Installation in Linux:

https://docs.opencv.org/3.4.1/d7/d9f/tutorial_linux_install.html

The following steps have been tested for Ubuntu 10.04 but should work with other distros as well.

Required Packages

- GCC 4.4.x or later
- CMake 2.8.7 or higher
- Git
- GTK+2.x or higher, including headers (libgtk2.0-dev)
- pkg-config
- Python 2.6 or later and Numpy 1.5 or later with developer packages (python-dev, python-numpy)
- ffmpeg or libav development packages: libavcodec-dev, libavformat-dev, libswscale-dev
- [optional] libtbb2 libtbb-dev
- [optional] libdc1394 2.x
- [optional] libjpeg-dev, libpng-dev, libtiff-dev, libjasper-dev, libdc1394-22-dev
- [optional] CUDA Toolkit 6.5 or higher

The packages can be installed using a terminal and the following commands or by using Synaptic Manager:
[compiler] sudo apt-get install build-essential
[required] sudo apt-get install cmake git libgtk2.0-dev pkg-config libavcodec-dev libavformat-dev libswscale-dev
[optional] sudo apt-get install python-dev python-numpy libtbb2 libtbb-dev libjpeg-dev libpng-dev libtiff-dev libjasper-dev libdc1394-22-dev

Getting OpenCV Source Code

You can use the latest stable OpenCV version or you can grab the latest snapshot from our Git repository.

Getting the Latest Stable OpenCV Version

- Go to our downloads page.
- Download the source archive and unpack it.

Getting the Cutting-edge OpenCV from the Git Repository

Launch Git client and clone OpenCV repository. If you need modules from OpenCV contrib repository then clone it as well.

For example

cd ~/<my_working_directory>
git clone https://github.com/opencv/opencv.git
git clone https://github.com/opencv/opencv_contrib.git

Building OpenCV from Source Using CMake

1. Create a temporary directory, which we denote as <cmake_build_dir>, where you want to put the generated Makefiles, project files as well the object files and output binaries and enter there.
    - For example
        i. cd ~/opencv
        ii. mkdir build
        iii. cd build
2. Configuring. Run cmake [<some optional parameters>] <path to the OpenCV source directory>
    - For example
        i. cmake -D CMAKE_BUILD_TYPE=Release -D CMAKE_INSTALL_PREFIX=/usr/local ..
    - or cmake-gui
        i. set full path to OpenCV source code, e.g. /home/user/opencv
        ii. set full path to <cmake_build_dir>, e.g. /home/user/opencv/build
        iii. set optional parameters
        iv. run: “Configure”
        v. run: “Generate”
    - Notecd
        i. Use cmake -DCMAKE_BUILD_TYPE=Release -DCMAKE_INSTALL_PREFIX=/usr/local .. (without spaces after -D if the above example doesn't work.)
3. Description of some parameters
    - build type: CMAKE_BUILD_TYPE=Release\Debug
    - to build with modules from opencv_contrib set OPENCV_EXTRA_MODULES_PATH to <path to opencv_contrib/modules/>
    - set BUILD_DOCS for building documents
    - set BUILD_EXAMPLES to build all examples
4. [optional] Building python. Set the following python parameters:
    - PYTHON2(3)_EXECUTABLE = <path to python>
    - PYTHON_INCLUDE_DIR = /usr/include/python<version>
    - PYTHON_INCLUDE_DIR2 = /usr/include/x86_64-linux-gnu/python<version>
    - PYTHON_LIBRARY = /usr/lib/x86_64-linux-gnu/libpython<version>.so
    - PYTHON2(3)_NUMPY_INCLUDE_DIRS = /usr/lib/python<version>/dist-packages/numpy/core/include/
5. [optional] Building java.

- Unset parameter: BUILD_SHARED_LIBS
- It is useful also to unset BUILD_EXAMPLES, BUILD_TESTS, BUILD_PERF_TESTS - as they all will be statically linked with OpenCV and can take a lot of memory.

6. Build. From build directory execute *make*, it is recommended to do this in several threads
    - For example
        i. make -j7 # runs 7 jobs in parallel
7. [optional] Building documents. Enter <cmake_build_dir/doc/> and run make with target "doxygen"
    - **PERSONAL SIDE NOTE*** by Josh Sullivan
        i. In opencv/build -> mkdir doxygen
        ii. git clone https://github.com/doxygen/doxygen.git
        cd doxygen
        mkdir build
        cd build
        cmake -G "Unix Makefiles" ..
        make
        make install

    ** END SIDE NOTE...CONTINUING WITH TUTORIAL**
    - For example
        i. cd ~/opencv/build/doxygen
            1. **side note ** The new directory is doxygen if you followed my side note. Otherwise the tutorial says cd ~/opencv/build/doc

        make -j7 doxygen
8. To install libraries, execute the following command from build directory
    - sudo make install
9. [optional] Running tests
    - Get the required test data from [OpenCV extra repository](https://github.com/opencv/opencv_extra).
    - For example
        i. git clone https://github.com/opencv/opencv_extra.git
            1. set OPENCV_TEST_DATA_PATH environment variable to <path to opencv_extra/testdata>.
            2. execute tests from build directory.
    - For example
        i. <cmake_build_dir>/bin/opencv_test_core

Note

If the size of the created library is a critical issue (like in case of an Android build) you can use the install/strip command to get the smallest size possible. The *stripped* version appears to be twice as small. However, we do not recommend using this unless those extra megabytes do really matter.