NVIDIA Jetson Nano - OpenPose Installation Guide

Zacharia Karami

June 2019

1 Introduction

This guide is for installing OpenPose on the NVIDIA Jetson Nano. It requires an internet connection on the Nano and an SD card reader.

If simply using the image provided with the complete installation only an SD card reader is necessary. I recommend the use of a standard USB camera for testing your installation.

1.1 Requirements

- NVIDIA Jetson Nano (though a TX2 may also work)
- a USB Camera
- a keyboard, mouse and monitor
- an internet connection to your Nano
- a large enough MicroSD card (tested with 32 GB)

2 Installation of JetPack

JetPack is the OS used on the Jetson Nano, it is a Ubuntu-based OS customized by NVIDIA.

1. Download the SD card image from NVIDIA's website from the following link: https://developer.nvidia.com/embedded/dlc/jetson-nano-dev-kit-sd-card-image
2. Use an SD card writing software to write the image to a MicroSD card, I recommend a size of minimum 32 GB. For software, an option is Balena's Etcher (NVIDIA recommended – https://www.balena.io/etcher/) or Rufus.
3. Connect your Nano to a screen (HDMI), ethernet, a USB keyboard and a USB mouse.
4. Connect your Nano to power. You should now see the Ubuntu installation

screen.
5. Follow the Ubuntu installation instructions.

3 Building OpenPose

In this next section we will download and build OpenPose from the source on github. Open up a terminal on your Nano and input the following commands.

```
#customary sudo apt-get update
sudo apt-get update

#need to retrieve cmake
sudo apt-get -y install git

#if we want to use MP4 files we'll need ffmpeg
sudo apt-get -y install ffmpeg

#building OpenPose requires the latest version of cmake, so let's build cmake
sudo apt -y remove cmake

#we need https to be able to download several projects later
sudo apt-get -y install libcurl4-openssl-dev

#be sure to be in a folder where you want to download/build programs
git clone https://github.com/Kitware/CMake/archive/v3.14.0.zip
cd CMake

#bootstrap downloads and installs dependencies
./bootstrap --system-curl

#build cmake (takes approx. 13 minutes)
time make -j4
sudo make install

#reboot after installation
sudo reboot

#confirming the cmake version
cmake --version

#let's install the OpenPose repo
git clone https://github.com/CMU-Perceptual-Computing-Lab/openpose

cd openpose
```

```
#install depdencies
sudo bash ./scripts/ubuntu/install_deps.sh
mkdir build
cd build
cmake ..
make clean

#building on 4-cores will take approximately 30 minutes
time make -j4

#install the built openpose
sudo make install
```

With that, OpenPose should now be installed on your Nano board. However, this is not the end of the steps we must take to actually be able to use this.

4 Installing a Swap File

Running OpenPose's various pre-installed examples now will result in a freeze of the board, to the point where the mouse movement will be sluggish and no result will ever occur. The main cause of this is the lack of available RAM memory while running the program, as the Nano only has 4 GB of RAM.

A solution to this problem is installing a swap file on the external memory of the Nano (the MicroSD card). To do this, we will use an open-source application called installSwapFile[1]. Open up a terminal and use the following commands:

```
#clone installSwapFile
git clone https://github.com/JetsonHacksNano/installSwapfile

cd installSwapfile

#running the default will install a 6 GB swap file
#make sure you have enough space
./installSwapfile
```

With this, your Nano should have enough memory to run the OpenPose examples.

5 Running OpenPose's Examples

Now that your Nano has a bit more memory to work with, let's verify that OpenPose actually works by running a few of the examples.

5.1 Pose Estimation on an Image

Let's start with the simplest example, estimating the pose of people on a still image[2].

Enter the OpenPose folder on your Nano's filesystem and enter the following commands:

```
#this is one command with arguments
./build/examples/openpose/openpose.bin -image_dir ./examples/media/ --display 0
--model_folder ./models --write_images ./output/ --net_resolution 480x256
```

Then verify the result in the `/output/` folder. Congratulations, you now have a working OpenPose installation.

5.2 Pose Estimation on a Camera Feed

To use a USB Camera with OpenPose, we will need to install a few more things.

```
#canberra-gtk is necessary for running the camera example
sudo apt install canberra-gtk*
```

After this, start the camera example with the following command[3]:

```
./build/examples/openpose/openpose.bin  --model_folder ./models --net_resolution 320x176
```

The last argument is the resolution at which the poses are being estimated (not the display resolution of the camera application on-screen). At a resolution of 160x80 the application runs at 6 FPS, 10 FPS at a resolution 64x32. This is without any further optimizations.
Changing the resolution correlates positively with performance in terms of speed but negatively in terms of accuracy. Keep this in mind and adjust where necessary.

6 Conclusion

Now that we've gotten OpenPose working on a Jetson Nano the next step would be to use the library to create our own applications or to speed-up the current examples.

A Installing from Image

References

[1] *Jetson nano - use more memory*, 2019. [Online]. Available: https://www.jetsonhacks.com/2019/04/14/jetson-nano-use-more-memory/.

[2] Y. Sakamoto, *Nvidia jetson nano* で *openpose* をビルドする方法、動画から人体の骨格検出, 2019. [Online]. Available: http://www.neko.ne.jp/~freewing/raspberry_pi/nvidia_jetson_nano_build_openpose/.

[3] T. MAMMA, *Jetson nano* に *openpose* 入れてみる, 2019. [Online]. Available: https://toramamma.blogspot.com/2019/04/jetson-nano-openpose.html.