install opency 4.2 and contrib in ubantu linux

\* Edit

To install **OpenCV 4.2** with the **contrib modules** on **Ubuntu Linux**, follow these steps. This process compiles OpenCV from source, which gives you full flexibility and access to all contrib modules.

# Step-by-Step Installation of OpenCV 4.2 + Contrib on Ubuntu 1. Install Dependencies Dash Copy

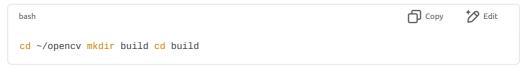
sudo apt update sudo apt install -y build-essential cmake git pkg-config libgtk-3-dev \ libavcodec-dev libavformat-dev libswscale-dev libv4l-dev \ libxvidcore-dev libx264-dev libjpeg-dev libpng-dev libtiff-dev \ gfortran openexr libatlas-base-dev python3-dev python3-numpy \ libtbb2 libtbb-dev libdc1394-22-dev

#### 2. Clone OpenCV and OpenCV-Contrib (v4.2.0)

bash

Copy

#### 3. Create Build Directory



#### 4. Configure with CMake



#### 5. Compile (This may take 20–90 minutes)

Use -j with number of CPU cores, e.g., -j\$(nproc)

bash	Сору	* Edit
make -j\$(nproc)		

#### 6. Install

bash	Сору	* Edit
sudo make install sudo ldconfig		

## 7. Verify Installation

Check version with Python:

bash	Сору	* Edit
<pre>python3 -c "import cv2; print(cv2version)"</pre>		

It should show 4.2.0.

Would you like help setting up a Python virtual environment with OpenCV 4.2 too?



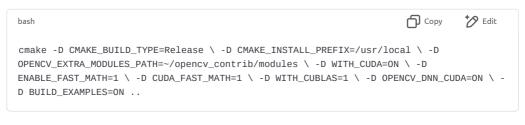
```
cmake -D CMAKE_BUILD_TYPE=Release \
    -D CMAKE_INSTALL_PREFIX=/usr/local \
    -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib/modules \
    -D BUILD_EXAMPLES=ON .. complie wit gpu
```

To compile OpenCV 4.2 with GPU (CUDA) support, you need to make sure your system has:

- An NVIDIA GPU
- CUDA and cuDNN installed
- Correct version compatibility between OpenCV 4.2, CUDA, and your NVIDIA driver

# ☑ Update CMake Command for GPU (CUDA) Support

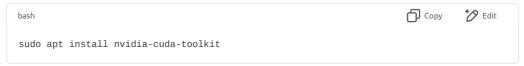
Here's the modified cmake command:



# X Additional Setup Steps

#### 1. Install CUDA and cuDNN

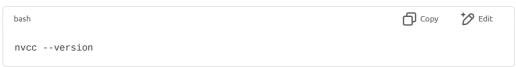
If you haven't already:



⚠ However, it's better to install CUDA and cuDNN directly from NVIDIA's website to match your GPU.

#### 2. Check CUDA Availability

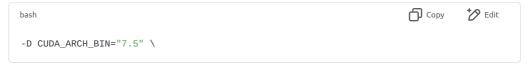
After installing, verify:



Make sure it's detected correctly.

#### 3. Optional: Set Compute Capability

To optimize for your GPU, add:



Replace 7.5 with your GPU's compute capability (check here).

Example:

-D CUDA\_ARCH\_BIN="6.1" # For GTX 1050 Ti

# After CMake:

make -j\$(nproc) sudo make install sudo ldconfig

# Test CUDA in OpenCV

#### Look for:

- CUDA = YES
- cuDNN = YES (if used)
- NVIDIA GPU arch = present

Would you like help verifying if your GPU is compatible with CUDA and OpenCV 4.2?

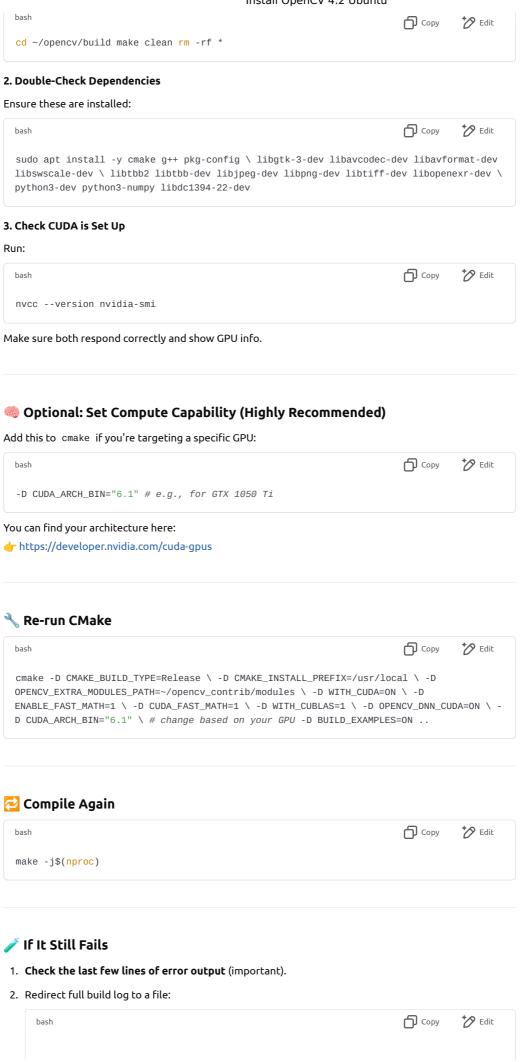
# 日の句のかのいく

torch importer.cpp.o [47%] Building CXX object modules/features2d/CMakeFiles/opencv\_test\_features2d.dir/test/t est nearestneighbors.cpp.o [47%] Building CXX object modules/features2d/CMakeFiles/opencv\_test\_features2d.dir/test/t est\_orb.cpp.o [47%] Building CXX object modules/features2d/CMakeFiles/opencv\_test\_features2d.dir/test/t est\_utils.cpp.o [ 47%] Linking CXX executable ../../bin/opencv\_test\_features2d [ 47%] Linking CXX executable ../../bin/opencv\_test\_dnn [ 47%] Built target opencv\_test\_features2d [47%] Built target opencv\_test\_dnn Makefile:162: recipe for target 'all' failed make: \*\*\* [all] Error 2make -j\$(nproc) clean change cmake cmake -D CMAKE\_BUILD\_TYPE=Release \ -D CMAKE\_INSTALL\_PREFIX=/usr/local \ -D OPENCV\_EXTRA\_MODULES\_PATH=~/opencv\_contrib/modules -D WITH\_CUDA=ON \ -D ENABLE\_FAST\_MATH=1 \ -D CUDA\_FAST\_MATH=1 \ -D WITH CUBLAS=1 \ -D OPENCV\_DNN\_CUDA=ON \ -D BUILD EXAMPLES=ON ..

You're encountering a **build error at 47%**, which likely occurred during CUDA-related linking or compilation. Let's go through a **clean rebuild process** and some key steps to fix it.

# ☑ Step-by-Step Fix: Clean Build with CUDA Support

#### 1. Clean Previous Build



```
make -j$(nproc) 2>&1 | tee build_log.txt
```

3. Share the last 20–30 lines of the log with me, and I can pinpoint the cause.

Would you like me to help you find your GPU's exact CUDA compute capability?



(base) sakthees@sakthees-monk:~/opencv/build\$ nvcc --version nvcc: NVIDIA (R) Cuda compiler driver Copyright (c) 2005-2016 NVIDIA Corporation Built on Tue\_Jan\_10\_13:22:03\_CST\_2017 Cuda compilation tools, release 8.0, V8.0.61 (base) sakthees@sakthees-monk:~/opencv/build\$ nvidia-smi Tue May 6 10:14:06 2025 NVIDIA-SMI 390.157 Driver Version: 390.157 | GPU Name Persistence-M| Bus-Id Disp.A | Volatile Uncorr. ECC | | Fan Temp Perf Pwr:Usage/Cap| | Memory-Usage | GPU-Util Compute M. | 0 GeForce GT 730 Off | 00000000:01:00.0 N/A | N/A | | 11% 48C P8 N/A / N/A | 240MiB / 4025MiB | N/A Default GPU Memory | | Processes: GPU PID Type Process name Usage |-----Not Supported

Thanks for the details — here's what we're dealing with:

#### Critical Issues

- 1. Your CUDA version is 8.0 (from 2016)
  - → **Too old** for OpenCV 4.2 (which expects CUDA ≥10.0).
- 2. **NVIDIA Driver is 390.157** 
  - $_{\rightarrow}$  Also outdated not compatible with newer CUDA versions.
- 3. GPU: GeForce GT 730
  - $_{\rightarrow}\,$  Only supports up to Compute Capability 3.5
  - $\rightarrow$  Many CUDA features (like opencv\_dnn\_cuda ) are **not supported** on this GPU.

# Recommendation

To build OpenCV 4.2 with CUDA, you **must** upgrade:

A. Upgrade to CUDA 10.2 or newer

But your **driver (390.157)** only supports CUDA 9.1 max.

So you must also upgrade your NVIDIA driver first.

#### **B.** Check Compatibility

GT 730 (Kepler, Compute Capability 3.5) support was dropped after CUDA 10.2.

This means:

- You can only use CUDA 10.2 with GT 730.
- Anything newer (11.x, 12.x) won't support your GPU.

# TOP Option 1: Stay with GT 730

- 1. Upgrade to CUDA 10.2 + driver >= 440.x
- 2. Compile OpenCV 4.2 with:



This will work, but will exclude advanced GPU features like opencv\_dnn\_cuda.

# TOPTION 2: Skip CUDA (if GPU is too weak)

Use OpenCV 4.2 without CUDA, just:

```
bash

Copy

Copy
```

Still supports CPU-based DNN and everything else.



Component	Your System	Required	Fix
CUDA	8.0	≥10.2	Install CUDA 10.2 manually
NVIDIA Driver	390.157	≥440.x	Upgrade NVIDIA driver
GPU (GT 730)	Compute 3.5	for 10.2 only	Set CUDA_ARCH_BIN="3.5"
OpenCV 4.2 DNN GPU	X Not supported	Needs CC ≥5.3	Use CPU DNN instead

Would you like a step-by-step guide to upgrade your driver and install CUDA 10.2?

```
日内ののかのいく
```

```
cmake -D CMAKE_BUILD_TYPE=Release \
    -D CMAKE_INSTALL_PREFIX=/usr/local \
    -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib/modules \
    -D BUILD_EXAMPLES=ON .. ok compile for cmake -D
CMAKE_BUILD_TYPE=Release \
    -D CMAKE_INSTALL_PREFIX=/usr/local \
```

```
-D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib/modules
  -D BUILD_EXAMPLES=ON .. ★ Summary
Component Your System Required Fix
CUDA 8.0 ≥10.2 Install CUDA 10.2 manually
NVIDIA Driver 390.157 ≥440.x Upgrade NVIDIA driver
GPU (GT 730) Compute 3.5  only Set
CUDA ARCH BIN="3.5"
OpenCV 4.2 DNN GPU X Not supported Needs CC ≥5.3 Use
CPU DNN instead [ 47%] Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/t
est_detectors_invariance.cpp.o
[47%] Building CXX object
modules/dnn/CMakeFiles/opencv_test_dnn.dir/test/test_onnx_impo
rter.cpp.o
[47%] Building CXX object
modules/features2d/CMakeFiles/opencv test features2d.dir/test/t
est detectors regression.cpp.o
[47%] Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/t
est drawing.cpp.o
[47%] Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/t
est_fast.cpp.o
[47%] Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/t
est keypoints.cpp.o
[47%] Building CXX object
modules/features2d/CMakeFiles/opencv test features2d.dir/test/t
est main.cpp.o
[47%] Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/t
est matchers algorithmic.cpp.o
[47%] Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/t
est_mser.cpp.o
[47%] Building CXX object
modules/dnn/CMakeFiles/opencv_test_dnn.dir/test/test_tf_importe
r.cdd.o
[47%] Building CXX object
modules/dnn/CMakeFiles/opencv_test_dnn.dir/test/test_torch_imp
[47%] Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/t
est_nearestneighbors.cpp.o
[47%] Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/t
est orb.cpp.o
[47%] Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/t
est utils.cpp.o
[ 47%] Linking CXX executable ../../bin/opencv_test_features2d
[ 47%] Linking CXX executable ../../bin/opencv_test_dnn
[ 47%] Built target opencv_test_features2d
[ 47%] Built target opencv_test_dnn
Makefile:162: recipe for target 'all' failed
make: *** [all] Error 2
(base) sakthees@sakthees-monk:~/opencv/build$
```

You're building OpenCV 4.2 **without CUDA now**, which is correct for your current setup — but it's still **failing at 47%** during the make stage.



- It's building and linking opencv\_test\_features2d and opencv\_test\_dnn
- It passed half the modules already
- CUDA is **not** involved now cleaner build path

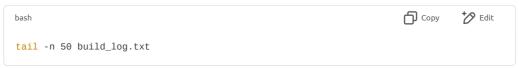


make -j\$(nproc) 2>&1 | tee build\_log.txt After it fails, do:

\* Edit

Copy

bash



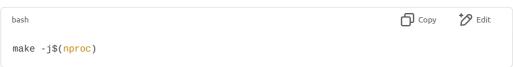
Then share the output here — it'll include the real compiler/linker error just before make: \*\*\* [all] Error 2.

#### In the Meantime: Skip Tests (Optional)

If the problem is in test modules (which it appears to be), you can build OpenCV without test targets:



#### Then re-run:



This skips test suites like opencv\_test\_dnn which are not needed unless you're developing or validating OpenCV itself.

Would you like me to help interpret the tail output once you grab it from  $build_log.txt$ ?



(base) sakthees@sakthees-monk:~/opencv/build\$ sudo ldconfig /sbin/ldconfig.real: /usr/local/cuda-8.0/targets/x86\_64linux/lib/libcudnn.so.5 is not a symbolic link

#### General configuration for OpenCV 4.11.0

\_\_\_\_\_

Version control: 4.11.0-dirty

Platform:

Timestamp: 2025-01-16T09:53:30Z Host: Linux 5.15.0-1074-azure x86\_64

CMake: 3.31.1

CMake generator: Unix Makefiles
CMake build tool: /bin/gmake
Configuration: Release

Algorithm Hint: ALGO\_HINT\_ACCURATE

CPU/HW features:

Baseline: SSE SSE2 SSE3 requested: SSE3

Dispatched code generation: SSE4\_1 SSE4\_2 AVX FP16 AVX2

AVX512\_SKX

SSE4\_1 (16 files): + SSSE3 SSE4\_1

SSE4\_2 (1 files): + SSSE3 SSE4\_1 POPCNT SSE4\_2
AVX (8 files): + SSSE3 SSE4\_1 POPCNT SSE4\_2 AVX
FP16 (0 files): + SSSE3 SSE4\_1 POPCNT SSE4\_2 AVX FP16
AVX2 (36 files): + SSSE3 SSE4\_1 POPCNT SSE4\_2 AVX FP16

AVX2 FMA3

AVX512\_SKX (5 files): + SSSE3 SSE4\_1 POPCNT SSE4\_2 AVX FP16 AVX2 FMA3 AVX\_512F AVX512\_COMMON AVX512\_SKX

C/C++:

Built as dynamic libs?: NO C++ standard: 11

C++ Compiler: /opt/rh/devtoolset-10/root/usr/bin/c++ (ver

10.2.1)

C++ flags (Release): -Wl,-strip-all -fsigned-char -W -Wall - Wreturn-type -Wnon-virtual-dtor -Waddress -Wsequence-point - Wformat -Wformat-security -Wmissing-declarations -Wundef -Winitself -Wpointer-arith -Wshadow -Wsign-promo -Wuninitialized - Wsuggest-override -Wno-delete-non-virtual-dtor -Wno-comment - Wimplicit-fallthrough=3 -Wno-strict-overflow -fdiagnostics-show-option -pthread -fomit-frame-pointer -ffunction-sections -fdata-sections -msse3 -fvisibility=hidden -fvisibility-inlines-hidden -O3 - DNDEBUG -DNDEBUG

C++ flags (Debug): -WI,-strip-all -fsigned-char -W -Wall - Wreturn-type -Wnon-virtual-dtor -Waddress -Wsequence-point - Wformat -Wformat-security -Wmissing-declarations -Wundef -Winit-self -Wpointer-arith -Wshadow -Wsign-promo -Wuninitialized - Wsuggest-override -Wno-delete-non-virtual-dtor -Wno-comment - Wimplicit-fallthrough=3 -Wno-strict-overflow -fdiagnostics-show-option -pthread -fomit-frame-pointer -ffunction-sections -fdata-sections -msse3 -fvisibility=hidden -fvisibility-inlines-hidden -g -O0 - DDEBUG -D\_DEBUG

C Compiler: /opt/rh/devtoolset-10/root/usr/bin/cc
C flags (Release): -Wl,-strip-all -fsigned-char -W -Wall Wreturn-type -Waddress -Wsequence-point -Wformat -Wformatsecurity -Wmissing-declarations -Wmissing-prototypes -Wstrictprototypes -Wundef -Winit-self -Wpointer-arith -Wshadow Wuninitialized -Wno-comment -Wimplicit-fallthrough=3 -Wno-strictoverflow -fdiagnostics-show-option -pthread -fomit-frame-pointer ffunction-sections -fdata-sections -msse3 -fvisibility=hidden -O3 DNDEBUG -DNDEBUG

C flags (Debug): -WI,-strip-all -fsigned-char -W -Wall - Wreturn-type -Waddress -Wsequence-point -Wformat -Wformat-security -Wmissing-declarations -Wmissing-prototypes -Wstrict-prototypes -Wundef -Winit-self -Wpointer-arith -Wshadow - Wuninitialized -Wno-comment -Wimplicit-fallthrough=3 -Wno-strict-overflow -fdiagnostics-show-option -pthread -fomit-frame-pointer -

ffunction-sections -fdata-sections -msse3 -fvisibility=hidden -g -O0 - DDEBUG -D\_DEBUG

Linker flags (Release): -Wl,--exclude-libs,libippicv.a -Wl,--exclude-libs,libippiw.a -L/ffmpeg\_build/lib -Wl,--gc-sections -Wl,--as-needed -Wl,--no-undefined

Linker flags (Debug): -Wl,--exclude-libs,libippicv.a -Wl,--exclude-libs,libippiw.a -L/ffmpeg\_build/lib -Wl,--gc-sections -Wl,--as-needed -

Wl,--no-undefined

ccache: YES

Precompiled headers: NO

Extra dependencies: /lib64/libopenblas.so Qt5::Core Qt5::Gui Qt5::Widgets Qt5::Test Qt5::Concurrent /usr/local/lib/libpng.so /usr/lib64/libz.so dl m pthread rt

3rdparty dependencies: libprotobuf ade ittnotify libjpeg-turbo libwebp libtiff libopenjp2 IlmImf ippiw ippicv

#### OpenCV modules:

To be built: calib3d core dnn features2d flann gapi highgui imgcodecs imgproc ml objdetect photo python3 stitching video videoio

Disabled: world
Disabled by dependency:

Unavailable: java python2 ts

Applications:

Documentation: NO
Non-free algorithms: NO

GUI: QT5

QT: YES (ver 5.15.16)
QT OpenGL support: NO

GTK+: NO VTK support: NO

#### Media I/O:

ZLib: /usr/lib64/libz.so (ver 1.2.7)

JPEG: build-libjpeg-turbo (ver 3.0.3-70)

SIMD Support Request: YES SIMD Support: YES

WEBP: build (ver decoder: 0x0209, encoder: 0x020f,

demux: 0x0107)

AVIF: NO

PNG: /usr/local/lib/libpng.so (ver 1.6.44)

TIFF: build (ver 42 - 4.6.0)

JPEG 2000: build (ver 2.5.0)

OpenEXR: build (ver 2.3.0)

GIF: NO
HDR: YES
SUNRASTER: YES
PXM: YES
PFM: YES

#### Video I/O:

FFMPEG: YES

avcodec: YES (59.37.100)
avformat: YES (59.27.100)
avutil: YES (57.28.100)
swscale: YES (6.7.100)
avresample: NO

GStreamer: NO

v4l/v4l2: YES (linux/videodev2.h)

Parallel framework: pthreads

Trace: YES (with Intel ITT)

Other third-party libraries:

Intel IPP: 2021.12.0 [2021.12.0]

at: /io/\_skbuild/linux-x86\_64-3.9/cmake-

build/3rdparty/ippicv/ippicv\_lnx/icv
Intel IPP IW: sources (2021.12.0)

at: /io/\_skbuild/linux-x86\_64-3.9/cmake-

build/3rdparty/ippicv/ippicv\_lnx/iw

VA: NO

Lapack: YES (/lib64/libopenblas.so)

Eigen: NO
Custom HAL: NO
Protobuf: build (3.19.1)

Flatbuffers: builtin/3rdparty (23.5.9)

OpenCL: YES (no extra features)

Include path: /io/opencv/3rdparty/include/opencl/1.2

Link libraries: Dynamic load

Python 3:

Interpreter: /opt/python/cp39-cp39/bin/python3.9 (ver

3.9.20)

Libraries: libpython3.9m.a (ver 3.9.20)
Limited API: YES (ver 0x03060000)

numpy: /home/ci/.local/lib/python3.9/site-

packages/numpy/\_core/include (ver 2.0.2) install path: python/cv2/python-3

Python (for build): /opt/python/cp39-cp39/bin/python3.9

Java:

ant: NO
Java: NO
JNI: NO
Java wrappers: NO
Java tests: NO

Install to: /io/\_skbuild/linux-x86\_64-3.9/cmake-install

\_\_\_\_\_

Thanks for sharing the detailed build and system info. Here's a breakdown and your next steps:

# Diagnosis Summary

#### 1. CMake Config Success V

- You're configuring OpenCV 4.11.0 with contrib modules, Qt5 GUI, FFmpeg, and Python 3 bindings.
- CUDA is **not being used**, which is good given your hardware and driver limitations.

#### 2. Build Failure (from earlier) X

- make is failing at 47%, likely during one of the test modules, since it mentioned opencv\_test\_dnn , etc.
- Your current system includes CUDA 8.0, which is outdated, but not currently being used, so the
  error is not from CUDA.
- 3. sudo ldconfig warning 1



This is just a warning. It won't break OpenCV. But to clean it up:

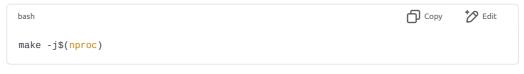


# Recommended Fixes

#### 1. X Avoid building test targets (probably causing the error)

Re-run cmake with testing disabled:





#### 2. Clean Previous Build First

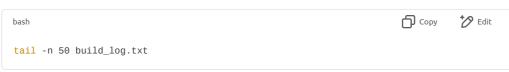
If the build fails again, try a full clean first:



# 3. $\bigcirc$ If Still Failing: Capture Error Details

If it still fails, do:





Paste that output here so I can pinpoint the error.

Would you like help removing CUDA 8.0 cleanly from your system if you're not using it anymore?



cv2.so where to change

If you're asking **where to change or find** cv2.so (the OpenCV Python binding), here's a detailed breakdown depending on your intent:

# @ 1. If You're Asking "Where is cv2.so Installed?"

After you build OpenCV with Python bindings (BUILD\_opencv\_python3=0N), cv2.so will be generated and installed to a location like:

<b>*</b> Edit	Сору
	_ р сору

# 🔧 2. If You Want to Use a Custom cv2.so or Change Python to Use It

If you built OpenCV manually and want your Python to use the custom cv2.so instead of the system one:

#### Option A: Add your custom path to PYTHONPATH:

bash	Сору	* Edit
export PYTHONPATH=/your/custom/path/to/cv2/python-3:\$PYTHONPATH		

Or make it permanent in ~/.bashrc.

#### Option B: Copy cv2.so directly

If you know where cv2.so was built (e.g., in your build/python/cv2/ directory), you can copy it manually:



(Adjust path depending on Python version and system)

#### 3. If You Want to Modify the C++ Source That Affects cv2.so

#### Then:

- Modify files in modules/python/src2/ or specific OpenCV modules like modules/imgproc/src/, etc.
- · Rebuild using CMake and make.

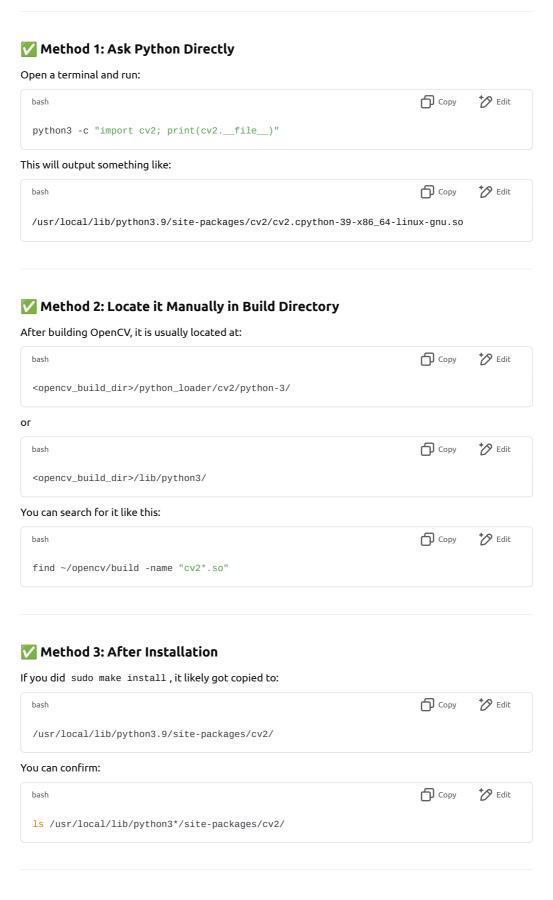
After the new .so is built, repeat the install step to update the Python bindings.

Would you like me to locate the cv2.so generated from your current OpenCV 4.11.0 build directory?



<install path>/lib/python3.x/site-packages/cv2/cv2.so where is this

To find where the cv2.so (OpenCV Python binding) is located on your system after building from source, you can follow one of these steps:



Let me know if you'd like to copy or relocate cv2.so to a specific Python environment (like virtualenv or conda).

日の句のかのいく

/home/sakthees/anaconda3/lib/python3.12/sitepackages/cv2/ init .py how to put inside environment

To properly install OpenCV ( cv2.so ) into your Anaconda environment, you have two clean options depending on how you've built OpenCV:

# Option 1: If You Built OpenCV from Source You need to copy the built cv2.so (e.g., cv2.cpython-312-x86\_64-linux-gnu.so) into your Anaconda environment: Step 1: Locate your built .so file Run: Copy \* Edit bash find ~/opencv/build -name "cv2\*.so" You'll likely get something like: \* Edit Copy bash ~/opencv/build/lib/python3/cv2.cpython-312-x86\_64-linux-gnu.so Step 2: Copy into Anaconda environment Now move that file into the site-packages/cv2/ directory of your Anaconda environment: **介** Copy \* Edit cp ~/opencv/build/lib/python3/cv2\*.so /home/sakthees/anaconda3/lib/python3.12/sitepackages/cv2/ If cv2/ doesn't exist, create it: \* Edit bash Copy mkdir -p /home/sakthees/anaconda3/lib/python3.12/site-packages/cv2 Option 2: Use Python Wheels ( .whl ) for Cleaner Install (Recommended) If you want a cleaner and reproducible install: 1. Inside your OpenCV build directory (e.g., ~/opencv/build), run: \* Edit Сору bash cd ~/opencv/build make install # if not done already cd ../python\_loader # or where your Python bindings were built 2. Build a wheel: **⁺⊘** Edit Copy bash cd ../opencv/python python3 setup.py bdist\_wheel 3. Install the wheel into your Anaconda environment: **⁺⊘** Edit Copy bash

pip install dist/opencv\_python-\*.whl

# Verify Installation

In your Anaconda environment, check:

bash

python -c "import cv2; print(cv2.\_\_file\_\_)"

bash

python -c "import cv2; print(cv2.\_\_file\_\_)"

Would you like me to help you create the .whl file from your OpenCV build?

日の句のかのいく

cv2\*.so/home/sakthees/anaconda3/lib/python3.12/site-packages/cv2/

cp: cannot stat '/home/sakthees/opencv/build/lib/python3/cv2\*.so':

No such file or directory

(base) sakthees@sakthees-monk:~/opencv/build/lib\$ pwd

/home/sakthees/opencv/build/lib

(base) sakthees@sakthees-monk:~/opencv/build/lib\$ ls

cv2.so libopencv\_ml.so libade.a libopencv\_ml.so.4.2 libopencv\_aruco.so libopencv\_ml.so.4.2.0 libopencv\_aruco.so.4.2 libopencv\_objdetect.so libopency aruco.so.4.2.0 libopency objdetect.so.4.2 libopencv\_objdetect.so.4.2.0 libopencv\_bgsegm.so libopencv\_optflow.so libopencv\_bgsegm.so.4.2 libopencv\_bgsegm.so.4.2.0 libopencv\_optflow.so.4.2 libopencv\_bioinspired.so libopencv\_optflow.so.4.2.0

libopencv\_bioinspired.so.4.2 libopencv\_phase\_unwrapping.so libopencv\_bioinspired.so.4.2.0

libopencv\_phase\_unwrapping.so.4.2

libopencv\_calib3d.so libopencv\_phase\_unwrapping.so.4.2.0

libopencv\_calib3d.so.4.2 libopencv\_photo.so libopencv\_calib3d.so.4.2.0 libopencv\_photo.so.4.2 libopencv\_ccalib.so libopencv\_photo.so.4.2.0 libopencv\_ccalib.so.4.2 libopencv\_plot.so libopencv\_ccalib.so.4.2.0 libopencv\_plot.so.4.2 libopencv\_core.so libopencv\_plot.so.4.2.0 libopencv\_core.so.4.2 libopencv\_quality.so libopencv\_quality.so.4.2 libopencv\_core.so.4.2.0

libopencv\_datasets.so libopencv\_quality.so.4.2.0 libopencv\_datasets.so.4.2 libopencv\_reg.so libopencv\_datasets.so.4.2.0 libopencv\_reg.so.4.2 libopencv\_reg.so.4.2.0 libopencv\_dnn\_objdetect.so.4.2 libopencv\_rgbd.so libopencv\_dnn\_objdetect.so.4.2.0 libopencv\_rgbd.so.4.2.0

libopencv\_dnn.so libopencv\_rgbd.so.4.2.0 libopencv\_rgbd.so.4.2.0 libopencv\_dnn.so.4.2 libopencv\_saliency.so libopencv\_dnn.so.4.2.0 libopencv\_saliency.so.4.2 libopencv\_saliency.so.4.2 libopencv\_saliency.so.4.2.0 libopencv\_saliency.so.4.2.0 libopencv\_dnn\_superres.so.4.2 libopencv\_shape.so libopencv\_dnn\_superres.so.4.2.0 libopencv\_shape.so.4.2.0 libopencv\_dpm.so libopencv\_shape.so.4.2.0

libopencv\_dpm.so.4.2 libopencv\_stereo.so libopencv\_face.so libopencv\_face.so.4.2 libopencv\_stereo.so.4.2.0 libopencv\_face.so.4.2 libopencv\_stereo.so.4.2.0 libopencv\_stereo.so.4.2.0 libopencv\_stereo.so.4.2.0

libopencv face.so.4.2.0 libopency stitching.so.4.2 libopencv\_features2d.so libopencv\_stitching.so.4.2.0 libopencv\_features2d.so.4.2 libopencv\_structured\_light.so libopency features2d.so.4.2.0 libopencv\_structured\_light.so.4.2 libopencv\_flann.so libopencv\_structured\_light.so.4.2.0 libopency flann.so.4.2 libopencv superres.so libopencv flann.so.4.2.0 libopencv superres.so.4.2 libopency freetype.so libopencv superres.so.4.2.0 libopencv\_freetype.so.4.2 libopencv\_surface\_matching.so libopencv\_freetype.so.4.2.0 libopencv\_surface\_matching.so.4.2 libopency fuzzy.so libopency surface matching.so.4.2.0 libopencv\_text.so libopencv\_fuzzy.so.4.2 libopencv\_fuzzy.so.4.2.0 libopencv\_text.so.4.2 libopencv\_gapi.so libopencv\_text.so.4.2.0 libopencv\_gapi.so.4.2 libopency tracking.so libopencv\_tracking.so.4.2 libopencv\_gapi.so.4.2.0 libopency hdf.so libopency tracking.so.4.2.0 libopencv hdf.so.4.2 libopency videoio.so libopency hdf.so.4.2.0 libopencv videoio.so.4.2 libopencv\_hfs.so libopencv\_videoio.so.4.2.0 libopencv\_hfs.so.4.2 libopencv\_video.so libopency hfs.so.4.2.0 libopencv video.so.4.2 libopencv\_highgui.so libopencv\_video.so.4.2.0 libopencv\_highgui.so.4.2 libopencv\_videostab.so libopencv\_highgui.so.4.2.0 libopencv\_videostab.so.4.2 libopencv\_imgcodecs.so libopencv\_videostab.so.4.2.0 libopencv\_xfeatures2d.so libopencv\_imgcodecs.so.4.2 libopency imgcodecs.so.4.2.0 libopencv xfeatures2d.so.4.2 libopencv img hash.so libopencv xfeatures2d.so.4.2.0 libopencv\_img\_hash.so.4.2 libopencv ximgproc.so libopencv\_img\_hash.so.4.2.0 libopencv\_ximgproc.so.4.2 libopencv\_imgproc.so libopencv\_ximgproc.so.4.2.0 libopencv imgproc.so.4.2 libopency xobjdetect.so libopencv\_imgproc.so.4.2.0 libopencv\_xobjdetect.so.4.2 libopencv\_java420.so libopencv\_xobjdetect.so.4.2.0 libopencv\_line\_descriptor.so libopencv\_xphoto.so libopency line descriptor.so.4.2 libopency xphoto.so.4.2 libopencv\_line\_descriptor.so.4.2.0 libopencv\_xphoto.so.4.2.0

It seems that the cv2.so file is located in the opencv/build/lib directory as a shared object (i.e., cv2.so). However, this file is not specifically tied to Python bindings yet.

To install OpenCV properly into your Python environment (like Anaconda), follow these steps:

#### Step 1: Copy the CV2.SO file to your Anaconda environment

Since you're using Python 3.12 in Anaconda, you can copy the cv2.so to the following directory:

bash	Сору	* Edit
<pre>cp /home/sakthees/opencv/build/lib/cv2.so /home/sakthees/ana packages/cv2/</pre>	conda3/lib/python3	3.12/site-
If the cv2/ folder doesn't exist inside /home/sakthees/anaconda3/lib create it first:	/python3.12/site-p	ackages/,
bash	<b>П</b> Сору	* Edit
bash  mkdir -p /home/sakthees/anaconda3/lib/python3.12/site-packag		* Edit

#### **Step 2: Verify Installation**

Once the cv2.so file is in place, you can check if it's successfully loaded in your Python environment by running:

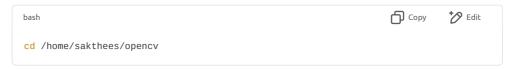


This should show the path to the cv2.so file in the site-packages/cv2/ directory.

## Alternative Method: Build Python Bindings for OpenCV

If cv2.so isn't working directly, you might need to **build Python bindings** for OpenCV to link it with your Python installation:

1. Go to the OpenCV directory:



2. Build the Python bindings by running the following:

```
mkdir -p build && cd build cmake -D CMAKE_BUILD_TYPE=Release -D
CMAKE_INSTALL_PREFIX=/home/sakthees/anaconda3 -D PYTHON_EXECUTABLE=$(which python)
.. make -j$(nproc) make install
```

This will generate and install the cv2.so bindings directly for Python. Afterward, verify using the import cv2 command again.

Let me know if this helps or if you'd like more guidance!

You're seeing version **4.11.0** when importing cv2, which means your Python environment is still using the **pre-installed or pip-installed OpenCV**, not your manually built **4.2.0** version.



#### Step-by-step fix:

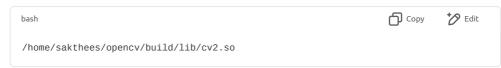
1. Remove the existing OpenCV (4.11.0) from Anaconda:

Run this to avoid conflicts:

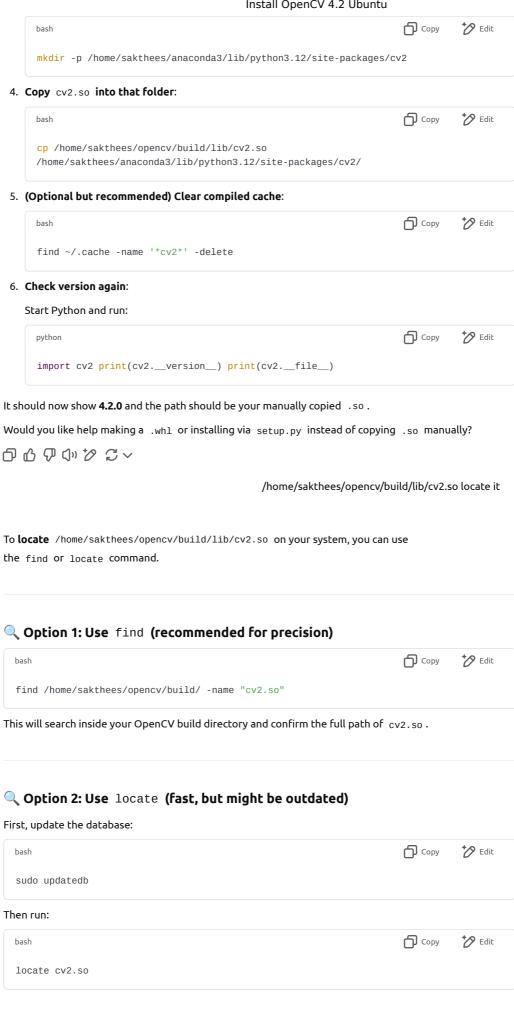


2. Locate your built cv2.so:

You already have it here:



3. Create a proper cv2 package folder in site-packages:



Since you already mentioned this earlier:

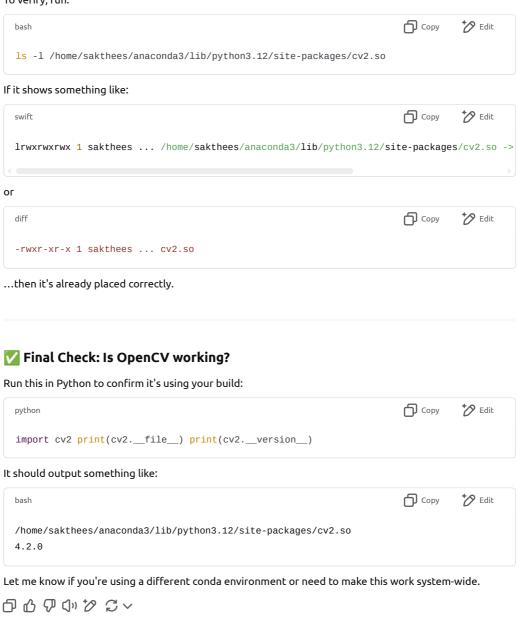
swift	Сору	* Edit
/home/sakthees/opencv/build/lib/cv2.so		
This is your OpenCV Python binding. You can <b>copy</b> or <b>symlink</b> it to your Ana	aconda site-	
Would you prefer to symlink instead of copy, so it auto-updates if rebuilt?		
D B 日 O O O C >		
/home/sakti	nees/opencv/build	d/lib/cv2.so
Great — since you have:		
swift	Сору	* Edit
/home/sakthees/opencv/build/lib/cv2.so		
you can now install this custom OpenCV build into your Anaconda environtion following:	onment by doing o	one of the
Option 1: Symlink cv2.so into your Anaconda env (red	commended)	
bash	Сору	* Edit
<pre>ln -s /home/sakthees/opencv/build/lib/cv2.so /home/sakthees/anaconda3/lib/python3.12/site-packages/cv2.so</pre>		
⚠ Make sure you're not inside the base environment unless intended. If environment, adjust the path accordingly.	f you're using a co	onda
Option 2: Copy cv2.so into site-packages		
bash	Сору	* Edit
<pre>cp /home/sakthees/opencv/build/lib/cv2.so /home/sakthees/anacor packages/</pre>	nda3/lib/python:	3.12/site-
Optional: Check it's working		
Launch Python and test:		
python	Сору	* Edit
<pre>import cv2 print(cv2version)</pre>		
f it prints the expected version (like 4.2.0), you're all set.		
Do you want to link this into a specific Conda environment instead of the ba	ase one?	
ጋ ይ ዊ ሳ େ େ ፎ <		

\$ cp /home/sakthees/opencv/build/lib/cv2.so /home/sakthees/anaconda3/lib/python3.12/site-packages/cp: '/home/sakthees/opencv/build/lib/cv2.so' and

'/home/sakthees/anaconda3/lib/python3.12/site-packages/cv2.so' are the same file

That message means you're trying to copy the file to the **same location** — likely because it's **already there** or has been symlinked before.

To verify, run:



Traceback (most recent call last):

File "<stdin>", line 1, in <module>
ImportError:
/home/sakthees/anaconda3/bin/../lib/libfontconfig.so.1: undefined
symbol: FT\_Done\_MM\_Var

This error indicates that there's a **conflict between your OpenCV build and the version of FreeType** (font rendering library) bundled with Anaconda — specifically, the libfontconfig.so.1 library is expecting a symbol (FT\_Done\_MM\_Var) from **FreeType**, but it's missing or mismatched.

>>> import cv2

>>>

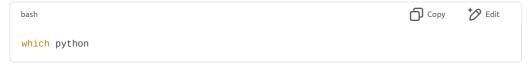


# ✓ 1. Force OpenCV to use system FreeType instead of Anaconda's

Run	vour:	script	with	svstem	libraries	prioritized:

Run your script with system in	pranies prioritized.		
bash		Сору	* Edit
LD_PRELOAD=/usr/lib/x86	_64-linux-gnu/libfreetype.so.6 python		
Or if you're inside an Anaconc	da environment:		
bash		Сору	* Edit
LD_PRELOAD=/usr/lib/x86 /home/sakthees/anaconda	_64-linux-gnu/libfreetype.so.6 3/bin/python		
Make sure that path exists usi	ing:		
bash		Сору	* Edit
ls /usr/lib/x86_64-linu	x-gnu/libfreetype.so.6		
If that works, you can export i	itinyour .bashrc:		
bash		Сору	<b>†</b> ⊘ Edit
export LD_PRELOAD=/usr/	lib/x86_64-linux-gnu/libfreetype.so.6		
✓ 2. Temporarily disable An			
Try running Python with a clea	an LD_LIBRARY_PATH:		40
bash		Сору	Edit
LD_LIBRARY_PATH="" pyth	on		
	flicting libraries from being loaded. If OpenCV rui	ns now, then	
<b>2</b> 3. Rebuild OpenCV with F  If you can recompile, try confi			
bash		Сору	* Edit
-D WITH_FREETYPE=OFF			
Then recompile and install. Th	nis will remove dependency on the conflicting Fred	eType functior	ns.
Avended on the second of the s			
Would you like me to walk you 日	u through one of these options step-by-step?		
	Python 3.6.9 (default, Mar 10 2023, 16:46:0 [GCC 8.4.0] on linux  Type "help", "copyright", "credits" or "licens" >>> import cv2 >>> print(cv2version)  4.11.0 >>> which python		formation.

It seems you are using Python 3.6.9. To check which Python executable you're currently using, run the following command in the terminal:



This will show you the path to the active Python interpreter. If you're working within a specific environment, like Anaconda or a virtualenv, it will show the path to that environment's Python executable.

Let me know what the output is!



cp -r /home/sakthees/cv2.so /usr/lib/python3.6

It seems like you're trying to copy the cv2.so file to the Python 3.6 library directory (/usr/lib/python3.6). To do this, you need to ensure that the target directory is correct and accessible.

If you intend to copy the cv2.so file to the correct location, you can run:



However, please note:

- The directory /usr/lib/python3.6 is typically for system-wide Python installations. It's better to
  place the cv2.so file inside the site-packages folder, which is where Python packages are typically
  stored for your environment.
- 2. If you're using Anaconda, it's generally better to install OpenCV using conda instead of manually copying the .so files. This will ensure compatibility with other packages.

If you still prefer to manually move the cv2.so, ensure that the target path is correct, especially the site-packages directory for your Python environment.

Let me know how you'd like to proceed!



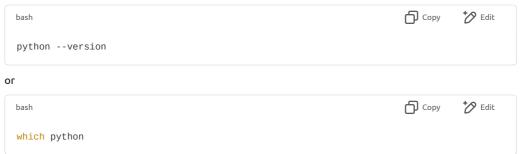
how to check

To check if your cv2.so is correctly placed and if OpenCV is working in your environment, follow these steps:

#### 1. Verify Python Version

Make sure you are using the correct Python version where you want OpenCV to work.

Run this in your terminal to check the active Python version:



This will tell you which Python version and executable are being used.

#### 2. Check Python Site-Packages Directory

You need to check where the site-packages folder is located for your specific Python environment (e.g., Anaconda, system Python, etc.).

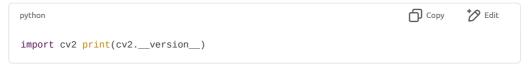
In Python, run this:



This will print the directories where Python packages are installed.

#### 3. Check if OpenCV is Installed

To check if OpenCV is correctly installed and working, run this in the Python shell or a script:



If this prints the OpenCV version (like 4.11.0), OpenCV is correctly installed.

#### 4. Verify cv2.so File Placement

To ensure that cv2.so is in the correct place, check if it is located in the appropriate site-packages directory. In Anaconda, it should typically be inside the lib/pythonX.X/site-packages/cv2/ folder (replace X.X with your version).

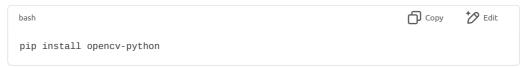
Use 1s to verify the file location:



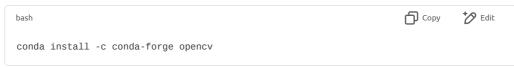
#### 5. Reinstall OpenCV via Package Manager (Optional)

If the manual installation isn't working or you face issues with dependencies, it's a good idea to install OpenCV through pip or conda.

#### For pip (if using a virtual environment):



# For conda (if using Anaconda):



Let me know if you need any more guidance!

