OpenCV4 installation without words

Ubuntu 22.04 LTS Intel Core i5 CPU 2,20GHz x 4 16GiB RAM

! ! ! commands in red and directories in blue ! ! !

```
prog@BF:~$ sudo apt-get update ⏎

prog@BF:~$ sudo apt-get install build-essential ⏎

prog@BF:~$ sudo apt-get install cmake git libgtk2.0-dev pkg-config ⏎

prog@BF:~$ sudo apt-get libavcodec-dev libavformat-dev libswscale-dev ⏎

prog@BF:~$ mkdir ~/src ⏎

prog@BF:~$ cd ~/src ⏎

prog@BF:~/src$ git clone https://github.com/opencv/opencv.git ⏎

prog@BF:~/src$ cd opencv ⏎

prog@BF:~/src/opencv$ mkdir build && cd build ⏎

prog@BF:~/src/opencv/build$ cmake -D CMAKE_BUILD_TYPE=RELEASE ⏎
> -D CMAKE_INSTALL_PREFIX=/usr/local ⏎
> -D INSTALL_PYTHON_EXAMPLES=ON ⏎
> -D INSTALL_C_EXAMPLES=ON .. ⏎

prog@BF:~/src/opencv/build$ make -j$(nproc) ⏎

prog@BF:~/src/opencv/build$ sudo make install ⏎

prog@BF:~/src/opencv/build$ pkg-config --cflags opencv4 ⏎

prog@BF:~/src/opencv/build$ pkg-config --libs opencv4 ⏎


------------------ TEST INSTALLATION ---------------------


prog@BF:~/src/opencv/build$ cd ~/src/opencv/samples ⏎

prog@BF:~/src/opencv/samples$ cmake . ⏎

prog@BF:~/src/opencv/samples$ make ⏎
```

---------- TEST OPENCV C++ VERSION ----------

```
// This file is part of OpenCV project.
// It is subject to the license terms in the LICENSE file found in the
// top-level directory
// of this distribution and at http://opencv.org/license.html

//opencv_version.cpp
#include <opencv2/core/utility.hpp>
#include <iostream>

static const std::string keys = "{ b build | | print complete build info }"
"{ h help | | print this help }";

int main(int argc, const char* argv[])
{
cv::CommandLineParser parser(argc, argv, keys);
parser.about("This sample outputs OpenCV version and build configuration.");
if (parser.has("help"))
{
parser.printMessage();
}
else if (!parser.check())
{
parser.printErrors();
}
else if (parser.has("build"))
{
std::cout << cv::getBuildInformation() << std::endl;
}
else
{
std::cout << "OpenCV version is " << CV_VERSION << std::endl;
}
return 0;
}
```

```
prog@BF:~/0_OpenCV4/my_opencv_version$ ls ⏎
opencv_version.cpp

prog@BF:~/0_OpenCV4/my_opencv_version$ g++ -ggdb opencv_version.cpp \
 -o opencv_version `pkg-config --cflags --libs opencv4` ⏎

prog@BF:~/0_OpenCV4/my_opencv_version$ ls ⏎
opencv_version  opencv_version.cpp

prog@BF:~/0_OpenCV4/my_opencv_version$ ./opencv_version  ⏎
OpenCV version is 4.5.4
```

```
------------------- TEST PYTHON3 BINDINGS ----------------

prog@BF:~$ python3 ⏎
Python 3.10.4 (main, Jun 29 2022, 12:14:53) [GCC 11.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.

>>> import cv2 ⏎

>>> print(cv2.__version__) ⏎
4.5.4

>>> [ctrl] + D

prog@BF:~$
```

```
--------------- OpenCV-C++ program with image --------------

//edge.cpp
#include "opencv2/core/utility.hpp"
#include "opencv2/imgproc.hpp"
#include "opencv2/imgcodecs.hpp"
#include "opencv2/highgui.hpp"
#include<iostream>

using namespace cv;
using namespace std;

int edgeThresh = 1;
int edgeThreshScharr=1;

Mat image, gray, blurImage, edge1, edge2, cedge;

const char* window_name1 = "Edge map : Canny default (Sobel gradient)";
const char* window_name2 = "Edge map : Canny with custom gradient (Scharr)";

// define a trackbar callback
static void onTrackbar(int, void*){

blur(gray, blurImage, Size(3,3));

// Run the edge detector on grayscale
Canny(blurImage, edge1, edgeThresh, edgeThresh*3, 3);
cedge = Scalar::all(0);

image.copyTo(cedge, edge1);
imshow(window_name1, cedge);

/// Canny detector with scharr
Mat dx,dy;
Scharr(blurImage,dx,CV_16S,1,0);
Scharr(blurImage,dy,CV_16S,0,1);
Canny( dx,dy, edge2, edgeThreshScharr, edgeThreshScharr*3 );
/// Using Canny's output as a mask, we display our result
cedge = Scalar::all(0);
image.copyTo(cedge, edge2);
imshow(window_name2, cedge);
}

static void help(const char** argv)
{
printf("\nThis sample demonstrates Canny edge detection\n"
"Call:\n"
" %s [image_name -- Default is logo.png]\n\n", argv[0]);
}

const char* keys =
{
"{help h||}{@image |logo.png|input image name}"
};
```

```
int main( int argc, const char** argv )
{
help(argv);
CommandLineParser parser(argc, argv, keys);
string filename = parser.get<string>(0);

image = imread(samples::findFile(filename), IMREAD_COLOR);
if(image.empty())
{
printf("Cannot read image file: %s\n", filename.c_str());
help(argv);
return -1;
}
cedge.create(image.size(), image.type());
cvtColor(image, gray, COLOR_BGR2GRAY);

// Create a window
namedWindow(window_name1, 1);
namedWindow(window_name2, 1);

// create a toolbar
createTrackbar("Canny threshold default", window_name1, &edgeThresh, 100, onTrackbar);
createTrackbar("Canny threshold Scharr", window_name2, &edgeThreshScharr, 400, onTrackbar);

// Show the image
onTrackbar(0, 0);

// Wait for a key stroke; the same function arranges events processing
waitKey(0);

return 0;
}
```

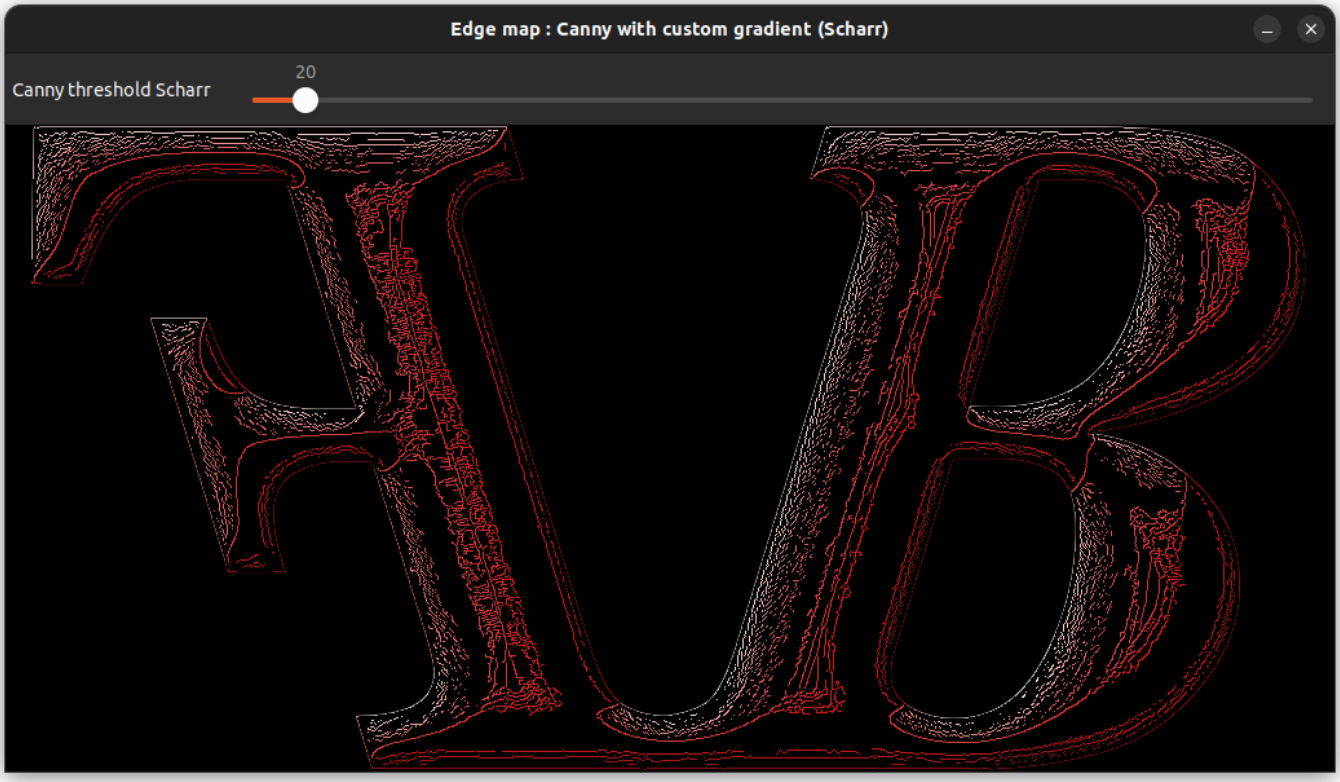
logo.png

```
prog@BF:~/0_OpenCV4/edge$ ls ⏎
edge.cpp  logo.png

prog@BF:~/0_OpenCV4/edge$ g++ -ggdb edge.cpp -o edge \
`pkg-config --cflags --libs opencv4` ⏎

prog@BF:~/0_OpenCV4/edge$ ./edge ⏎
```


Edge map : Canny with custom gradient (Scharr)
Canny threshold Scharr
0


Edge map : Canny with custom gradient (Scharr)
Canny threshold Scharr
20