

Installation of ORB- SLAM3/2

Pascal Cosimo Fina

August 2022

0.1 Introduction

This is a step by step tutorial focusing on the ORB-SLAM 3 installation. It is essential that you use the described versions. Based on those versions, I will show how to solve each bug that might occur in the installation process. Since the installation of ORB-Slam2 is quite similar, there will be a short chapter about ORB-SLAM2 installation.

1 Installation of Pangolin

1.1 Tested Versions/set up

- Pangolin: standard version(provided by the Github link beneath)
- Ubuntu 20.04 headless
- Raspberry pi 4
- Cmake compiler: 3.23.3
- Python version: 2.7.18
- Gcc: 9.4.0

In order to find out which version is used, try:

```
<application> --version
```

This description is based on the version which are listed above. Therefore I would strongly advise to use those. Otherwise there might occur other errors

1.2 Installation

Fortunately there shouldn't be any bugs if you use the setup from above.

Lets start...

Create a new directory called 'Dev' and access it:

```
mkdir Dev
cd Dev
```

Clone the provided repository, and enter Pangolin directory Create a new directory 'build' Build Pangolin, compile and install it

```
cd Pangolin
mkdir build
cd build
cmake .. -D CMAKE_BUILD_TYPE=Release
///nproc-> number of cores/threads
make -j <nproc + 1>
sudo make install
```

That's it, Pangolin should be successfully installed now!:)

1.3 References

Description of installation (section: "Install Pangolin")

2 Installation of OpenCV

2.1 Tested Versions/set up

- OpenCV: 3.2.0
Note: You should definitely use this OpenCV version otherwise ORB SLAM 2 might not work!!
- Ubuntu 20.04 headless
- Raspberry pi 4
- cmake compiler: 3.23.3
- python version: 2.7.18
- gcc: 9.4.0

In order to find out which version is used, try:

```
<application> --version
```

This description is based on the version which are listed above. Therefore I would strongly advise to use those. Otherwise there might occur other errors

2.2 Installation process

I would recommend to install OpenCV also in the Dev directory. Clone the repository and enter OpenCV directory...

```
cd Dev
git clone https://github.com/opencv/opencv.git
cd opencv
git checkout 3.2.0
```

2.3 First fix

Use the following command to edit this file:

```
vim ./modules/videoio/src/cap_ffmpeg_impl.hpp
```

Now add the following lines at the beginning of the file(right beneath the first include):

```
#define AV_CODEC_FLAG_GLOBAL_HEADER (1 << 22)
#define CODEC_FLAG_GLOBAL_HEADER AV_CODEC_FLAG_GLOBAL_HEADER
#define AVFMT_RAWPICTURE 0x0020
```

Save the file and close it(using ":x")! Create a build directory and navigate in it:

```
mkdir build
cd build
```

2.4 Second error/ fix

Move to the opencv directory using and open the following file:

```
cd ..
vim /cmake/ OpenCVCompilerOptions.cmake
```

In the image below is a code snippet of the provided file. Unfortunately there is one 'endif' missing which needs to be inserted manually(at cursor location) otherwise compilation won't succeed!!

```

if(ENABLE_CCACHE AND NOT CMAKE_COMPILER_IS_CCACHE)
# This works fine with Unix Makefiles and Ninja generators
find_host_program(CCACHE_PROGRAM ccache)
if(CCACHE_PROGRAM)
    message(STATUS "Looking for ccache - found (${CCACHE_PROGRAM})")
    get_property(_OLD_RULE_LAUNCH_COMPILE GLOBAL PROPERTY RULE_LAUNCH_COMPILE)
    if(_OLD_RULE_LAUNCH_COMPILE)
        message(STATUS "Can't replace CMake compiler launcher")
    else()
        set_property(GLOBAL PROPERTY RULE_LAUNCH_COMPILE "${CCACHE_PROGRAM}")
        # NOTE: Actually this check doesn't work as expected.
        # "RULE_LAUNCH_COMPILE" is ignored by CMake during try_compile() step.
        # ocv_check_compiler_flag(CXX "" IS_CCACHE_WORKS)
        set(IS_CCACHE_WORKS 1)
        if(IS_CCACHE_WORKS)
            set(CMAKE_COMPILER_IS_CCACHE 1)
        else()
            message(STATUS "Unable to compile program with enabled ccache, reverting...")
            set_property(GLOBAL PROPERTY RULE_LAUNCH_COMPILE "${_OLD_RULE_LAUNCH_COMPILE}")
        endif()
    endif()
else()
    message(STATUS "Looking for ccache - not found")
endif()
endif()

```

Press 'i' to insert missing text and save and close it using ':x'

2.5 Third error/fix

You need to move to the opencv directory and open the following file. There is a missing 'const' which needs to be inserted manually(marked line).

```
vim modules/python/src2/cv2.cpp
```

```

template<
bool pyopencv_to(PyObject* obj, String& value, const char* name)
{
    (void)name;
    if(!obj || obj == Py_None)
        return true;
    const char* str = PyString_AsString(obj); //error fix pascal inserted const vim modules/python/src2/cv2.cpp
    if(!str)
        return false;
    value = String(str);
    return true;
}

```

Press 'i' to insert missing text and save and close it using ':x'

2.6 Building with cmake

After you fixed all those bugs it is time to build openCV...

It is now essential using the following command:

```
cmake -D CMAKE_BUILD_TYPE=Release -D WITH_CUDA=OFF
-D CMAKE_INSTALL_PREFIX=/usr/local -DENABLE_PRECOMPILED_HEADERS=OFF ..
```

Thereby you have to use this flag:

```
-DENABLE_PRECOMPILED_HEADERS=OFF
```

Otherwise compiler complains about not being able to find the C++ version of
stdlib.h

Now run:

```
///nproc-> number of cores/threads  
make -j <nopcr + 1>  
sudo make install
```

OpenCV 2.3 should be successfully installed now!:))

2.7 References

Description of OpenCv 3.2 installation(GitHub Repo)

Fixing Bug: fatal error: stdlib.h: No such file or directory

Fixing Bug: error: invalid conversion from ‘const char*’ to ‘char*’

3 Installation of Eigen library

3.1 Tested Versions/set up

- Eigen: 3.1.0
Note: You should use this Eigen version. Newer versions might be possible as well(not recommended)!
- Ubuntu 20.04 headless
- Raspberry pi 4
- cmake compiler: 3.23.3
- python version: 2.7.18
- gcc: 9.4.0

In order to find out which version is used, try:

```
<application> --version
```

This description is based on the version which are listed above. Therefore I would strongly advise to use those. Otherwise there might occur other errors

3.2 Installation

Fortunately, there are not any bugs to fix:). Just get the recommended Eigen Version(3.1:)). Unzip the folder and store it in the "Dev" Directory:

```
unzip eigen-3.1.0 -d Dev  
cd eigen-3.1.0  
mkdir build && cd build
```

Now configure and install the Eigen library:

```
sudo cmake ..  
sudo make install  
sudo cp -r /usr/local/include/eigen3/Eigen /usr/local/include
```

That's it! The Eigen library should be successfully installed now!)

3.3 References

Eigen Versions
Short Tutorial
Main Repo

4 Installation of DBoW2 and g2o

These applications are already included in the third-party folder!(no further installation needed!!)

5 Installation of ROS

In case you want to install ROS as well I would recommend you to install ROS Melodic. Just go to the official website ROS Melodic and install it. There should not be any issues as far as this is concerned...

6 Installation of ORB- SLAM3

6.1 First steps

This description is based on the version which are listed above. Therefore I would strongly advise to use those. Otherwise there might occur other errors.

In case you installed all of the dependency's we can go on installing...

As usual move to the Dev folder and clone the following Repo:

```
cd Dev  
git clone https://github.com/UZ-SLAMLab/ORB_SLAM3.git ORB_SLAM3
```

And now we need to fix some more bugs...

6.2 First bug/fix

In the file below the "const" in line 51 is in the wrong position. In C++ the position of a "const" makes a difference if one uses pointers.

You have to change:

```
Eigen::aligned_allocator<std::pair<const KeyFrame*, g2o::Sim3> > > KeyFrameAndPose;
```

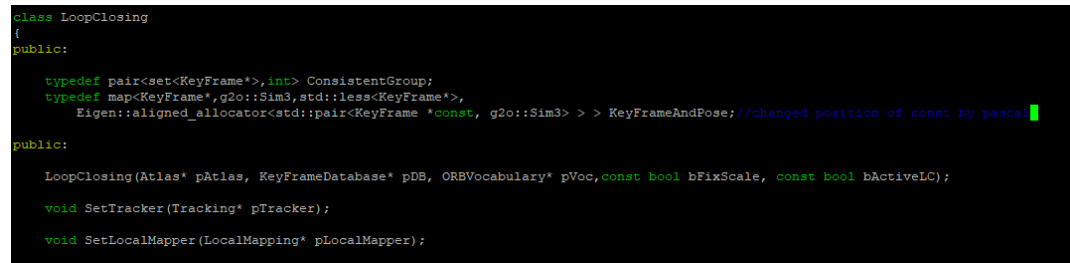
to

```
Eigen::aligned_allocator<std::pair<KeyFrame *const, g2o::Sim3> > > KeyFrameAndPose;
```

Therefore navigate in ORB-SLAM3 folder and edit the file using i and save it using :x! :

```
cd ORB_SLAM3
vim ./include/LoopClosing.h
```

The file should look like this now:



```
class LoopClosing
{
public:
    typedef pair<set<KeyFrame*>,int> ConsistentGroup;
    typedef map<KeyFrame*,g2o::Sim3,std::less<KeyFrame*>,
        Eigen::aligned_allocator<std::pair<KeyFrame *const, g2o::Sim3> > > > KeyFrameAndPose; //changed position of const by pascal
public:
    LoopClosing(Atlas* pAtlas, KeyFrameDatabase* pDB, ORBVocabulary* pVoc,const bool bFixScale, const bool bActiveLC);
    void SetTracker(Tracking* pTracker);
    void SetLocalMapper(LocalMapping* pLocalMapper);
```

It might be possible that the error has been already fixed by the publishers. You should check out this file in any case though!!

6.3 Second bug/fix

The CMakeLists.txt is compiling with C++ 11. Unfortunately, some files can't be compiled using this compiler version.

Therefore, you have to update the compiler version to at least C++14. **I tested it with C++14, so I would recommend you to do the same!!!** Just use the commands below, otherwise you would face a "firework of errors":

```
cd ORB_SLAM3
sed -i 's/++11/++14/g' CMakeLists.txt
```

(this is hacky shit but it works)

6.4 Third bug/fix

It might occur that the compiler is complaining that your OpenCV version is too old. Therefore we need to change one more line in the CMakeLists.txt ((in short range of) line 31):

```
vim CMakeLists.txt
```

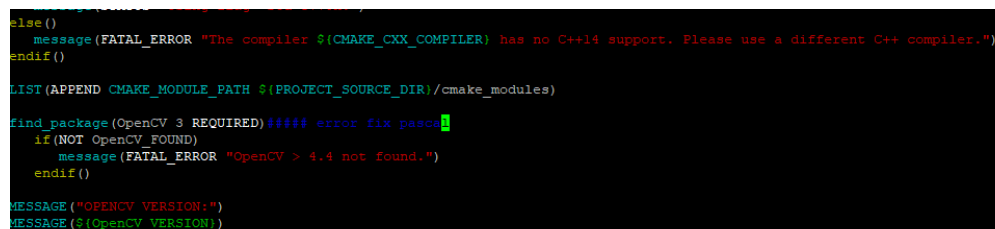
You have to replace:

```
find_package(OpenCV 4.4)
```

with:

```
find_package(OpenCV 3 REQUIRED)
```

The file should look like this now:



```
else()
    message(FATAL_ERROR "The compiler ${CMAKE_CXX_COMPILER} has no C++14 support. Please use a different C++ compiler.")
endif()

LIST(APPEND CMAKE_MODULE_PATH ${PROJECT_SOURCE_DIR}/cmake_modules)

find_package(OpenCV 3 REQUIRED)#### error fix pasce
if(NOT OpenCV_FOUND)
    message(FATAL_ERROR "OpenCV > 4.4 not found.")
endif()

MESSAGE("OPENCV VERSION:")
MESSAGE(${OpenCV_VERSION})
```

Note: You don't need to change the print statement beneath.

6.5 Further installation

```
cd ORB_SLAM3
chmod +x build.sh
//in case a permission denied error might occur
sudo su
./build.sh
```

6.6 One more problem- memory storage!

While installing, one further error might happen.

C++: fatal error: Killed signal terminated program cc1plus

This is a common case if one uses virtual machines, raspberry pis and so on ... However, there is also the possibility that nothing of this is happening to you- maybe you got a lucky day. In case you don't, two solutions will be presented in the further part ...

Fixing this error can be tricky. This error message implies that you don't possess enough memory storage for the installation. First of all you should reboot the whole system once or even twice and run the installation(section above) again. In case the error still remains, we need to allocate more memory by creating a swap partition.

6.6.1 Swap partition

First we create a partition path:

```
sudo mkdir -p /var/cache/swap/
```

Furthermore I recommend to set the size of the partition $bs=64M$, $count=64$, so the swap space size is $bs*count=4096MB=4GB$ - and grant permission to the directory:

```
sudo dd if=/dev/zero of=/var/cache/swap/swap0 bs=64M count=64
sudo mkswap /var/cache/swap/swap0
```

Next you need to activate the file and evaluate whether the swap is correct:

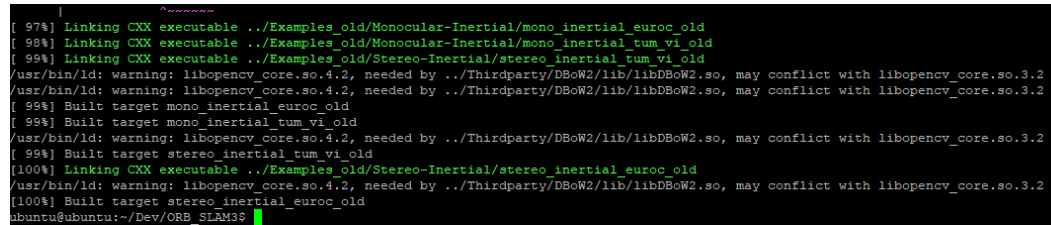
```
sudo swapon /var/cache/swap/swap0
sudo swapon -s
```

Now you should be able to compile without getting the error. So just use the commands from section "Further installation"!

6.7 Steps after installation

Installation is finally done! :D

Your final output will probably look like this:



```
|
[ 97%] Linking CXX executable ../Examples_old/Monocular-Inertial/mono_inertial_euroc_old
[ 98%] Linking CXX executable ../Examples_old/Monocular-Inertial/mono_inertial_tum_vi_old
[ 99%] Linking CXX executable ../Examples_old/Stereo-Inertial/stereo_inertial_tum_vi_old
/usr/bin/ld: warning: libopencv_core.so.4.2, needed by ../Thirdparty/DBow2/lib/libDBow2.so, may conflict with libopencv_core.so.3.2
/usr/bin/ld: warning: libopencv_core.so.4.2, needed by ../Thirdparty/DBow2/lib/libDBow2.so, may conflict with libopencv_core.so.3.2
[ 99%] Built target mono_inertial_euroc_old
[ 99%] Built target mono_inertial_tum_vi_old
/usr/bin/ld: warning: libopencv_core.so.4.2, needed by ../Thirdparty/DBow2/lib/libDBow2.so, may conflict with libopencv_core.so.3.2
[ 99%] Built target stereo_inertial_tum_vi_old
[100%] Linking CXX executable ../Examples_old/Stereo-Inertial/stereo_inertial_euroc_old
/usr/bin/ld: warning: libopencv_core.so.4.2, needed by ../Thirdparty/DBow2/lib/libDBow2.so, may conflict with libopencv_core.so.3.2
[100%] Built target stereo_inertial_euroc_old
ubuntu@ubuntu:~/Dev/ORB_SLAM3$
```

Furthermore, I would recommend to free the memory you have allocated for the compilation- you don't need it anymore!

Commands to delete the swap partition:

```
sudo swapoff /var/cache/swap/swap0
sudo rm /var/cache/swap/swap0
```

Command to free the allocated memory:

```
sudo swapoff -a
```

From now on I would kindly direct you to the original Repo ORB-SLAM3. There you can download provided data sets and test your ORB-SLAM. There is also the opportunity to calibrate it with your own camera(further explanation in the referenced Repo). I hope this guide could help to install ORB-SLAM3 successfully:)))

6.8 References

Solving first bug
Solving second bug
Solving third bug
Solution - swap partition
ORB-SLAM3
ORB-SLAM3 Installation tutorial

7 Short guide to install ORB SLAM2

7.1 Introduction

In general, both Slams are quite similar. They both use feature extraction and use almost the same algorithms as far as mapping and tracking is concerned. The main difference between those two slams is that ORB-SLAM3 is also compatible with pin-hole and fish-eye lens models. Therefore, the installation of OpenCV, Eigen, Pangolin etc. is exactly the same. If you haven't done that so far, just move back to the first chapter, carry out each step until "Installation of ORB-SLAM3" and get back here to the installation of ORB-SLAM2.

7.2 Installation of ORB-SLAM2

Move in the Dev directory and clone the Repo there:

```
cd Dev
git clone https://github.com/raulmur/ORB_SLAM2.git ORB_SLAM2
```

7.2.1 First error/fix

Similar to ORB-SLAM3, the CMakeLists.txt is compiling with C++ 11. Unfortunately, some files can't be compiled using this compiler version.

Therefore, you have to update the compiler version to at least C++14. **I tested it with C++14, so I would recommend you to do the same!!!** Just use the commands below, otherwise you would face a "firework of errors":

```
cd ORB_SLAM3
sed -i 's/++11/++14/g' CMakeLists.txt
```

(this is still hacky shit but it works)

7.2.2 Second error/fix

It might also occur that the compiler is complaining that your OpenCV version is too old(like in ORB-SLAM3). Therefore we need to change one more line in the CMakeLists.txt ((in short range of) line 31):

```
vim CMakeLists.txt
```

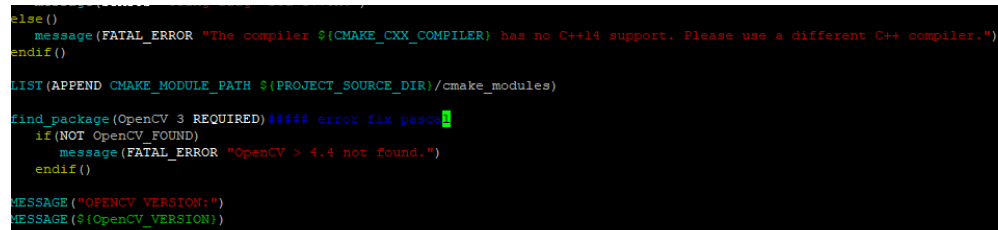
You have to replace:

```
find_package(OpenCV 4.4)
```

with:

```
find_package(OpenCV 3 REQUIRED)
```

The file should look like this now:



```
else()
    message(FATAL_ERROR "The compiler ${CMAKE_CXX_COMPILER} has no C++14 support. Please use a different C++ compiler.")
endif()

LIST(APPEND CMAKE_MODULE_PATH ${PROJECT_SOURCE_DIR}/cmake_modules)

find_package(OpenCV 3 REQUIRED)#### error fix pasce
if(NOT OpenCV_FOUND)
    message(FATAL_ERROR "OpenCV > 4.4 not found.")
endif()

MESSAGE("OPENCV VERSION:")
MESSAGE(${OpenCV_VERSION})
```

7.3 Final steps

Carry out the commands beneath:

```
cd ORB_SLAM3
chmod +x build.sh
//in case a permission denied error might occur
sudo su
./build.sh
```

In case you got the error:

```
C++: fatal error: Killed signal terminated program cc1plus
```

You need to create a swap partition, therefore I would kindly direct you to subsection: "swap partition"(section: "installation of ORB-SLAM3) By now installation should have been successfully. Finally I direct you to the main repo of ORB-SLAM2. Good luck!