**Wybór urządzenia na którym zastanie zaimplementowane środowisko oraz testowane będzie działanie systemu wykrywania obiektów przygotowanie środowiska oraz instalacja bibliotek:**

**Wybór urządzenia:**

Raspberry Pi - popularnie zwana malinka, jest to platforma komputerowa stworzona przez Raspberry Pi Foundation. Urządzenie składa się z płytki drukowanej i zostało stworzone z myślą o wspieraniu i rozwoju informatyki. Pierwsza wersja skłąda się z układu Broadcom BCM2835, procesora ARM1176JZFS-S o częstotliwości 700MHz, VideoCore IV 256MB lub 512MB.

Co ciekawe urządzenie nie ma w sobie dysku twardego jednak umożliwia uruchomienie kodu z karty mikroSD. Raspberry Pi działa pod kontrolą systemu operacyjnego jakim jest Linux oraz RISC OS, natomiast po wyjściu wersji Raspberry Pi 2 B użytkownicy mają możliwość działania pod kontrolą Windows10.

Projekt opiera się na wersji Raspberry Pi 4, która posiada nowszy układ Broadcom BCM2711, procesor ARM-8 Cortex-A72 (64 bit) o mocy 1.5GHz z czterema rdzeniami oraz 4GB fizycznej pamięci SDRAM o częstotliwości 2400Hz. Wersja Pi 4 daje możliwość podłączenia kamerki, którą możemy sparować z Pi 4 za pomocą taśmy, co ciekawe urządzenie posiada mocną matrycę HD wspierająca tryb 720p oraz 1080p.

**Wybór biblioteki:**

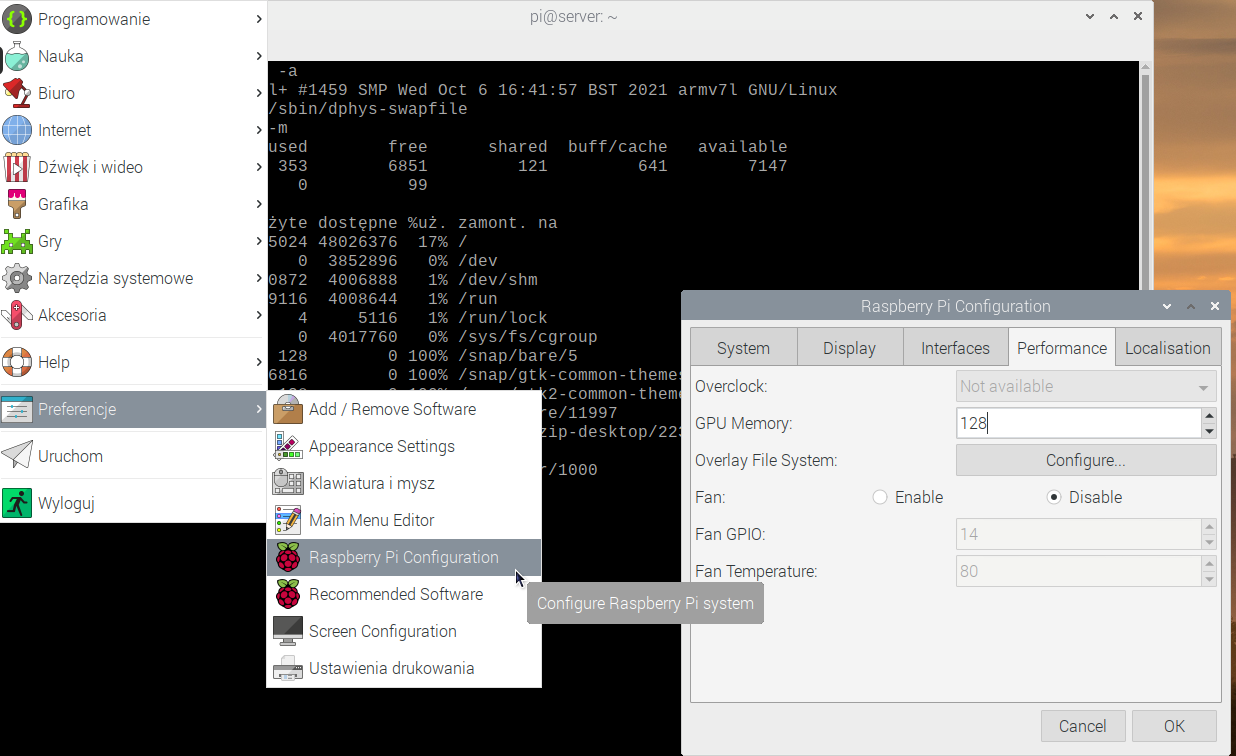
OpenCV (Open Source Computer Vision) jest wieloplatformową (Windows, Linux, Mac),otwartą biblioteką programistyczną do zastosowań w widzeniu maszynowym. Prace nad biblioteką zostały rozpoczęte w 1999 roku w laboratoriach Intela. Wersja alfa została udostępniona w roku 2000, natomiast pierwsza dojrzała wersja 1.0 została opublikowana w roku 2006. Biblioteka OpenCV jest wciąż rozwijana a jej możliwości już w obecnej wersji wykraczają poza typowy zakres widzenia maszynowego. Jednym z ważniejszych kierunków rozwoju OpenCV jest analiza obrazów 3D.

Biblioteka ma ponad 2500 zoptymalizowanych algorytmów, które obejmują kompleksowy zestaw zarówno klasycznych, jak i najnowocześniejszych algorytmów wizji komputerowej i uczenia maszynowego. Jest ona zaimplementowana w języku C/C++ z interfejsami m.in. do języka Pythona. Oficjalna strona OpenCV: <https://opencv.org/>

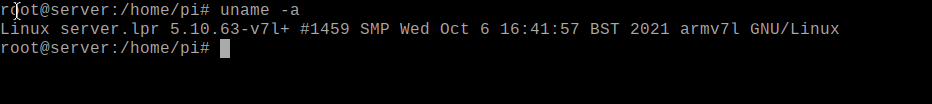
OpenCV od wersji 2.3 w Pythonie do reprezentowania obrazów używa tablic z pakietu NumPy. Dlatego wszystkie podstawowe operacje na obrazach (np. odczytanie wartości jakiegoś piksela) można wykonywać tak jak na zwykłych tablicach wielowymiarowych na przykład krotkach. Zarówno algorytmy przetwarzania obrazu, jak i algorytmy wdzenia komputerowego (CV) przyjmują obraz jako dane wejściowe, jednak w przewarzaniu obrazu wyjściem jest również obraz, podczas gdy w wizji komputerowej wyjściem mogą być pewne cechy informacje o obrazie. OpenCV to najlepsze rozwiązanie w przypadku wykorzystania do klasyfikacji zdjęć, do wykrywania obiektów i segmentacji obrazu używając wcześniej wytrenowanej sieci DNN.

**Przygotowanie systemu:**

Zwiększenie pamięci GPU do 128 MB – wchodzimy w Raspberry PI configuration zakładka **Performance** i ustawiamy **GPU memory**  na 128 MB:



Zanim zaczniemy instalować bibliotekę OpenCV na Raspberry Pi 4 musimy najpierw sprawdzić wersję systemu jaką posiadamy 32 czy 64 bitową i taką należy później pobrać.



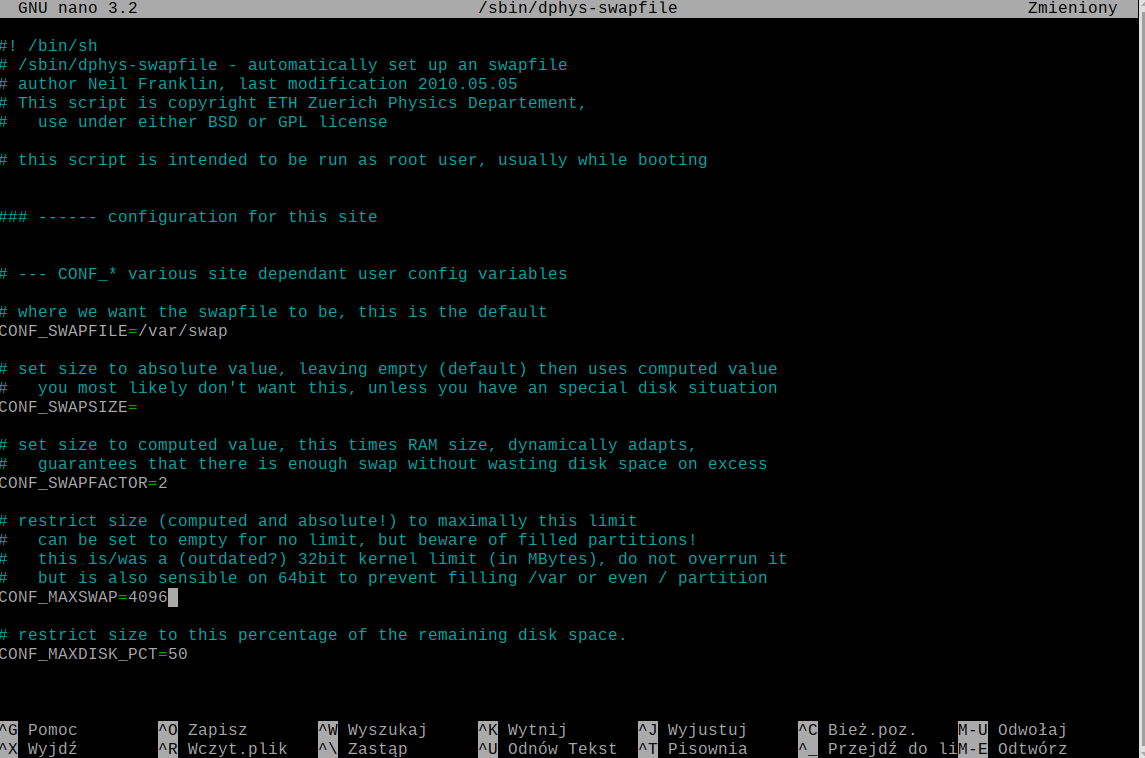
*Armv71* wskazuje nam że nasza wersja systemu jest 32 bitowa.

**Ustawianie pamięci swap:**

Biblioteki OpenCV potrzebują dużo pamięci podczas kompilowania dlatego będziemy musieli zwiększyć standardową z 2048 MB do 4096 MB w tym celeu należy edytować plik */sbin/dhphys-swapfile.*

**

Zmieniamy wartość ***CONF\_MAXSWAP=*** z **2048** na **4096** – Ctrl+o zapisujemy ctrl+x wychodzimy.



**Aktualizacja apt-get oraz instalacja niezbędnych bibliotek niezbędnych do zadziałania OpenCV:**

*$ sudo apt-get update*

*$ sudo apt-get upgrade*

*$ sudo apt-get install cmake gfortran*

*$ sudo apt-get install python3-dev python3-numpy*

*$ sudo apt-get install libjpeg-dev libtiff-dev libgif-dev*

*$ sudo apt-get install libgstreamer1.0-dev gstreamer1.0-gtk3*

*$ sudo apt-get install libgstreamer-plugins-base1.0-dev gstreamer1.0-gl*

*$ sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev*

*$ sudo apt-get install libgtk2.0-dev libcanberra-gtk\**

*$ sudo apt-get install libxvidcore-dev libx264-dev libgtk-3-dev*

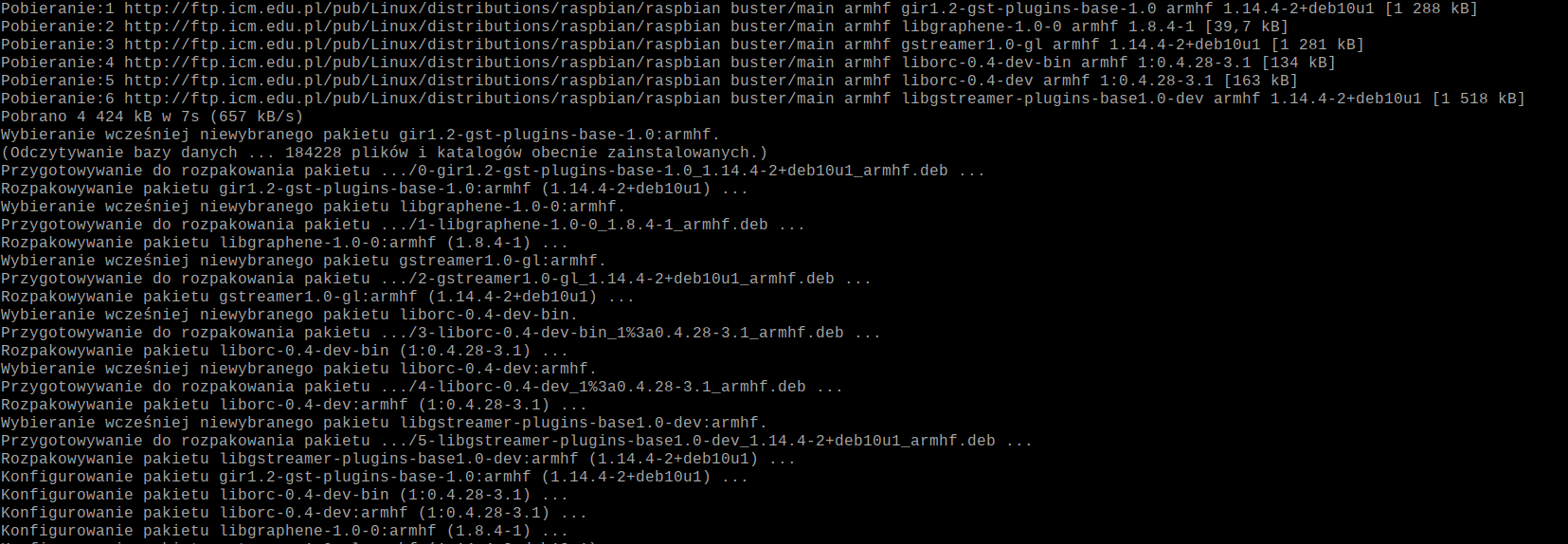
*$ sudo apt-get install libtbb2 libtbb-dev libdc1394-22-dev libv4l-dev*

*$ sudo apt-get install libopenblas-dev libatlas-base-dev libblas-dev*

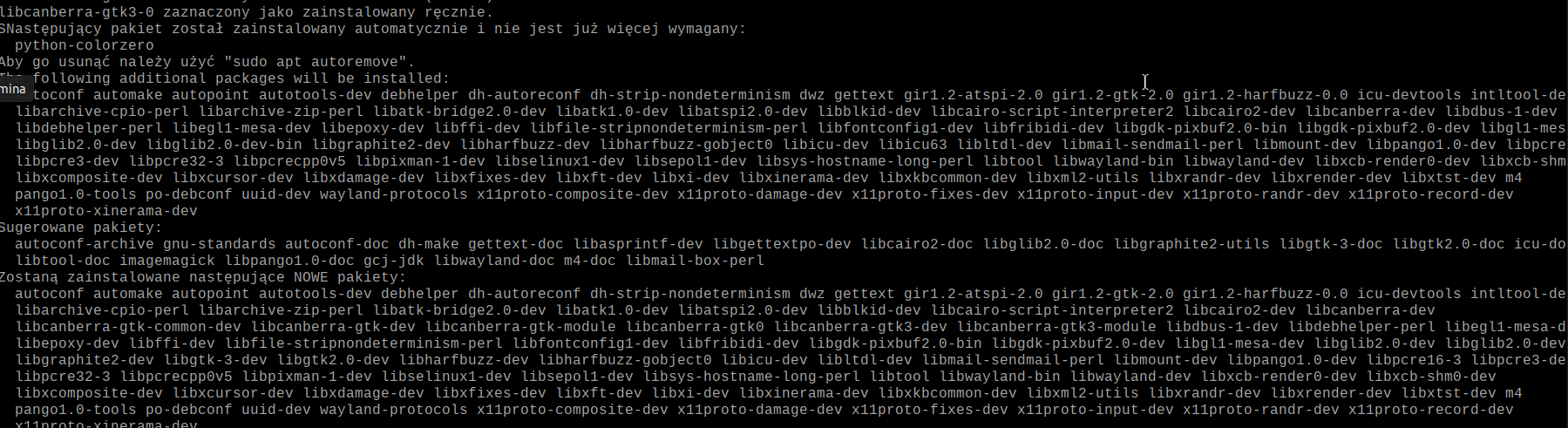
*$ sudo apt-get install libjasper-dev liblapack-dev libhdf5-dev*

*$ sudo apt-get install protobuf-compiler*

*$ sudo apt-get install python-dev python-numpy*

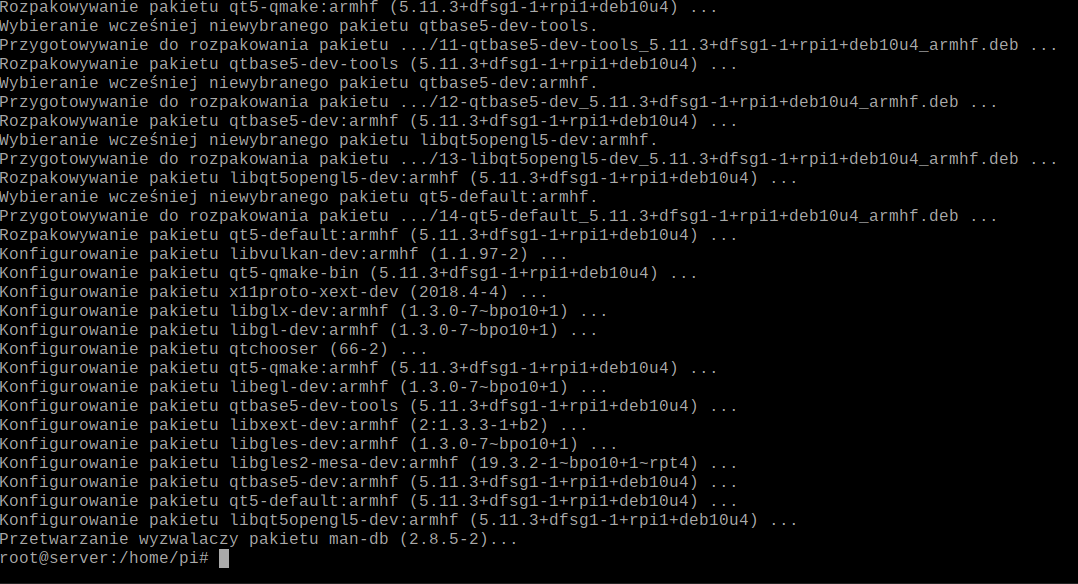


I



**Instalacja Qt5 – graficznego interfejsu użytkownika wspomagającego wyświetlanie i edycję.**

***sudo apt-get install qt5-default***

******

**Pobieranie biblioteki OpenCV 4.5.4:**

$ wget -O opencv.zip https://github.com/opencv/opencv/archive/4.5.4.zip

$ wget -O opencv\_contrib.zip https://github.com/opencv/opencv\_contrib/archive/4.5.4.zip

# Rozpakowywanie:

$ unzip opencv.zip

$ unzip opencv\_contrib.zip

# Zmiana nazw katalogów:

$ mv opencv-4.5.4 opencv

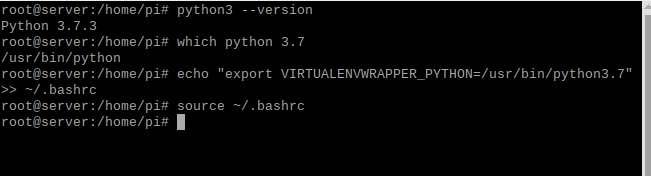
$ mv opencv\_contrib-4.5.4 opencv\_contrib

# usuwanie zbędnych plików

$ rm opencv.zip

$ rm opencv\_contrib.zip

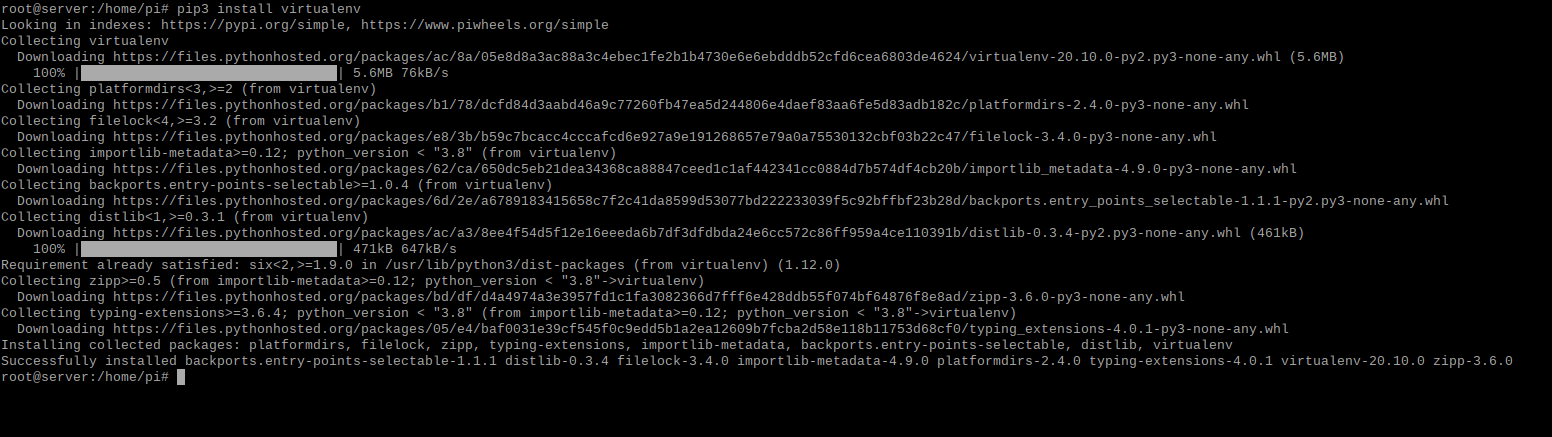
**Przygotowanie wirtualnego środowiska:**

****

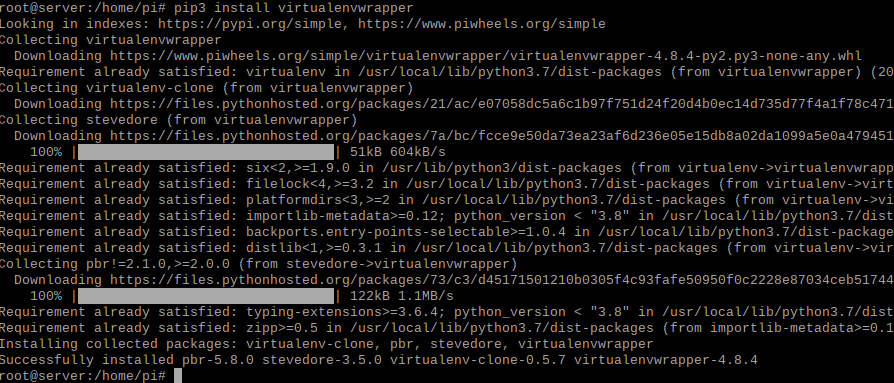
Instalacja wirtualnego środowiska:

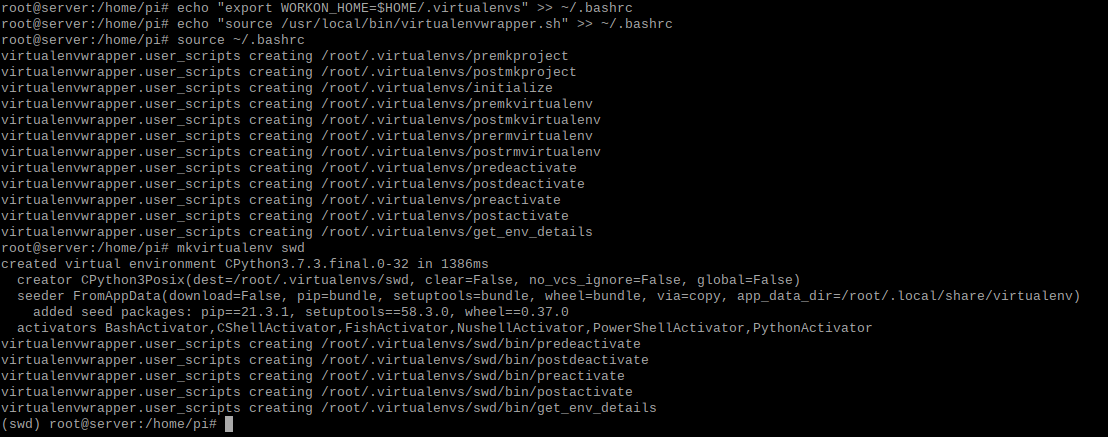
*pip3 install virtualenv*

*pip3 install virtualenvwrapper*

****

****

Eksportowanie ścieżki oraz stworzenie testowego środowiska swd (szlag-włodyga-drewniak):



Instalacja numpy poprzez wydanie komendy:

*pip3 install numpy*

*Tworzenie katalogu build przed kompilacją openCV:*

$ cd ~/opencv/

$ mkdir build

$ cd build

*Kompilacja:*

$ cmake -D CMAKE\_BUILD\_TYPE=RELEASE \

-D CMAKE\_INSTALL\_PREFIX=/usr/local \

-D OPENCV\_EXTRA\_MODULES\_PATH=~/opencv\_contrib/modules \

-D ENABLE\_NEON=ON \

-D ENABLE\_VFPV3=ON \

-D WITH\_OPENMP=ON \

-D WITH\_OPENCL=OFF \

-D BUILD\_ZLIB=ON \

-D BUILD\_TIFF=ON \

-D WITH\_FFMPEG=ON \

-D WITH\_TBB=ON \

-D BUILD\_TBB=ON \

-D BUILD\_TESTS=OFF \

-D WITH\_EIGEN=OFF \

-D WITH\_GSTREAMER=ON \

-D WITH\_V4L=ON \

-D WITH\_LIBV4L=ON \

-D WITH\_VTK=OFF \

-D WITH\_QT=OFF \

-D OPENCV\_ENABLE\_NONFREE=ON \

-D INSTALL\_C\_EXAMPLES=OFF \

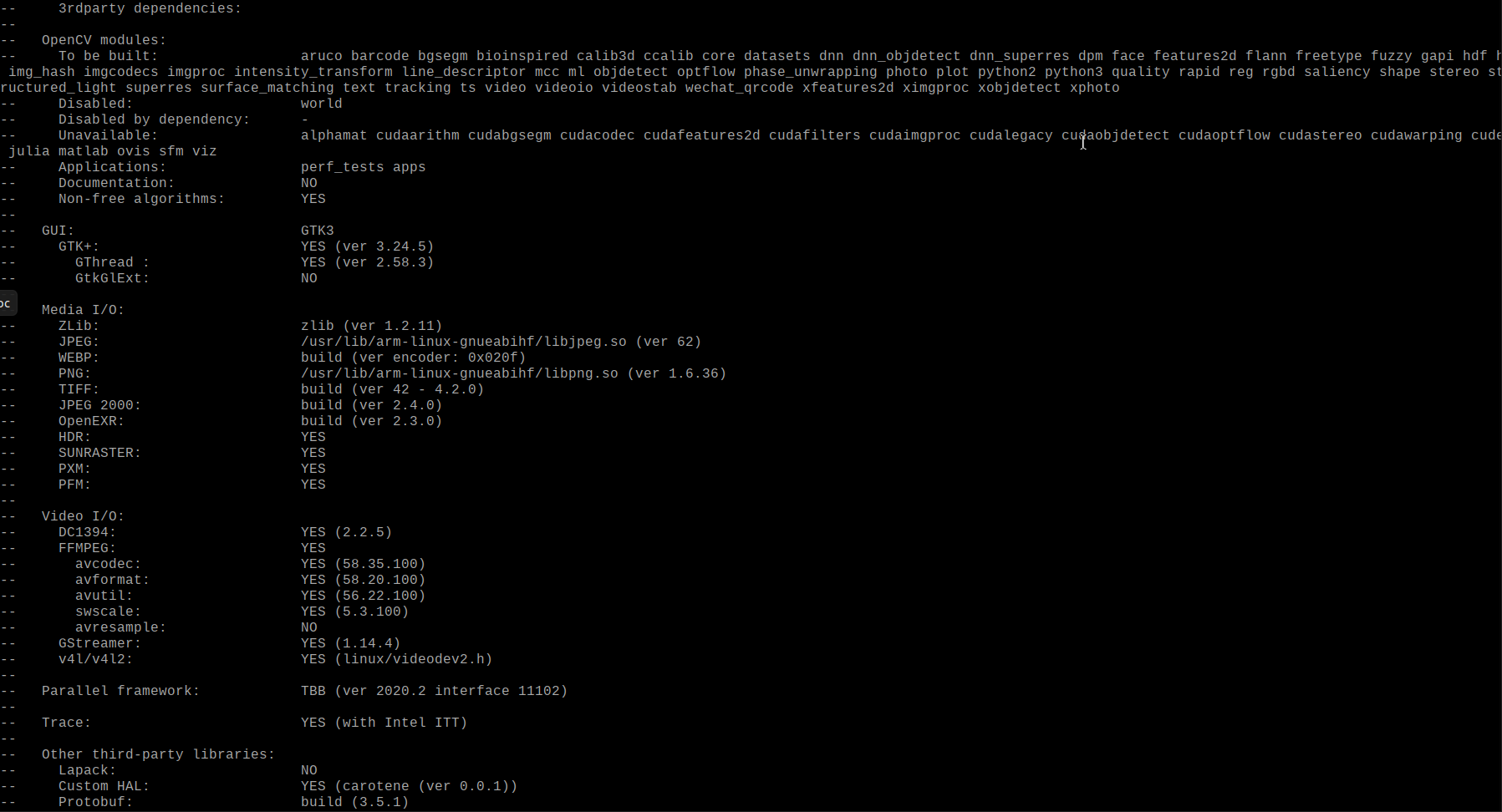
-D INSTALL\_PYTHON\_EXAMPLES=OFF \

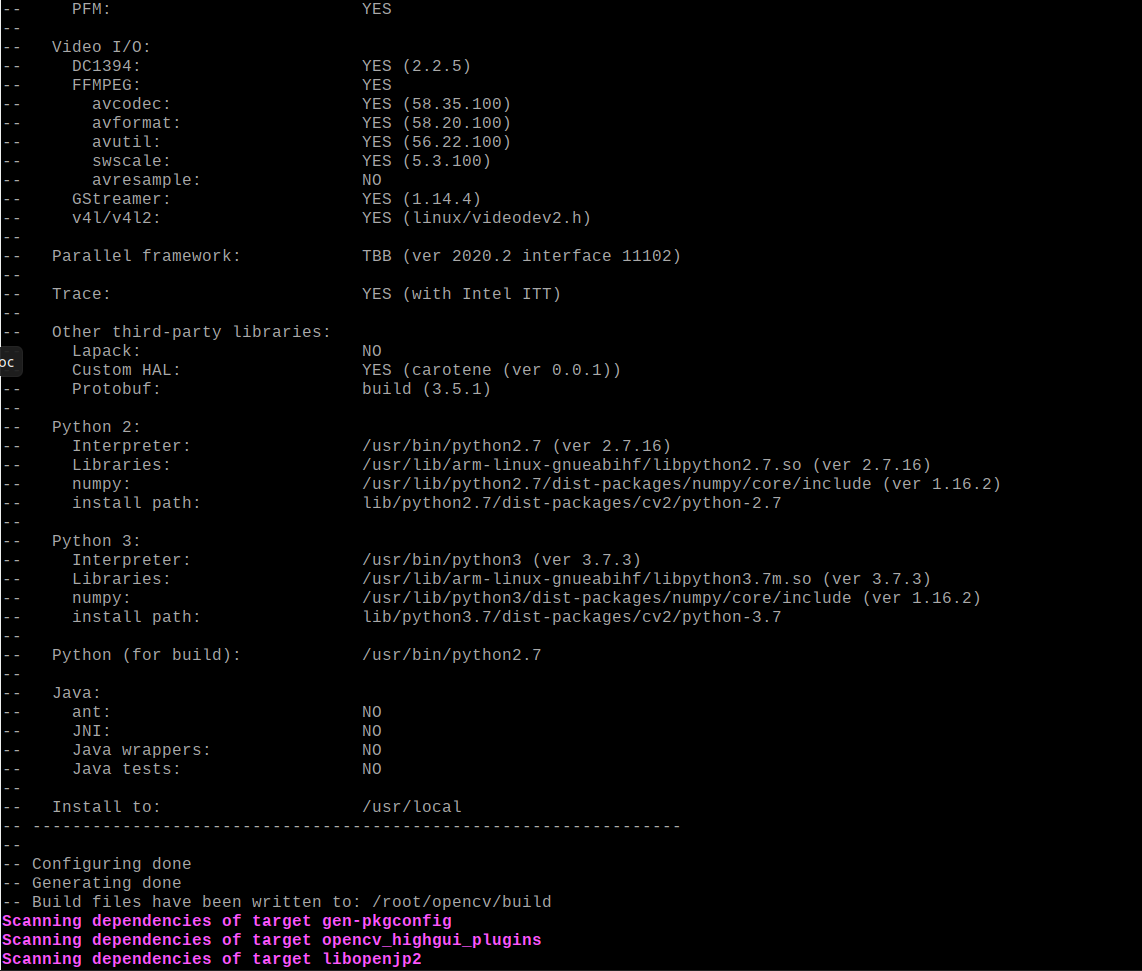
-D BUILD\_opencv\_python3=TRUE \

-D OPENCV\_GENERATE\_PKGCONFIG=ON \

-D BUILD\_EXAMPLES=OFF ..

*Wynik:*

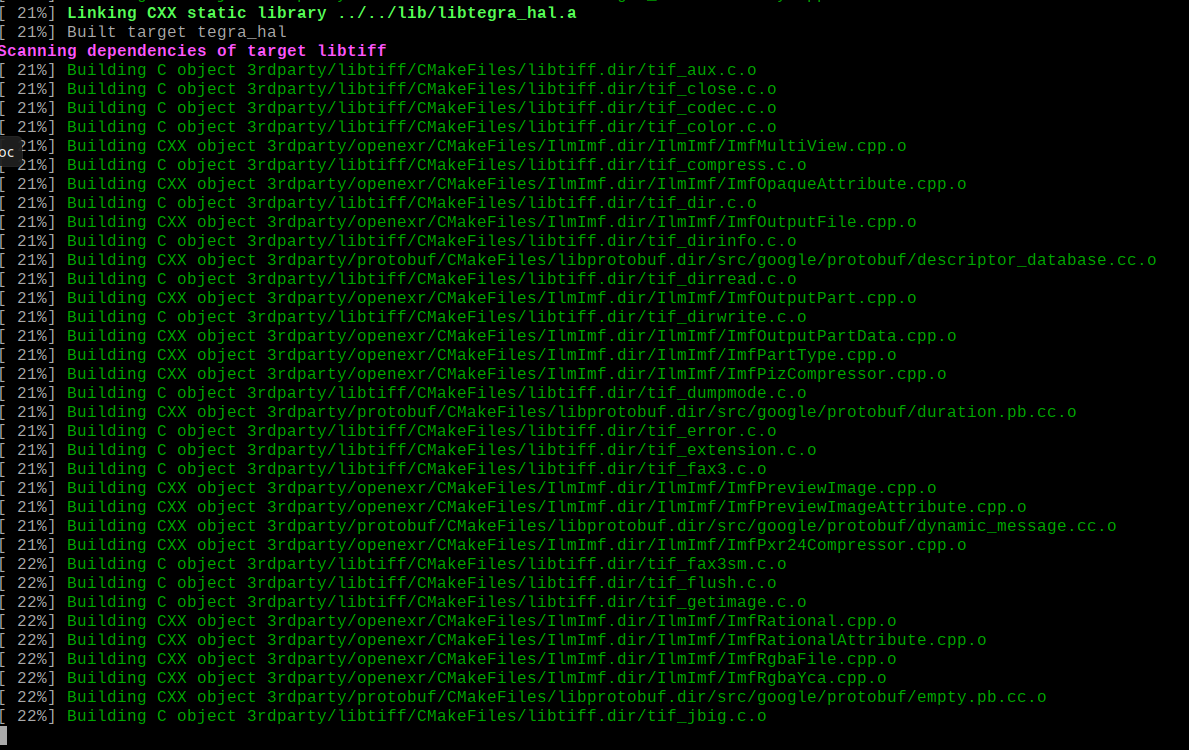
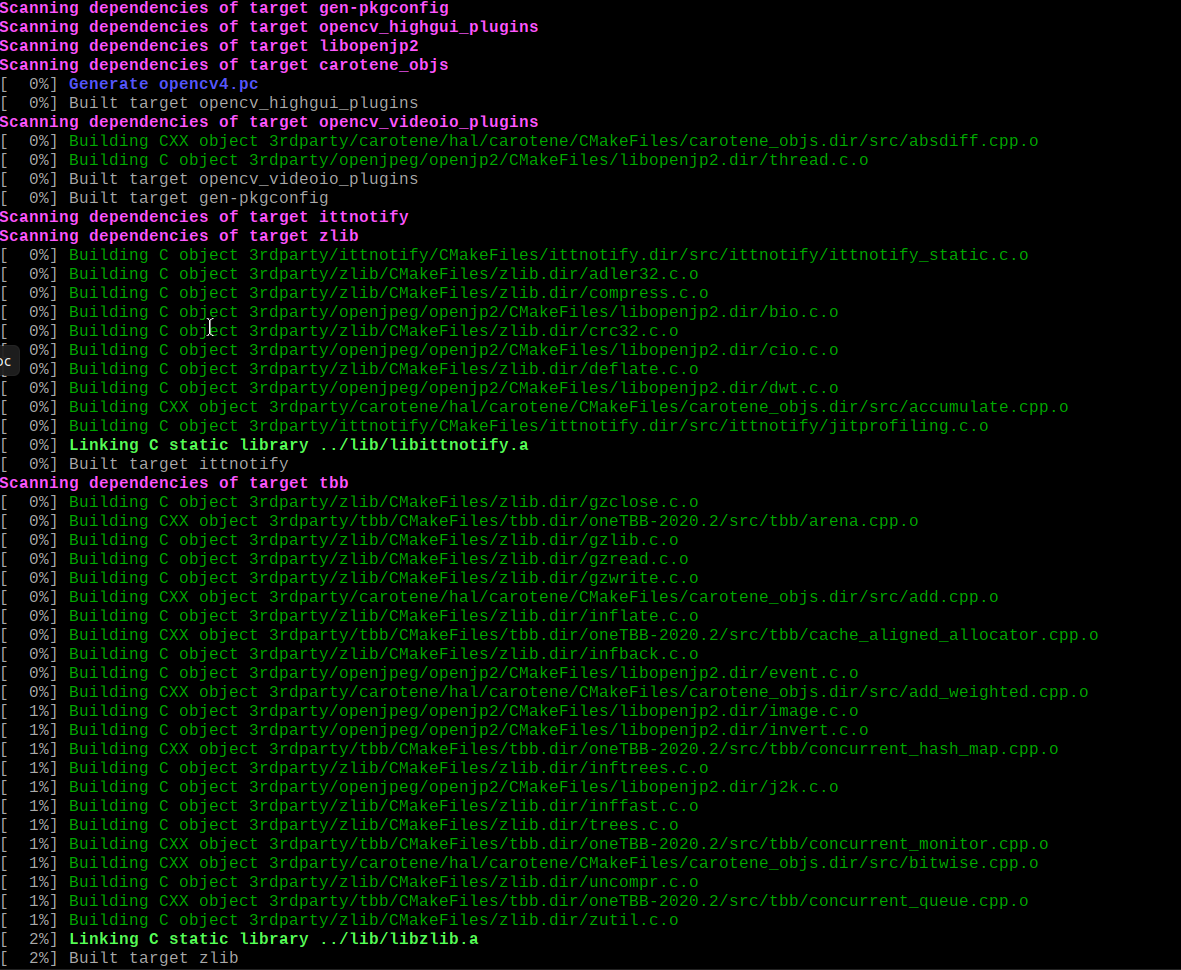
**

**

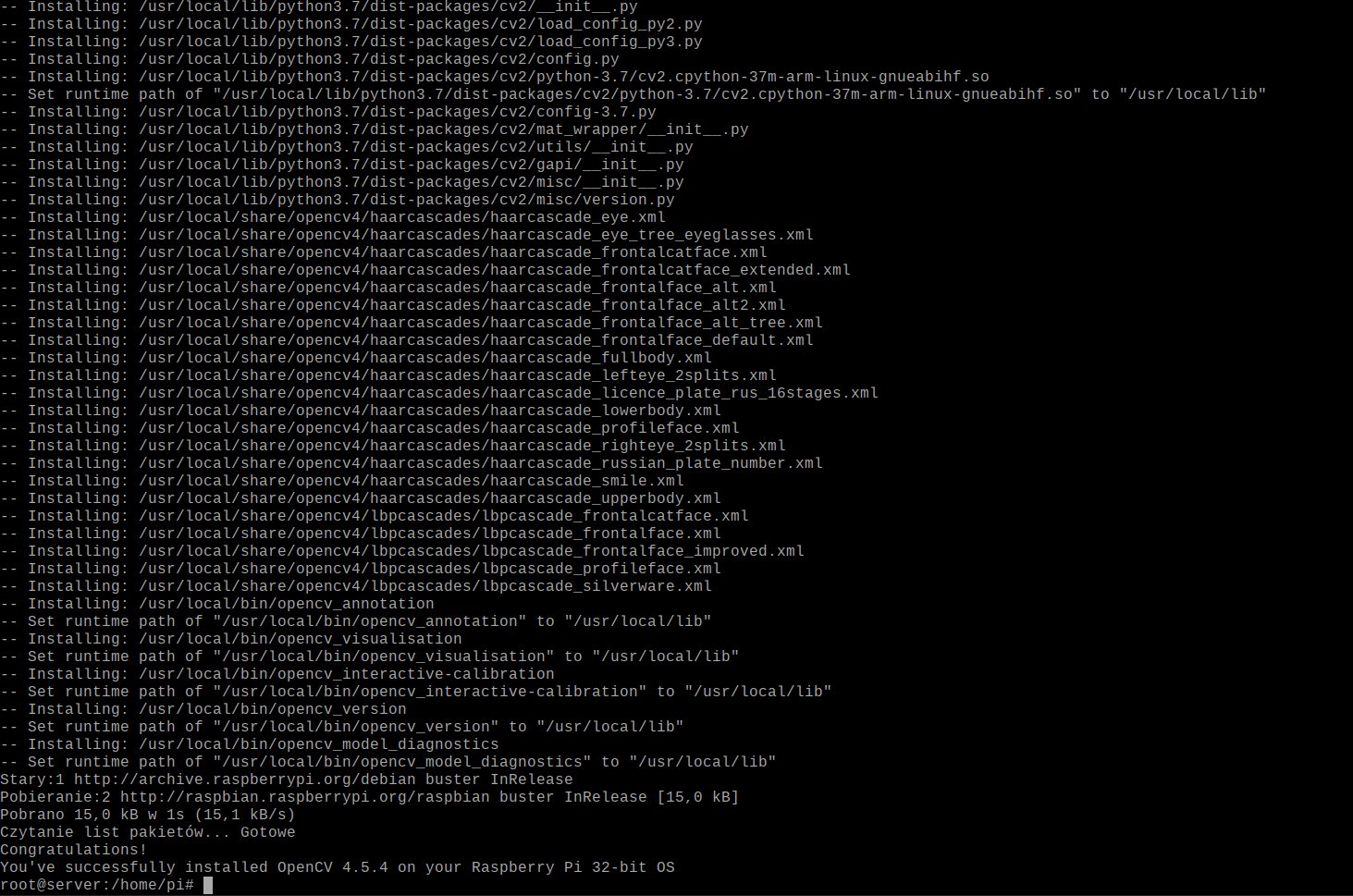
*Make:*

$ make -j4

*Na Raspberry Pi 4 8GB kompilacja potrwa około 1 godziny:*

**

*Make install:*

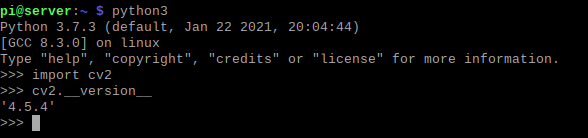
**

*Dodanie symbolicznego linku do biblioteki:*

(swd) root@server:~/.virtualenvs/swd/lib/python3.7/site-packages# ls -s /usr/local/lib/python3.7/dist-packages/cv2/python-3.7/cv2.cpython-37m-arm-linux-gnueabihf.so

8036 /usr/local/lib/python3.7/dist-packages/cv2/python-3.7/cv2.cpython-37m-arm-linux-gnueabihf.so

*Sprawdzanie poprawności linku:*

**

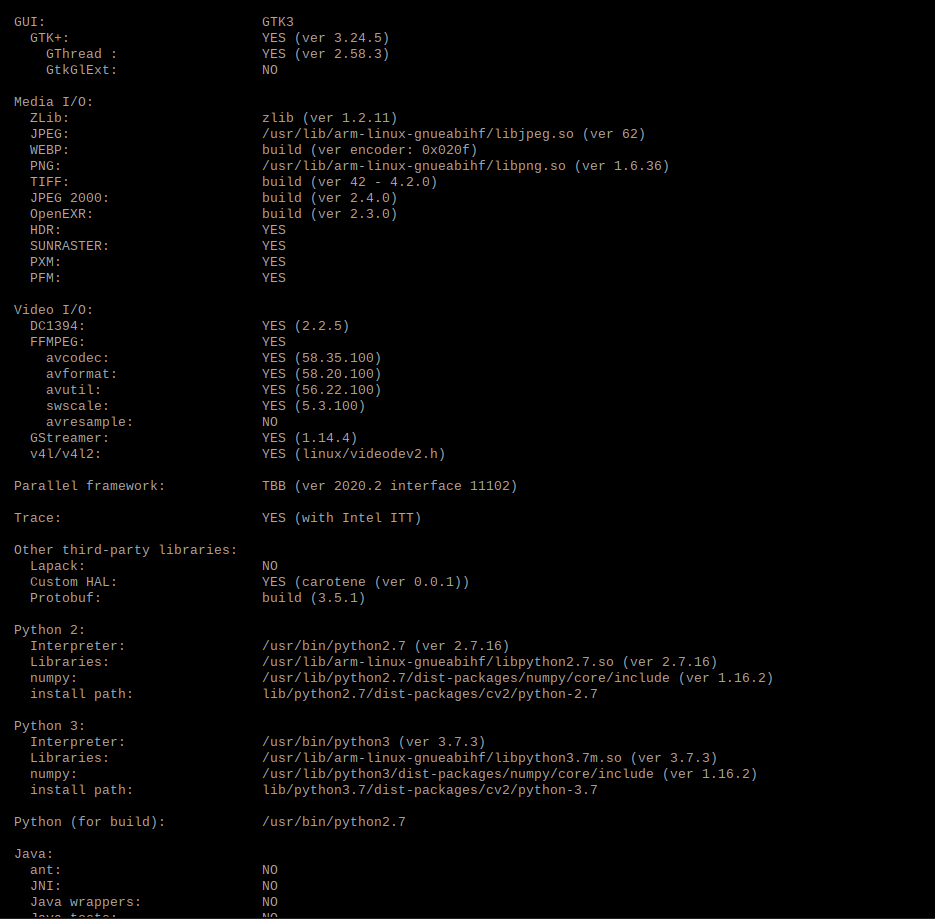
*Sprawdzanie informacji o kompilacji OpenCV2:*

$ python

>>> import cv2

>>> print( cv2.getBuildInformation() )

**

**