

**You said:**

install opencv 4.2 and contrib in ubuntu linux

ChatGPT said:

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

```
bash
CopyEdit
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
CopyEdit
cd ~
git clone -b 4.2.0 https://github.com/opencv/opencv.git
git clone -b 4.2.0 https://github.com/opencv/opencv_contrib.git
```

### 3. Create Build Directory

```
bash
CopyEdit
cd ~/opencv
mkdir build
cd build
```

### 4. Configure with CMake

```
bash
CopyEdit
cmake -D CMAKE_BUILD_TYPE=Release \
    -D CMAKE_INSTALL_PREFIX=/usr/local \
    -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib/modules \
    -D BUILD_EXAMPLES=ON ..
```

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

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

```
bash
CopyEdit
make -j$(nproc)
```

## 6. Install

```
bash
CopyEdit
sudo make install
sudo ldconfig
```

## 7. Verify Installation

Check version with Python:

```
bash
CopyEdit
python3 -c "import cv2; print(cv2.__version__)"
```

It should show 4.2.0.

---

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

40

**You said:**

```
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
```

ChatGPT said:

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:

```
bash
CopyEdit
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 ..
```

---

## Additional Setup Steps

### 1. Install CUDA and cuDNN

If you haven't already:

```
bash
CopyEdit
sudo apt install nvidia-cuda-toolkit
```

⚠ 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:

```
bash
CopyEdit
nvcc --version
```

Make sure it's detected correctly.

### 3. Optional: Set Compute Capability

To optimize for your GPU, add:

```
bash
CopyEdit
-D CUDA_ARCH_BIN="7.5" \
```

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

Example:

```
bash
CopyEdit
-D CUDA_ARCH_BIN="6.1" # For GTX 1050 Ti
```

---

### ✓ After CMake:

```
bash
CopyEdit
make -j$(nproc)
sudo make install
sudo ldconfig
```

## Test CUDA in OpenCV

```
python
CopyEdit
import cv2
print(cv2.getBuildInformation())
```

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?

40

**You said:**

```
torch_importer.cpp.o [ 47%] Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/test_nearestneighbors.cpp.o [ 47%]
Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/test_orb.cpp.o [ 47%] Building
CXX object modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/test_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 ..
```

**ChatGPT said:**

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

```
bash
CopyEdit
cd ~/opencv/build
make clean
rm -rf *
```

### 2. Double-Check Dependencies

Ensure these are installed:

```
bash
CopyEdit
sudo apt install -y cmake g++ pkg-config \
    libgtk-3-dev libavcodec-dev libavformat-dev libswscale-dev \
    libtbb2 libtbb-dev libjpeg-dev libpng-dev libtiff-dev libopenexr-dev \
    python3-dev python3-numpy libdc1394-22-dev
```

### 3. Check CUDA is Set Up

Run:

```
bash
```

```
CopyEdit
nvcc --version
nvidia-smi
```

Make sure both respond correctly and show GPU info.

---

## Optional: Set Compute Capability (Highly Recommended)

Add this to `CMake` if you're targeting a specific GPU:

```
bash
CopyEdit
-D CUDA_ARCH_BIN="6.1" # e.g., for GTX 1050 Ti
```

You can find your architecture here:

<https://developer.nvidia.com/cuda-gpus>

---

## Re-run CMake

```
bash
CopyEdit
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 CUDA_ARCH_BIN="6.1" \
      -D BUILD_EXAMPLES=ON ..          # change based on your GPU
```

---

## Compile Again

```
bash
CopyEdit
make -j$(nproc)
```

---

## If It Still Fails

1. Check the last few lines of error output (important).

2. Redirect full build log to a file:

```
bash
CopyEdit
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.

---

40

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

This means:

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

## Option 1: Stay with GT 730

1. Upgrade to CUDA 10.2 + driver  $\geq 440.x$
2. Compile OpenCV 4.2 with:

```
bash
CopyEdit
-D WITH_CUDA=ON
-D CUDA_ARCH_BIN="3.5"
```

This will work, but will **exclude** advanced GPU features like `opencv_dnn_cuda`.

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

Use OpenCV 4.2 **without CUDA**, just:

```
bash
CopyEdit
cmake -D CMAKE_BUILD_TYPE=Release \
-D CMAKE_INSTALL_PREFIX=/usr/local \
-D OPENCV_EXTRA_MODULES_PATH=~/.opencv_contrib/modules \
-D BUILD_EXAMPLES=ON ..
```

Still supports CPU-based DNN and everything else.

## Summary

Component	Your System	Required	Fix
CUDA	8.0	$\geq 10.2$	Install CUDA 10.2 manually
NVIDIA Driver	390.157	$\geq 440.x$	Upgrade NVIDIA driver
GPU (GT 730)	Compute 3.5	✓ for 10.2 only	Set <code>CUDA_ARCH_BIN="3.5"</code>
OpenCV 4.2 DNN GPU	✗ Not supported	Needs CC $\geq 5.3$	Use CPU DNN instead

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

40

**You said:**

```
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  $\geq 10.2$ 
```

```

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 ✗ Not
supported Needs CC ≥5.3 Use CPU DNN instead [ 47%] Building CXX object modules/features2d/
CMakeFiles/opencv_test_features2d.dir/test/test_detectors_invariance.cpp.o [ 47%] Building CXX
object modules/dnn/CMakeFiles/opencv_test_dnn.dir/test/test_onnx_importer.cpp.o [ 47%]
Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/test_detectors_regression.cpp.o
[ 47%] Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/test_drawing.cpp.o [ 47%]
Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/test_fast.cpp.o [ 47%] Building
CXX object modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/test_keypoints.cpp.o
[ 47%] Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/test_main.cpp.o [ 47%] Building
CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/test_matchers_algorithmic.cpp.o
[ 47%] Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/test_mser.cpp.o [ 47%] Building
CXX object modules/dnn/CMakeFiles/opencv_test_dnn.dir/test/test_tf_importer.cpp.o [ 47%]
Building CXX object modules/dnn/CMakeFiles/opencv_test_dnn.dir/test/test_torch_importer.cpp.o
[ 47%] Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/test_nearestneighbors.cpp.o [ 47%]
Building CXX object
modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/test_orb.cpp.o [ 47%] Building
CXX object modules/features2d/CMakeFiles/opencv_test_features2d.dir/test/test_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$

```

ChatGPT said:

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.

### ✓ Good Signs:

- 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

---

### ✗ But Then:

```

bash
CopyEdit
Makefile:162: recipe for target 'all' failed
make: *** [all] Error 2

```



This generic error means **some compilation failed**, but it doesn't say **which one**. We need the **last few lines before** the error to diagnose the actual failure.

---

### ✓ Try This to Get the Real Error:

Run this command to capture all errors to a file:

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

After it fails, do:

```
bash
CopyEdit
tail -n 50 build_log.txt
```

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**:

```
bash
CopyEdit
cmake -D CMAKE_BUILD_TYPE=Release \
      -D CMAKE_INSTALL_PREFIX=/usr/local \
      -D OPENCV_EXTRA_MODULES_PATH=~/.opencv_contrib/modules \
      -D BUILD_TESTS=OFF \
      -D BUILD_PERF_TESTS=OFF \
      -D BUILD_EXAMPLES=ON ..
```

Then re-run:

```
bash
CopyEdit
make -j$(nproc)
```

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`?

40

**You said:**

```
(base) sakthees@sakthees-monk:~/opencv/build$ sudo ldconfig /sbin/ldconfig.real: /usr/local/cuda-8.0/targets/x86_64-linux/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 -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 -O3 -DNDEBUG -DNDEBUG C++ flags (Debug): -Wl,-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 -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 -O3 -DNDEBUG -DNDEBUG C flags (Debug): -Wl,-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 itnotify 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 -----

ChatGPT said:

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

---

## Diagnosis Summary

### 1. CMake Config Success ✓

- 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) ✗

- 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 ⚠

```
bash
CopyEdit
/usr/local/cuda-8.0/targets/x86_64-linux/lib/libcudnn.so.5 is not a
symbolic link
```

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

```
bash
CopyEdit
sudo mv /usr/local/cuda-8.0/targets/x86_64-linux/lib/libcudnn.so.5
/usr/local/cuda-8.0/targets/x86_64-linux/lib/libcudnn.so.5.bak
sudo ln -s libcudnn.so.5.1.10
/usr/local/cuda-8.0/targets/x86_64-linux/lib/libcudnn.so.5
```

---

## ✓ Recommended Fixes

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

Re-run cmake with testing disabled:

```
bash
CopyEdit
cmake -D CMAKE_BUILD_TYPE=Release \
      -D CMAKE_INSTALL_PREFIX=/usr/local \
      -D OPENCV_EXTRA_MODULES_PATH=~/.opencv_contrib/modules \
      -D BUILD_TESTS=OFF \
      -D BUILD_PERF_TESTS=OFF \
      -D BUILD_EXAMPLES=ON ..
```

Then build again:

```
bash
CopyEdit
make -j$(nproc)
```

---

### 2. Clean Previous Build First

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

```
bash
CopyEdit
cd ~/opencv
rm -rf build
mkdir build
cd build
# Then rerun the cmake above
```

---

### 3. If Still Failing: Capture Error Details

If it still fails, do:

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

Then:

```
bash
CopyEdit
tail -n 50 build_log.txt
```

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?

40

**You said:**

cv2.so where to change

ChatGPT said:

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=ON), cv2 . so will be generated and installed to a location like:

```
bash
CopyEdit
<install path>/lib/python3.x/site-packages/cv2/cv2.so
```

To check where it's currently installed:

```
bash
CopyEdit
python3 -c "import cv2; print(cv2.__file__)"
```

---

### 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
CopyEdit
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:

```
bash
CopyEdit
sudo cp build/python/cv2/cv2*.so /usr/lib/python3.x/site-packages/cv2/
```

(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?

40

**You said:**

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

ChatGPT said:

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:

---

### ✓ **Method 1: Ask Python Directly**

Open a terminal and run:

```
bash
CopyEdit
python3 -c "import cv2; print(cv2.__file__)"
```

This will output something like:

```
bash
CopyEdit
/usr/local/lib/python3.9/site-packages/cv2/cv2.cpython-39-x86_64-linux-gnu.so
```

---

### ✓ **Method 2: Locate it Manually in Build Directory**

After building OpenCV, it is usually located at:

```
bash
CopyEdit
<opencv_build_dir>/python_loader/cv2/python-3/
```

or

```
bash
CopyEdit
<opencv_build_dir>/lib/python3/
```

You can search for it like this:

```
bash
CopyEdit
find ~/opencv/build -name "cv2*.so"
```

---

### ✓ **Method 3: After Installation**

If you did `sudo make install`, it likely got copied to:

```
bash
CopyEdit
/usr/local/lib/python3.9/site-packages/cv2/
```

You can confirm:

```
bash
CopyEdit
ls /usr/local/lib/python3*/site-packages/cv2/
```

---

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

40

**You said:**

`/home/sakthees/anaconda3/lib/python3.12/site-packages/cv2/__init__.py` how to put inside environment

ChatGPT said:

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:

```
bash
CopyEdit
find ~/opencv/build -name "cv2*.so"
```

You'll likely get something like:

```
bash
CopyEdit
~/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:

```
bash
CopyEdit
cp ~/opencv/build/lib/python3/cv2*.so
/home/sakthees/anaconda3/lib/python3.12/site-packages/cv2/
```

If `cv2/` doesn't exist, create it:

```
bash
```

```
CopyEdit
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:

```
bash
CopyEdit
cd ~/opencv/build
make install # if not done already
cd ../python_loader # or where your Python bindings were built
```

2. Build a wheel:

```
bash
CopyEdit
cd ../opencv/python
python3 setup.py bdist_wheel
```

3. Install the wheel into your Anaconda environment:

```
bash
CopyEdit
pip install dist/opencv_python-*.whl
```

---

## ✓ Verify Installation

In your Anaconda environment, check:

```
bash
CopyEdit
python -c "import cv2; print(cv2.__file__)"
```

---

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

4o

**You said:**

```
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 libopencv_aruco.so.4.2.0 libopencv_objdetect.so.4.2 libopencv_bgsegm.so
libopencv_objdetect.so.4.2.0 libopencv_bgsegm.so.4.2 libopencv_optflow.so
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\_core.so.4.2.0 libopencv\_quality.so.4.2 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\_dnn\_objdetect.so 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 libopencv\_dnn.so  
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\_dnn\_superres.so 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 libopencv\_dpm.so libopencv\_shape.so.4.2.0 libopencv\_dpm.so.4.2  
libopencv\_stereo.so libopencv\_dpm.so.4.2.0 libopencv\_stereo.so.4.2 libopencv\_face.so  
libopencv\_stereo.so.4.2.0 libopencv\_face.so.4.2 libopencv\_stitching.so libopencv\_face.so.4.2.0  
libopencv\_stitching.so.4.2 libopencv\_features2d.so libopencv\_stitching.so.4.2.0  
libopencv\_features2d.so.4.2 libopencv\_structured\_light.so libopencv\_features2d.so.4.2.0  
libopencv\_structured\_light.so.4.2 libopencv\_flann.so libopencv\_structured\_light.so.4.2.0  
libopencv\_flann.so.4.2 libopencv\_superres.so libopencv\_flann.so.4.2.0 libopencv\_superres.so.4.2  
libopencv\_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  
libopencv\_fuzzy.so libopencv\_surface\_matching.so.4.2.0 libopencv\_fuzzy.so.4.2 libopencv\_text.so  
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 libopencv\_tracking.so libopencv\_gapi.so.4.2.0 libopencv\_tracking.so.4.2  
libopencv\_hdf.so libopencv\_tracking.so.4.2.0 libopencv\_hdf.so.4.2 libopencv\_videoio.so  
libopencv\_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 libopencv\_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\_imgcodecs.so.4.2 libopencv\_xfeatures2d.so  
libopencv\_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 libopencv\_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  
libopencv\_line\_descriptor.so.4.2 libopencv\_xphoto.so.4.2 libopencv\_line\_descriptor.so.4.2.0  
libopencv\_xphoto.so.4.2.0

ChatGPT said:

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
CopyEdit
cp /home/sakthees/opencv/build/lib/cv2.so
/home/sakthees/anaconda3/lib/python3.12/site-packages/cv2/
```

If the `cv2/` folder doesn't exist inside

`/home/sakthees/anaconda3/lib/python3.12/site-packages/`, create it first:

```
bash
CopyEdit
mkdir -p /home/sakthees/anaconda3/lib/python3.12/site-packages/cv2/
```

Then move the `cv2.so` file into that folder.

---

## 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:

```
bash
CopyEdit
python -c "import cv2; print(cv2.__file__)"
```

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:

```
bash
CopyEdit
cd /home/sakthees/opencv
```

2. Build the Python bindings by running the following:

```
bash
CopyEdit
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 said:**

```
>>> print(cv2.__version__) 4.11.0
/home/sakthees/anaconda3/lib/python3.12/site-packages/cv2/cv2.so 4.2.0 no
```

**ChatGPT said:**

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.

## **To make Python use your custom-built OpenCV 4.2.0:**

### ✓ **Step-by-step fix:**

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

Run this to avoid conflicts:

```
bash
CopyEdit
pip uninstall opencv-python
pip uninstall opencv-contrib-python
```

#### **2. Locate your built cv2 . so:**

You already have it here:

```
bash
CopyEdit
/home/sakthees/opencv/build/lib/cv2.so
```

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

```
bash
CopyEdit
mkdir -p /home/sakthees/anaconda3/lib/python3.12/site-packages/cv2
```

#### **4. Copy cv2 . so into that folder:**

```
bash
CopyEdit
cp /home/sakthees/opencv/build/lib/cv2.so
/home/sakthees/anaconda3/lib/python3.12/site-packages/cv2/
```

#### **5. (Optional but recommended) Clear compiled cache:**

```
bash
CopyEdit
find ~/.cache -name '*cv2*' -delete
```

#### **6. Check version again:**

Start Python and run:

```
python
CopyEdit
import cv2
print(cv2.__version__)
print(cv2.__file__)
```

It should now show **4.2.0** and the path should be your manually copied . so.

Would you like help making a `.whl` or installing via `setup.py` instead of copying `.so` manually?

40

**You said:**

`/home/sakthees/opencv/build/lib/cv2.so` locate it

ChatGPT said:

To **locate** `/home/sakthees/opencv/build/lib/cv2.so` on your system, you can use the `find` or `locate` command.

---

### Option 1: Use `find` (recommended for precision)

```
bash
CopyEdit
find /home/sakthees/opencv/build/ -name "cv2.so"
```

This will search inside your OpenCV build directory and confirm the full path of `cv2.so`.

---

### Option 2: Use `locate` (fast, but might be outdated)

First, update the database:

```
bash
CopyEdit
sudo updatedb
```

Then run:

```
bash
CopyEdit
locate cv2.so
```

---

Since you already mentioned this earlier:

```
swift
CopyEdit
/home/sakthees/opencv/build/lib/cv2.so
```

This is your OpenCV Python binding. You can **copy** or **symlink** it to your Anaconda `site-packages/cv2/` folder to use it.

Would you prefer to symlink instead of copy, so it auto-updates if rebuilt?

40

**You said:**

`/home/sakthees/opencv/build/lib/cv2.so`

ChatGPT said:

Great — since you have:

```
swift
CopyEdit
/home/sakthees/opencv/build/lib/cv2.so
```

you can now **install this custom OpenCV build into your Anaconda environment** by doing one of the following:

---

### ✓ **Option 1: Symlink cv2 . so into your Anaconda env (recommended)**

This avoids manual copying and keeps things clean.

```
bash
CopyEdit
ln -s /home/sakthees/opencv/build/lib/cv2.so
/home/sakthees/anaconda3/lib/python3.12/site-packages/cv2.so
```

Make sure you're not inside the base environment unless intended. If you're using a conda environment, adjust the path accordingly.

---

### ✓ **Option 2: Copy cv2 . so into site-packages**

```
bash
CopyEdit
cp /home/sakthees/opencv/build/lib/cv2.so
/home/sakthees/anaconda3/lib/python3.12/site-packages/
```

---

### **Optional: Check it's working**

Launch Python and test:

```
python
CopyEdit
import cv2
print(cv2.__version__)
```

If 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 base one?

40

**You said:**

```
$ 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
```

ChatGPT said:

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:

```
bash
CopyEdit
ls -l /home/sakthees/anaconda3/lib/python3.12/site-packages/cv2.so
```

If it shows something like:

```
swift
CopyEdit
lrwxrwxrwx 1 sakthees ... /home/sakthees/anaconda3/lib/python3.12/site-packages/
cv2.so -> /home/sakthees/opencv/build/lib/cv2.so
```

or

```
diff
CopyEdit
-rwxr-xr-x 1 sakthees ... cv2.so
```

...then it's already placed correctly.

---

### ✓ **Final Check: Is OpenCV working?**

Run this in Python to confirm it's using your build:

```
python
CopyEdit
import cv2
print(cv2.__file__)
print(cv2.__version__)
```

It should output something like:

```
bash
CopyEdit
/home/sakthees/anaconda3/lib/python3.12/site-packages/cv2.so
4.2.0
```

Let me know if you're using a different conda environment or need to make this work system-wide.

40

**You said:**

```
>>> import cv2 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 >>>
```

ChatGPT said:

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.

---

## Solution Options

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

Run your script with system libraries prioritized:

```
bash
CopyEdit
LD_PRELOAD=/usr/lib/x86_64-linux-gnu/libfreetype.so.6 python
```

Or if you're inside an Anaconda environment:

```
bash
CopyEdit
LD_PRELOAD=/usr/lib/x86_64-linux-gnu/libfreetype.so.6
/home/sakthees/anaconda3/bin/python
```

Make sure that path exists using:

```
bash
CopyEdit
ls /usr/lib/x86_64-linux-gnu/libfreetype.so.6
```

If that works, you can export it in your `.bashrc`:

```
bash
CopyEdit
export LD_PRELOAD=/usr/lib/x86_64-linux-gnu/libfreetype.so.6
```

---

### 2. Temporarily disable Anaconda libraries

Try running Python with a clean `LD_LIBRARY_PATH`:

```
bash
CopyEdit
LD_LIBRARY_PATH="" python
```

This prevents Anaconda's conflicting libraries from being loaded. If OpenCV runs now, then Anaconda's `libfreetype` or `libfontconfig` are the culprits.

---

### 3. Rebuild OpenCV with FreeType OFF (optional)

If you can recompile, try configuring OpenCV with:

```
bash
CopyEdit
-D WITH_FREETYPE=OFF
```

Then recompile and install. This will remove dependency on the conflicting FreeType functions.

---

Would you like me to walk you through one of these options step-by-step?

**You said:**

Python 3.6.9 (default, Mar 10 2023, 16:46:00) [GCC 8.4.0] on linux Type "help", "copyright", "credits" or "license" for more information. >>> import cv2 >>> print(cv2.\_\_version\_\_) 4.11.0 >>> which python

ChatGPT said:

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:

```
bash
CopyEdit
which python
```

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!

4o mini

**You said:**

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

ChatGPT said:

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:

```
bash
CopyEdit
sudo cp /home/sakthees/opencv/build/lib/cv2.so /usr/lib/python3.6/
```

However, please note:

1. 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!

4o mini

**You said:**

how to check



ChatGPT said:

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:

```
bash
CopyEdit
python --version
```

or

```
bash
CopyEdit
which python
```

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:

```
python
CopyEdit
import site
print(site.getsitepackages())
```

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:

```
python
CopyEdit
import cv2
print(cv2.__version__)
```

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 `ls` to verify the file location:

```
bash
CopyEdit
ls /path/to/your/python/site-packages/cv2/
```

## 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):

```
bash
CopyEdit
pip install opencv-python
```

### For conda (if using Anaconda):

```
bash
CopyEdit
conda install -c conda-forge opencv
```

Let me know if you need any more guidance!

4o mini