Overview

This file is an overview of what is on the course web page for the course CS4102, Introduction to Computer Vision for the winter of 2019. 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 https://service.scs.carleton.ca/sites/default/files/course_outlines/outline_2019.pdf you will see the course outline. The link for the course web page is http://people.scs.carleton.ca/~roth/comp4102a-19 and this page just contains some software and data files that you should download. As opposed to some 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. In other words the course web page contains the software and other material you need for the course, but does not include the lectures.

Books

The directory Trucco&Verri on this course web page has the scanned chapters of this book that is a reference for the course. This book is no longer in print, but I have decided to use it for a while longer, this 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 beside the PowerPoint slides. 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 the lectures. There will be various small changes in the lecture slides as the course goes on but the latest version of these lectures is always on CuLearn. The lectures themselves will have a version number on the first page of the PowerPoint slides so that you can check if a new version has been produced for a given lecture. I also plan to record the lectures as video files and link to them on CuLearn the day after the lecture.

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 a 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. If you are going to use Python to do the assignments then you should have an understanding of the Numpy package. The web site for this book is
http://programmingcomputervision.com/

As I said the books that I mention make it easier for you to understand the slides and lectures. You are not responsible for what is in these 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 is required for programming the assignments. It is a complete computer vision library, including all source code, and is written in C++. OpenCV runs on Windows, Linux and even on Android and IOS operating systems. The web site for OpenCV is http://opencv.org/ and you can download OpenCV for different operating systems at the web site https://opencv.org/releases.html You do not use the Android or IOS versions but you should download either the Windows, or Linux/Mac versions and install OpenCV on your computer. The version of OpenCV that I recommend you use is 3.2.0 but other versions will also work. It is easiest to do the programming assignments in either C++ or in Python but you can use any language that can link with OpenCV. The installation instructions, which I will describe below, are slightly different for these two languages.

For Windows the OpenCV file that you download from the web site is an executable that will extract the OpenCV software into a directory (it is already pre-compiled for Windows), while for the Linux/Mac the zip file you download contains the source code for OpenCV software. In this case the normal procedure is to unzip and then compile OpenCV using Cmake/Make. On Linux/Mac it is possible to use package managers to install OpenCV instead of compiling it from source. I will describe this approach in more detail in the section below where I discuss how to install OpenCV.

Note that all the versions of OpenCV from 3.0 upwards are compatible with each other, and all the versions from 2.0 upwards are also compatible with each other. By compatible I mean that a program written for OpenCV 3.0 will run on any 3.X version, and similarly a program written for OpenCV 2.13 will run on any 2.X version. But 3.X OpenCV programs will likely not compile for 2.X OpenCV and vice-versa. The assignments will have some source code that runs in OpenCV 3.X and not OpenCV2.X, so you should use some 3.X version of OpenCV and program in either C++ or Python. However, if you find some other language that links with OpenCV you can use this language, it does not matter to me if you do not use C++ or Python but in this case you must install OpenCV on your own.

The documentation for OpenCV can be accessed online via the documentation link for each downloadable version of OpenCV at https://opencv.org/releases.html You can even download the complete documentation at https://docs.opencv.org/ to create a local copy of the documentation on your machine. There are also many detailed online tutorials for OpenCV written in C++ at https://docs.opencv.org/3.2.0/d9/df8/tutorial_root.htmlYou can select the version of OpenCV that you are using in the top left part of the screen if it is not version 3.2. Note that some of these tutorials are out of date. For example, you should ignore the installation instructions written there and instead follow my instructions for installing OpenCV.

Note that you will not require a live camera for any of the assignments or even be able to read and write video files. This means there is no need to compile OpenCV in a way that provides these capabilities (in some Linux installations making this work is a bit tricky). There are also Python versions of the C++ tutorials at https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_tutorials.html Along with the tutorials when you install OpenCV you get a number of sample programs (i.e. https://docs.opencv.org/3.2.0/examples.html) in the sources/samples directory. The tutorials, online documentation, and sample programs are a good resource.

Almost all of the Python tutorials will work with Python 2.7, even though the documentation says that they are written for Python 3.X. My Python installation instructions for Windows below are for Python 2.7 as is the Python source code for the assignments. You can use whatever version of Python that you like, either 2.7 or 3.X (the latest seems to be 3.6). The difference between these two python versions is very minor.

There are many different ways to install OpenCV described on the Internet and on Youtube. I will support the procedures that I describe below, and I suggest you follow my instructions. However, I will try to help you install OpenCV if my instructions do not fit your particular situation. If you have problems installing OpenCV please send me e-mail or come to my office during consulting hours. Try to install OpenCV and run some of the sample program as soon as possible.

Installing OpenCV on Windows

On windows the easiest approach is to use Visual Studio to write OpenCV programs in C++, but as I mentioned you can also use Python. If you are an experienced C++ programmer then using Visual Studio is a good idea, but if you have very little C++ experience then Python is a better alternative; but the choice is yours. For Visual Studio I have included on my web site an example OpenCV project that you can use for your programming assignments without modification. If you use this project then you can simply cut/paste your own C++ source code for each assignment into this project and thereby avoid having to know enough to properly setup a new VisualStudio project for OpenCV from scracth. This example project on the course web site has been setup to use the appropriate directories for OpenCV 3.2 during the compilation and linking process. I choose version OpenCV3.2 because I have used it successfully in the past, but you can use other 3.X version of OpenCV. However, if you do so then you must also update my example Visual Studio project to have the correct information for your installed version of OpenCV. If you follow my instructions below once you install Visual Studio along with OpenCV 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 versions of Visual Studio from Microsoft at https://newacct.scs.carleton.ca/scs_authentication/dreamspark-form.php?department=scs
Note that after you login and select the appropriate software then you will normally be sent a key to unlock the software. You can download an ISO file and burn a DVD, or download a web installer for your software, both will work. I suggest using either Visual Studio 15 (Enterprise or Community Edition) or Visual Studio 17 Community Edition. The easiest approach in my opinion is to use Visual Studio 17 Community edition, and the community editions do not need an unlock key as far as I understand. So the first step is to download Visual Studio 15 or 17 (you want to set up the C++ environment). However, you can use any development environment that you want (including other versions of Visual Studio), since you will only have to submit source code and program output for your assignments.

Below are the installation instructions for OpenCV program on Windows using VisualStudio.

a) You should first copy the file opencv-3. 2.0-vc14.exe (or opencv-3.0.0.exe) to any directory and then execute it, and choose the location for the extraction as the current directory. Then rename the extracted directory which is called opencv to OpenCV3.2 (or OpenCV3.0) and move it to the C: drive so that is now C:\OpenCV3.2 (or OpenCV3. 0).
b) Install Visual Studio on your machine, Version 17 Community Edition is the easiest to install. 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". Restart your system.
f) Copy the folder OpenCVExampleVC15-3.2 or OpenCVExampleVC13-3.0 onto your machine.
g) Go to that directory and open the file with the name of OpenCVExample of type MicroSoft Visual Studio Solution. This will start Visual Studio and then 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 followed by Start without Debugging. It should also work if you select the debug configuration and rebuild, you should try both debug and release mode.

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

The folder OpenCVExampleVC15-3.2 or OpenCVExampleVC13-3.0 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++ program 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 any 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 C++ programs from the OpenCV samples) in Visual Studio, and then pasting the new source into the file OpenCVExample.c++. This requires that you open the new source files in Visual Studio and then do a copy, and paste replacement into OpenCVExample++, and then finally select 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 but I think that Edge works even without a camera.

If you do not want to program in C++, but instead want to use Python for your assignments then you do not need to install Visual Studio at all. Once you have a version of Python with the appropriate packages installed then you must connect your Python to your OpenCV installation.

You can install Python in different ways, but below is one possible installation process for windows. You need Python and the numpy package to run an OpenCV python program. If you are using Python then I suggest installing OpenCV3.0 instead of OpenCV3.2.
a) First 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, and you should use OpenCV3.0, and not OpenCV3.2. But other versions of OpenCV will also work if you make the appropriate changes.

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 OpenCV source code and compile it for your Linux system. On a Linux system such as Ubuntu or Debian (even a Raspberry PI) the easiest installation method is to use a package manager by entering the following command:

sudo apt-get install libopencv-dev python-opencv

If this works, then you are done and you can now use OpenCV from both C++ and Python. If this does not work then for a Linux machine (not a Mac) you need to download the source code for OpenCV (I suggest using version 3.4). However, before you install OpenCV you first need to install some Linux dependencies that are required for running OpenCV. To do this look at step one of
https://www.pyimagesearch.com/2016/10/24/ubuntu-16-04-how-to-install-opencv/
After this is done then you must download and unzip the file 3.4.0.zip from the OpenCV web site and use the following commands to compile and install OpenCV.

1. cd opencv-3.4.0
2. mkdir build
3. cd build
4. cmake ..
5. make –j4 (will take a while)
6. sudo make install

Once this process is finished then you should have OpenCV 3.4 properly installed on your system. Now you can use either C++ or Python with OpenCV, and in fact both python2 and python3 will work.

To compile a C++ program that uses OpenCV on Linux you follow the procedure in
https://docs.opencv.org/2.4/doc/tutorials/introduction/linux_gcc_cmake/linux_gcc_cmake.html
This involves using cmake, followed by make. If you are going to use Linux you should know what cmake/make works since this is the main way that source code is compiled and installed. To use Python

with OpenCV go to opencv-3.4.0/samples/python, and then enter python2 <filename.py>, or python3 <filename.py> where <filename.py> is one of the python sample files. For example, try python2 edge.py or python3 edge.py.

Macbooks are a type of Linux underneath their GUI so the process for installing OpenCV on a Mac is similar to that on Linux, but is still slightly different. To install OpenCV on a Mac you first need to install a package manager and you also need to work in a terminal window. As I said packages is the standard way in which software is maintained and updated in the Linux environment. On an ordinary Linux system like Ubuntu a package manager comes pre-installed on the system, but you have to do something extra to make packages work on the Mac. There are two common ways to make packages work on the Macbook, they are using MacPorts and HomeBrew. In the past I have used MacPorts, but I now prefer to use HomeBrew.

Once you install HomeBrew on your Mac then you should be able to install OpenCV using this package manager. This procedure is described in the two web sites below. It is a bit confusing because in 2017 the process changed slightly, so I will explain it in detail.

https://robferguson.org/blog/2017/10/06/how-to-install-opencv-and-python-using-homebrew-on-macos-sierra/

https://www.learnopencv.com/install-opencv3-on-macos/

The following are the steps which I have condensed from the above two web pages are necessary:

1. Install XCode from the Mac app store.
2. Install homebrew
3. Install python3 (and optionally python2) using brew (which is the homebrew package command)
4. Install opencv using brew, but enter the command “brew install opencv” and not the longer and now incorrect “brew install opencv3 –with-contrib –with python3 (this change is discussed in the second web site above)
5. The instructions in these web sites use virtual environments, which you can also use but this is not necessary. As the last step you need to link python with your installed version of OpenCV, and if you use virtual environments you follow the instructions in the web sites. If you do not use virtual environments then you must link your version of python by entering the commands

   “cd /usr/local/lib/python3.6/site-packages” and then
   “ln –s /usr/local/opt/opencv/lib/python3.6/cv2cpython-36m-darwin.so cv2.so”.

Now you have an environment on your Mac which allows you to use OpenCV with C++ or Python in the same was as standard Linux system. To use C++ with OpenCV you follow the same procedure for an Linux system described in
https://docs.opencv.org/2.4/doc/tutorials/introduction/linux_gcc_cmake/linux_gcc_cmake.html
Since you installed either python3 or python2 then you do not use the standard python command (which is actually the Mac python, and is not what you want to use). Instead you enter python3 <fname.py> where <fname.py> is one of the sample python programs. If the brew installation of OpenCV fails using packages, then you install OpenCV from source as was done for Linux.