Installation Instructions: (For Windows Users)

INSTALLING OPENCV:

1. Download OpenCV from 2.4.9 from this link.
2. Once Download is finished, Run the executable file.
3. Set the Directory path to C:/

INSTALLING VISUAL STUDIO 2013

(Assuming all the workshop attendees have a LAN connection at their respective halls)

1. Click on Start
2. Search for "\\144.16.192.212\" where its written "Search Programs and Files" and click on Software Repository Folder which appears.
3. Browse to Microsoft->Visual-Studio
4. Copy the file "SW_DVD5_Visual_Studio_Pro_2013_English_-2_MLF_X19-27145.iso" to your Desktop.
5. Download Daemon Tools Lite from this link (or PowerISO - link)
6. Once done installing Daemon Tools, the ISO file on your desktop will be recognized by Daemon Tools, and now double-click to mount it.
7. Open My Computer and double click on the Virtual Drive to which the ISO has been mounted to start installation of Visual Studio 2013.
8. Follow the necessary steps (Agree to License and Terms) and on Optional Features to Install, you may deselect all of the selected options to save space, but if you don't mind about the space you can leave it as it is. (Anyways we won't be needing it)
9. Click on Install, and wait for it to finish, after which you should restart your computer.

INTEGRATING OpenCV with Visual Studio

1. Download PathEditor from here - Link
2. Open Path Editor from Desktop
3. Under 'System' Click on 'To edit System PATH (Launch as Admin)'
4. Click on Add under 'System'
5. Browse to
C:/opencv/build/x86/vc12/bin/ (For 32-Bit Users)
C:/opencv/build/x64/vc12/bin (For 64-Bit Users)
(vcXX where XX represents n-1, n being the version of Visual Studio that you installed, For example,
select vc12 if you have
Visual Studio 2013 installed)
(If you don't know if your system is 32-bit or 64-bit ,
Go to Computer, Right click, Go to Properties, Under System type, check whether your system is 32-
bit or 64-bit)
6. Click on 'Ok'
7. Open Visual Studio 2013 and click on New Project.

8. In the New Project Window, Select Visual C++ Under the 'Templates' section and select Win32
Console Application. At the top, change .NET Framework 4.5 to .NET Framework 4
9. Give the project the name "ComputerVision" and press 'OK'
10. Follow the Defaults in the Win32 application wizard.11. Under VIEW tab, Select Other Windows->Property Manager
12. This would form another tab for a window name 'Property Manager' along with 'Solution Explorer' at the right. Select it.
13. Expand the Project name and Right-Click on Debug(Win32)->Add New Project Property Sheet
14. Change the name to PropertySheet_Debug.props
15. Double click on the PropertySheet_Debug.props file in the Property Manager.
16. Click on VC++ Directories -> Executable directories and add the entry
C:\opencv\build\x86\vc12\bin (32 bit)
C:\opencv\build\x64\vc12\bin (64 bit)
17. Click on Library Directories and add the entry:
C:\opencv\build\x86\vc12\lib (32 bit)
C:\opencv\build\x64\vc12\lib (64 bit)
18. Now click on C/C++ on the Left Pane and under Additional Include directories add the entry:
C:\opencv\build\include
19. Click Linker -> General -> Additional Library Directories, add the entry:
C:\opencv\build\x64\vc12\lib (64 bit)
C:\opencv\build\x86\vc12\lib (32 bit)
20. Click Linker-> Input -> Additional Dependencies, add the following entries:
opencv_calib3d249d.lib
opencv_contrib249d.lib
opencv_core249d.lib
opencv_features2d249d.lib
opencv_flann249d.lib
opencv_gpu249d.lib
opencv_highgui249d.lib
opencv_imgproc249d.lib
opencv_legacy249d.lib
opencv_ml249d.lib
opencv_nonfree249d.lib
opencv_objdetect249d.lib
opencv_ocl249d.lib
opencv_photo249d.lib
opencv_stitching249d.lib
opencv_superres249d.lib
opencv_ts249d.lib
opencv_video249d.lib

opencv_videostab249d.lib
21. Now close the window and make a similiar Property Sheet for the Release(Win32) folder with the name PropertySheet_release.props and repeat steps 16-19.
22.Click Linker-> Input -> Additional Dependencies, add the following entries:
opencv_calib3d249.lib
opencv_contrib249.lib
opencv_core249.lib
opencv_features2d249.lib
opencv_flann249.lib
opencv_gpu249.lib
opencv_highgui249.lib
opencv_imgproc249.lib
opencv_legacy249.lib
opencv_ml249.lib
opencv_nonfree249.lib
opencv_objdetect249.lib
opencv_ocl249.lib
opencv_photo249.lib
opencv_stitching249.lib
opencv_superres249.lib
opencv_ts249.lib
opencv_video249.lib
opencv_videostab249.lib

23.Change back into Solution Explorer.
24. The last step , (in 64-Bit Machines) , go to BUILD->Configuration Manager, Under your Project
Name 'ComputerVision' , in 'Platform' column, click on <New..>, Change Platform from:ARM to x64 and press 'OK'.
And you're done!
To check whether it has been configured correctly , Right-Click on Source Files->Add->Existing Item.., browse to C:\opencv\sources\samples\cpp, select drawing.cpp. Click on Play to run the source
code. Happy Coding!

Installation Instructions: (For Ubuntu Users)

In this guide, we’ll install OpenCV with a lot of the features it provides. Here are some of the things that are going to be enabled when you are finished following through with this installation tutorial:

- viz module (3D visualization)
- Qt version of the HighGUI module (Better 2D window interface with zoom, image saving capabilities, etc)
- OpenGL support
- C++ interface and examples
- C interface and examples
- Python interface and examples
- Java interface and examples
- Intel Threading Building Blocks (TBB)

OK, so the first step is to make sure that everything in the system is updated and upgraded. Open the terminal and write this:

```
sudo apt-get
update
```

```
sudo apt-get
upgrade
```

Now, you need to install many dependencies, such as support for reading and writing image files, drawing on the screen, some needed tools, other libraries, etc… This step is very easy, you only need to write the following command in the Terminal:

```
sudo apt-get install build-essential libgtk2.0-dev
libjpeg-dev libtiff4-dev libjasper-dev libopenexr-dev
cmake python-dev python-numpy python-tk libtbb-dev
libeigen3-dev yasm libfaac-dev libopencore-amrnb-dev
libopencore-amrwb-dev libtheora-dev libvorbis-dev
libxvidcore-dev libx264-dev libqt4-dev libqt4-opengl-dev
sphinx-common texlive-latex-extra libv4l-dev
libdc1394-22-dev libavcodec-dev libavformat-dev
libswscale-dev default-jdk ant libvtk5-qt4-dev
```

Time to get the OpenCV 2.4.9 source code:

```
cd ~
```

```
wget http://sourceforge.net/projects/opencvlibrary/files/opencv-unix/2.4.9/opencv-2.4.9.zip
```

```
unzip opencv-2.4.9.zip
```

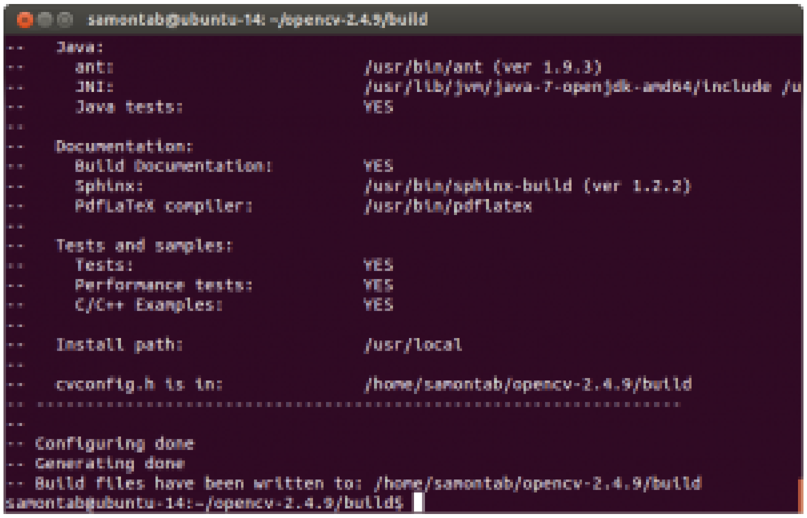
```
cd opencv-2.4.9
```

Now we have to generate the Makefile by using cmake. In here we can define which parts of OpenCV we want to compile. Since we want to use the viz module, Python, Java, TBB, OpenGL, Qt, work with videos, etc, here is where we need to set that. Just execute the following line at the terminal to create the appropriate Makefile. Note that there are two dots at the end of the line, it is an argument for the cmake program and it means the parent directory (because we are inside the build directory, and we want to refer to the OpenCV directory, which is its parent).

```
mkdir build
```

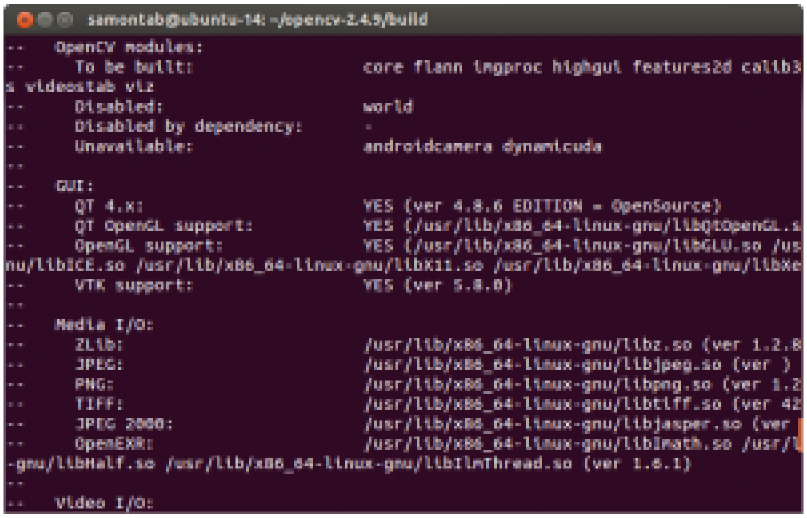
```
cd build
```

```
cmake -D WITH_TBB=ON -D BUILD_NEW_PYTHON_SUPPORT=ON -D WITH_V4L=ON -D INSTALL_C_EXAMPLES=ON -D INSTALL_PYTHON_EXAMPLES=ON -D BUILD_EXAMPLES=ON -D WITH_QT=ON -D WITH_OPENGL=ON -D WITH_VTK=ON ..
```



```
--   Java:
--     ant:                         /usr/bin/ant (ver 1.9.3)
--     JNI:                         /usr/lib/jvm/java-7-openjdk-amd64/include /u
--     Java tests:                  YES
--
--   Documentation:
--     Build Documentation:         YES
--     Sphinx:                      /usr/bin/sphinx-build (ver 1.2.2)
--     PdfLaTeX compiler:           /usr/bin/pdflatex
--
--   Tests and samples:
--     Tests:                       YES
--     Performance tests:           YES
--     C/C++ Examples:              YES
--
--   Install path:                  /usr/local
--
--   cvconfig.h is in:              /home/samontab/opencv-2.4.9/build
-- -----------------------------------------------------------------
--
-- Configuring done
-- Generating done
-- Build files have been written to: /home/samontab/opencv-2.4.9/build
samontab@ubuntu-14:~/opencv-2.4.9/build$
```

Check that the above command produces no error and that in particular it reports FFMPEG as YES. If this is not the case you will not be able to read or write videos. Check that Java, Python, TBB, OpenGL, V4L, OpenGL and Qt are all detected correctly.



```
--   OpenCV modules:
--     To be built:                 core flann imgproc highgui features2d calib3
s videostab viz
--     Disabled:                    world
--     Disabled by dependency:      -
--     Unavailable:                 androidcamera dynamicuda
--
--   GUI:
--     QT 4.x:                      YES (ver 4.8.6 EDITION = OpenSource)
--     QT OpenGL support:           YES (/usr/lib/x86_64-linux-gnu/libQtOpenGL.s
--     OpenGL support:              YES (/usr/lib/x86_64-linux-gnu/libGLU.so /us
nu/libICE.so /usr/lib/x86_64-linux-gnu/libX11.so /usr/lib/x86_64-linux-gnu/libXe
--     VTK support:                 YES (ver 5.8.0)
--
--   Media I/O:
--     ZLib:                        /usr/lib/x86_64-linux-gnu/libz.so (ver 1.2.8
--     JPEG:                        /usr/lib/x86_64-linux-gnu/libjpeg.so (ver )
--     PNG:                         /usr/lib/x86_64-linux-gnu/libpng.so (ver 1.2
--     TIFF:                        /usr/lib/x86_64-linux-gnu/libtiff.so (ver 42
--     JPEG 2000:                   /usr/lib/x86_64-linux-gnu/libjasper.so (ver
--     OpenEXR:                     /usr/lib/x86_64-linux-gnu/libImath.so /usr/l
-gnu/libHalf.so /usr/lib/x86_64-linux-gnu/libIlmThread.so (ver 1.6.1)
--
--   Video I/O:
```

Make sure you scroll up and check that the modules that are going to be built are these:

core flann imgproc highgui features2d calib3d ml video legacy objdetect photo gpu ocl nonfree contrib java python stitching superres ts videostab viz.

If anything is wrong, go back, correct the errors by maybe installing extra packages and then run cmake again.

Now, you are ready to compile and install OpenCV 2.4.9:

```
make
```

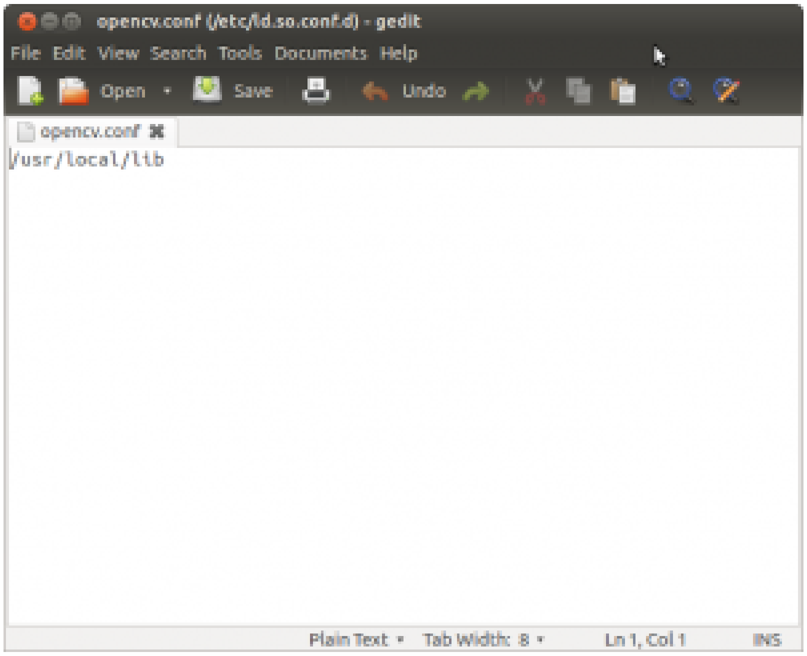
```
sudo make install
```

Now you have to configure OpenCV. First, open the opencv.conf file with the following code:

```
sudo gedit /etc/ld.so.conf.d/opencv.conf
```

Add the following line at the end of the file(it may be an empty file, that is ok) and then save it:

```
/usr/local/lib
```


opencv.conf (/etc/ld.so.conf.d) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
opencv.conf
Plain Text
Tab Width: 8
Ln 1, Col 1
INS

```
/usr/local/lib
```

Run the following code to configure the library:

```
sudo ldconfig
```

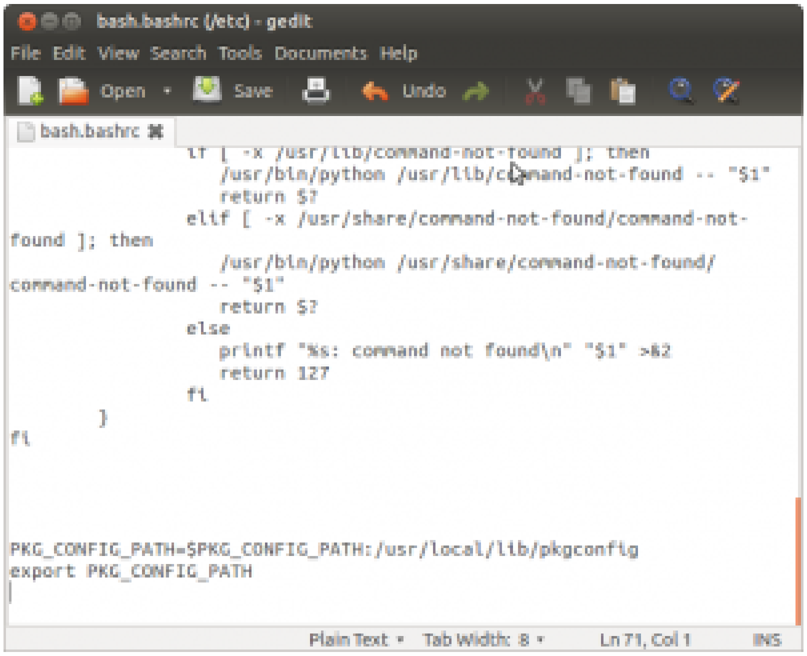
Now you have to open another file:

```
sudo gedit /etc/bash.bashrc
```

Add these two lines at the end of the file and save it:

```
PKG_CONFIG_PATH=$PKG_CONFIG_PATH:/usr/local/lib/pkgconfig
```

```
export PKG_CONFIG_PATH
```



bash.bashrc (/etc) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
bash.bashrc
Plain Text
Tab Width: 8
Ln 71, Col 1
INS

```
                if [ -x /usr/lib/command-not-found ]; then
                   /usr/bin/python /usr/lib/command-not-found -- "$1"
                   return $?
                elif [ -x /usr/share/command-not-found/command-not-
found ]; then
                   /usr/bin/python /usr/share/command-not-found/
command-not-found -- "$1"
                   return $?
                else
                   printf "%s: command not found\n" "$1" >&2
                   return 127
                fi
        }
fi


PKG_CONFIG_PATH=$PKG_CONFIG_PATH:/usr/local/lib/pkgconfig
export PKG_CONFIG_PATH
```

Finally, **close the console and open a new one, restart the computer or logout and then login again**. OpenCV will not work correctly until you do this.

Now you have OpenCV 2.4.9 installed in your computer with 3D visualization, Python, Java, TBB, OpenGL, video, and Qt support.

<u>Add the Compile Script:</u>

Given in this repository will be a script by the name “compile.sh”, download it and keep it in the folder where you will be compiling .cpp/.c files and then building them to binaries.

Run the following command in terminal:

1

```
chmod +x compile.sh
```

There you go!, you have your compile script ready, now all you need to do is use it like this:

2

```
./compile.sh
<FileName>.c/cpp
```