have installed (assuming you have followed the above instructions). The source code for this project is in the file OpenCVExample.c++ which is a copy of the program display_image.c++ that is in the same directory. This is a simple program that reads in a colour image, converts it to black and white, and draws a 45 degree red line in these two images and then displays them both on the screen. Using this example project you can write and compile your own program that links with OpenCV just by pasting your own source code into the file OpenCVExample++ and then rebuilding the project. Try opening laplace.c++ and edge.c++ (two example C++ programs from OpenCV) in Visual Studio, and then replacing the source of OpenCVExample++ with either of these files. In other words, open the other source files in Visual Studio and then do a copy, and paste replacement into OpenCVExample++. Then select build, rebuild and then run the program. The edge program shows some edges in an image, while the laplace program applies the laplacian operator in real-time to the input from a video camera. Laplace will work only if you have an usb camera, which is the case for most laptops.

If you do not want to use Visual Studio, but instead want to use Python for your assignments the installation process is different. First go to the directory where you have copied the DVD contents.
a) Then you put the extracted OpenCV3.0 directory into the C drive (as in step a above)
b) Run python-2.7.10.exe (choose default location)
c) Run numpy-1.9.1-win32-superpack-python2.7.exe (choose default location)
d) Copy the file C:\OpenCV3.0\build\python\2.7\x86\cv2.pyd into C:\Python27\Lib\site-packages
e) Copy the file C:\Opencv3.0\build\x86\vc12\bin\opencv_ffmpeg300.dll into C:\Python27
f) As was described above in the Visual Studio instructions for steps d) and e) add to the path environment variable but now add "; C:\Python27; C:\Python27\Scripts"
g) Restart your machine and in the start menu execute Python2.7\IDLE(Python GUI)
h) Perform a file open of C:\Opencv3.0\sources\samples\python2\contours.py and run it.
i) If you have a camera on your machine then do a file open of C:\Opencv3.0\sources\samples\python2\edge.py and run it.
j) You can try any other of the example python2 programs in C:\OpenCV3.0\sources\samples\python2.

When using Python you don't need to use Visual Studio at all, so you should use OpenCV3.0, and not OpenCV3.2.

Installing OpenCV on Linux/MAC

There are two different approaches to installing OpenCV on Linux/MAC. The first is to use a package manager and the second is to take the source code and compile it for your Linux system. Since many of you have Macs I will start by discussing the first approach. A Mac is a type of Linux underneath, but to

install OpenCV you need a package manager and you need to work in a terminal window. Packages are the standard way in which software is maintained and updated in the Linux world. On an ordinary Linux system like Ubuntu or the Raspberry PI the package manager comes with the system, but you have to do something extra to make packages work on the Mac. There are two common ways to make Linux packages work on the Mac, they are MacPorts and HomeBrew. In the past I have used MacPorts, but I now prefer to use HomeBrew.

When you use HomeBrew there should already be an OpenCV package that has been compiled for your MAC which you can try to install. Otherwise you must take the OpenCV source code which is in a zip file that you download and then recompile it for your machine. One way to install OpenCV using packages without recompiling on a MAC is described in
http://seeb0h.github.io/howto/howto-install-homebrew-python-opencv-osx-el-capitan/
To do this you first need to install Xcode, and then follow the instructions to install HomeBrew, followed by the installation of the OpenCV3.0 packages. This set of instructions installs OpenCV, along with the python interface to OpenCV. It uses a pre-defined package for OpenCV 3.0 that should be available from HomeBrew. If this works you do not need to compile OpenCV from source. But if this approach does not work, then try the method described in
http://www.pyimagesearch.com/2016/12/19/install-opencv-3-on-macos-with-homebrew-the-easy-way/
Both the above approaches will install the required Opencv 3.0 libraries, along the python interface to OpenCV.

If you have a Linux system which is not a Mac, like the Raspberry PI or any standard Linux system like Ubuntu then you will likely need to compile OpenCV from source. Unzip the file opencv-3.0.0.zip that you downloaded into a local directory, and then follow the instructions in
http://www.pyimagesearch.com/2015/06/22/install-opencv-3-0-and-python-2-7-on-ubuntu/

or http://docs.opencv.org/3.1.0/d7/d9f/tutorial_linux_install.html but try the first set of instructions before you try the second. Remember that you do not need to do a git checkout because you already have unzipped opencv-3.0.0.zip into a local directory which is the one you should use. Then when you are done you should have the opencv libraries installed on your machine. You can ignore the Python parts if you do not want to use Python. If an OpenCV package is available for your particular Linux version (say Ubuntu) then you can try to install this package, but it seems to be just as easy on Linux to compile from source.

Once this has been accomplished then if you use python then you are done, and you can simply run the python examples. But if you do not use Python then you need to do some setup in order to compile and run a C++ program on a Linux/MAC system that links with opencv by following the instructions in
http://docs.opencv.org/3.0.0/db/df5/tutorial_linux_gcc_cmake.html

Here you are using Cmake to create a make file, which is then used to compile the C++ program. I have successfully compiled OpenCV 3.0 on my Ubuntu Linux, and a Raspberry PI, and my Macbook. Both the C++ OpenCV sample programs and the Python sample programs work fine. Once OpenCV was properly installed on these machines then I used the same instructions in
http://docs.opencv.org/3.0.0/db/df5/tutorial_linux_gcc_cmake.html to compile this test program on all these Linux/Mac devices.