Installation in Linux

Next Tutorial: Using OpenCV with gcc and CMake

Original author	Ana Huamán
Compatibility	OpenCV >= 3.0

Table of Contents

Quick start

Build core modules

```
# Install minimal prerequisites (Ubuntu 18.04 as reference)
sudo apt update && sudo apt install -y cmake g++ wget unzip

# Download and unpack sources
wget -O opencv.zip https://github.com/opencv/opencv/archive/master.zip
unzip opencv.zip

# Create build directory
mkdir -p build && cd build

# Configure
cmake  ../opencv-master

# Build
cmake --build .
```

Build with opencv_contrib

```
# Install minimal prerequisites (Ubuntu 18.04 as reference)
sudo apt update && sudo apt install -y cmake g++ wget unzip

# Download and unpack sources
wget -O opencv.zip https://github.com/opencv/opencv/archive/master.zip
wget -O opencv_contrib.zip https://github.com/opencv/opencv_contrib/archive/master.zip
unzip opencv.zip
unzip opencv_contrib.zip

# Create build directory and switch into it
mkdir -p build && cd build

# Configure
cmake -DOPENCV_EXTRA_MODULES_PATH=../opencv_contrib-master/modules ../opencv-master

# Build
cmake --build .
```

Detailed process

This section provides more details of the build process and describes alternative methods and tools. Please refer to the **OpenCV installation overview** tutorial for general installation details and to the **OpenCV configuration options reference** for configuration options documentation.

Install compiler and build tools

- To compile OpenCV you will need a C++ compiler. Usually it is G++/GCC or Clang/LLVM:
  - Install GCC...
    ```
    sudo apt install -y g++
    ```
  - ... or Clang:
    ```
    sudo apt install -y clang
    ```
- OpenCV uses CMake build configuration tool:
  ```
  sudo apt install -y cmake
  ```
- CMake can generate scripts for different build systems, e.g. *make*, *ninja*:
  - Install Make...
    ```
    sudo apt install -y make
    ```
  - ... or Ninja:
    ```
    sudo apt install -y ninja-build
    ```

- Install tool for getting and unpacking sources:
  - *wget* and *unzip*...

    ```
    sudo apt install -y wget unzip
    ```

  - ... or *git*:

    ```
    sudo apt install -y git
    ```

Download sources

There are two methods of getting OpenCV sources:

- Download snapshot of repository using web browser or any download tool (~80-90Mb) and unpack it...

  ```
  wget -O opencv.zip https://github.com/opencv/opencv/archive/master.zip
  unzip opencv.zip
  mv opencv-master opencv
  ```

- ... or clone repository to local machine using *git* to get full change history (>470Mb):

  ```
  git clone https://github.com/opencv/opencv.git
  git -C opencv checkout master
  ```

Note

Snapshots of other branches, releases or commits can be found on the GitHub and the official download page.

Configure and build

- Create build directory:

  ```
  mkdir -p build && cd build
  ```

- Configure - generate build scripts for the preferred build system:
  - For *make*...

    ```
    cmake ../opencv
    ```

  - ... or for *ninja*:

    ```
    cmake -GNinja ../opencv
    ```

- Build - run actual compilation process:
  - Using *make*...

    ```
    make -j4
    ```

  - ... or *ninja*:

    ```
    ninja
    ```

Note

Configure process can download some files from the internet to satisfy library dependencies, connection failures can cause some of modules or functionalities to be turned off or behave differently. Refer to the **OpenCV installation overview** and **OpenCV configuration options reference** tutorials for details and full configuration options reference.

If you experience problems with the build process, try to clean or recreate the build directory. Changes in the configuration like disabling a dependency, modifying build scripts or switching sources to another branch are not handled very well and can result in broken workspace.

Make can run multiple compilation processes in parallel, `-j<NUM>` option means "run <NUM> jobs simultaneously". *Ninja* will automatically detect number of available processor cores and does not need `-j` option.

Check build results

After successful build you will find libraries in the `build/lib` directory and executables (test, samples, apps) in the `build/bin` directory:

```
ls bin
ls lib
```

CMake package files will be located in the build root:

```
ls OpenCVConfig*.cmake
ls OpenCVModules.cmake
```

Install

Warning

The installation process only copies files to predefined locations and does minor patching. Installing using this method does not integrate opencv into the system package registry and thus, for example, opencv can not be uninstalled automatically. We do not recommend system-wide installation to regular users due to possible conflicts with system packages.

By default OpenCV will be installed to the `/usr/local` directory, all files will be copied to following locations:

- `/usr/local/bin` - executable files
- `/usr/local/lib` - libraries (.so)
- `/usr/local/cmake/opencv4` - cmake package
- `/usr/local/include/opencv4` - headers
- `/usr/local/share/opencv4` - other files (e.g. trained cascades in XML format)

Since `/usr/local` is owned by the root user, the installation should be performed with elevated privileges (`sudo`):

```
sudo make install
```

or

```
sudo ninja install
```

Installation root directory can be changed with `CMAKE_INSTALL_PREFIX` configuration parameter, e.g. `-DCMAKE_INSTALL_PREFIX=$HOME/.local` to install to current user's local directory. Installation layout can be changed with `OPENCV_*_INSTALL_PATH` parameters. See **OpenCV configuration options reference** for details.