Istanbul Technical University

Control & Automation Engineering Department

Robotics Laboratory Implementation

The Ultimate Guide For NVIDIA Jetson TX2
ROS Camera Applications

Documented By:

-Samer Raed Aldabbas

-Moustafa M.M.M. Hassanein

Date: March 3, 2022

Contents

1 Introduction

1.1 NVDIA Jetson TX2 Developer Kit

The Jetson TX2 is the most powerful and energy-efficient embedded AI computer device. This 7.5-watt supercomputer-on-a-module enables real AI computation at the edge. It's based on an NVIDIA Pascal^{TM}-family GPU and comes with 8GB of RAM with memory bandwidth of 59.7GB/s. It has a number of standard hardware connections that allow it to be easily integrated into a wide range of devices and form factors.

Technical Specifications

JETSON TX2 MODULE

- NVIDIA Pascal™ Architecture GPU
- 2 Denver 64-bit CPUs + Quad-Core A57 Complex
- 8 GB L128 bit DDR4 Memory
- 32 GB eMMC 5.1 Flash Storage
- Connectivity to 802.11ac Wi-Fi and Bluetooth-Enabled Devices
- 10/100/1000BASE-T Ethernet

I/O

- USB 3.0 Type A
- USB 2.0 Micro AB (supports recovery and host mode)
- HDMI
- M.2 Key E
- PCI-E x4
- Gigabit Ethernet
- Full-Size SD
- SATA Data and Power
- GPIOs, I2C, I2S, SPI, CAN*
- TTL UART with flow control
- Display Expansion Header*
- Camera Expansion Header*
  - *I/O expansion headers: refer to product documentation for header specification.

POWER OPTIONS

- External 19V AC Adapter

JETSON CAMERA MODULE

- 5 MP Fixed Focus MIPI CSI Camera

BUTTONS

- Power On/Off
- Reset
- Force Recovery
- User-Defined

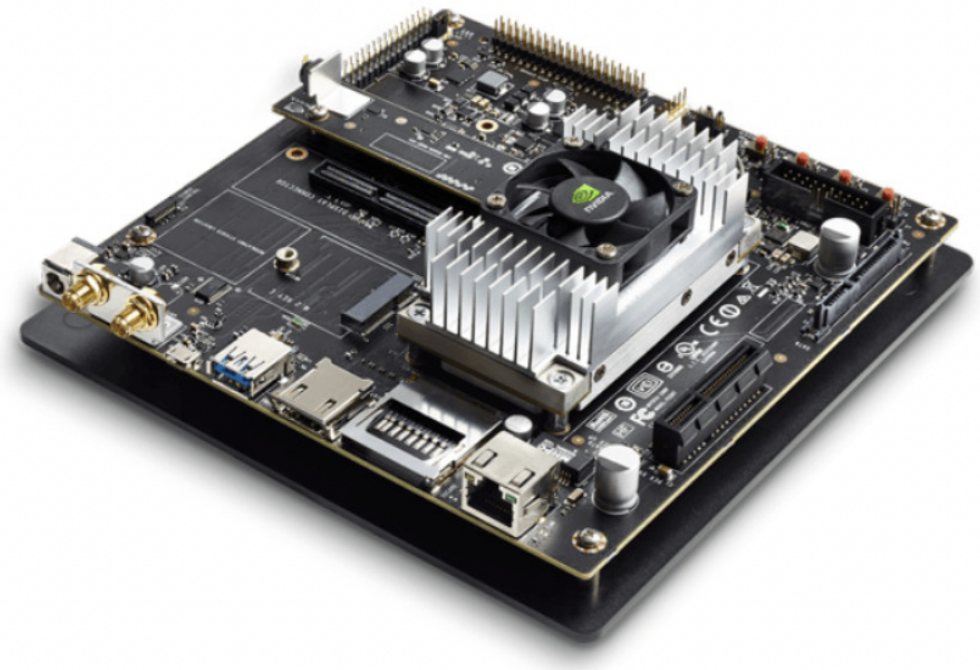


Figure 1.1: NVDIA Jetson TX2 Developer Kit

1.2 Orbbec Astra Pro

Astra is a powerful and reliable standalone 3D camera that includes the proprietary Orbbec 3D microchip and VGA color. Astra was developed to be highly compatible with existing OpenNI applications, making this 3D camera ideal for pre-existing apps that were built with OpenNI. Astra has a 0.4 to 8 meter range.

Figure 1.2: Orbbec Astra Pro Camera

Features:

- General purpose 3D camera;
- Longer sensing range;
- Alternative for Xtion/Kinect/Carmine 1.08
- Fully compatible with OpenNI

Specifications:

Model N°	ORBBEC ASTRA PRO
Size	160 x 30 x 40 (mm)
Weight	300 g
Range	0.4 -8 m
Depth Image Size	720p@30FPS
Field of View	60° horiz. x 49.5 ° vert. (73° diagonal)
Microphones	2
Operating Systems	Windows, Linux, Android

You can click here to be redirected to the original site for more information about Astra Pro.

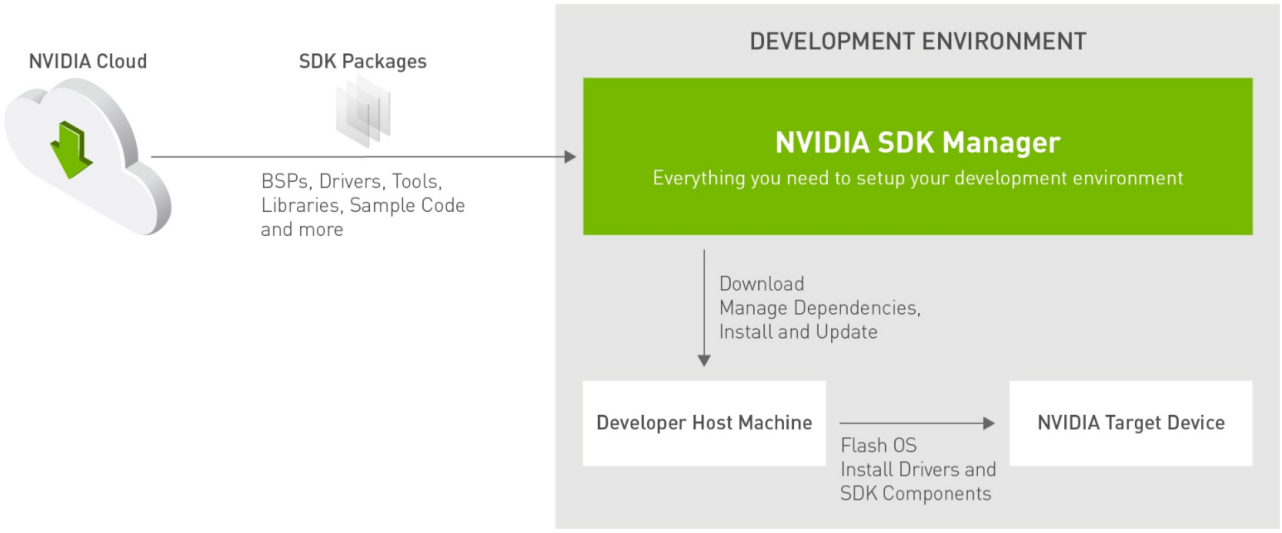
2 NVDIA Jetson TX2 setup

Hardware required:

- Jetson TX2 Developer Kit
- Monitor
- Keyboard
- Mouse
- Computer (OS: Ubuntu Linux x64 Version 18.04 or 16.04 / 32 GB free space)

Step 1) Install NVIDIA SDK Manager.

NVIDIA SDK Manager provides an end-to-end development environment setup solution for NVIDIA's DRIVE, Jetson, Clara Holoscan, Rivermax, DOCA and Ethernet Switch SDKs for both host and target devices.



After you download the Debian file (.deb), perform the following steps:

-From a terminal window, install the Debian package with the command:

```
$ sudo apt install ./sdkmanager_[version]-[build#]_amd64.deb
```

Note: Before continuing to the next step we need to setup (connect to computer) the development board and turn on the recovery mode. To do so, follow up with the "Recovery mode" section

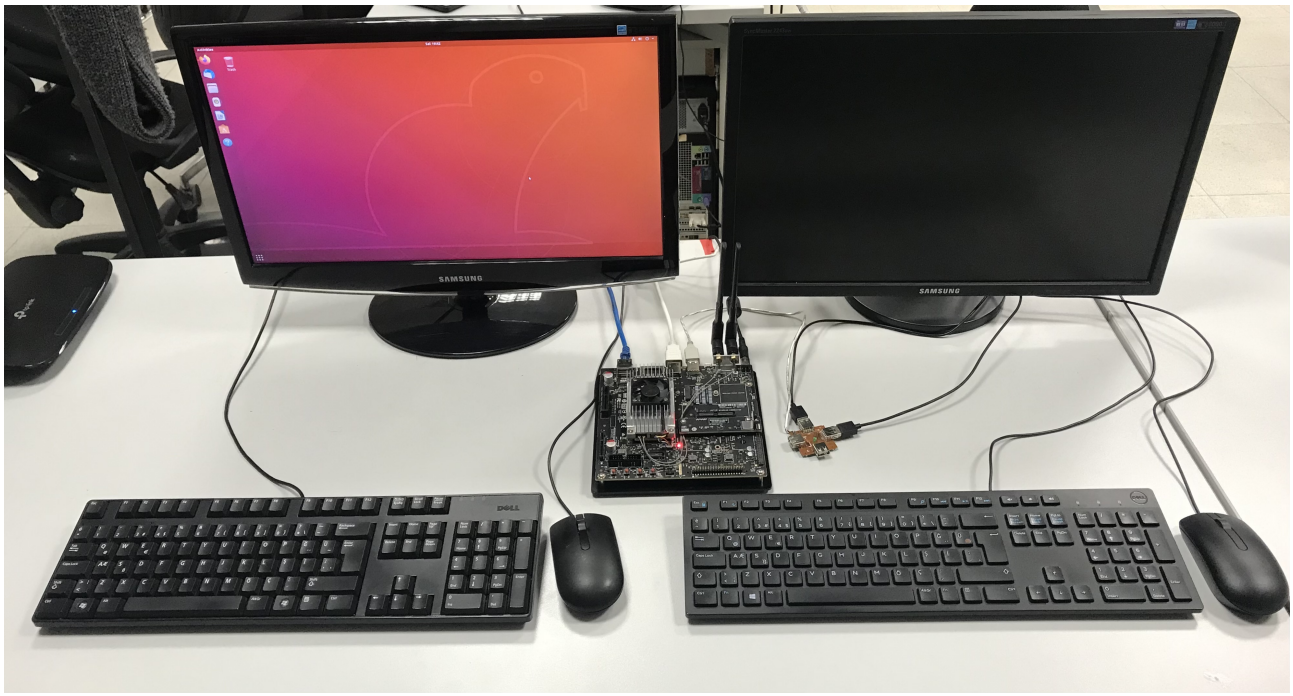
-From a terminal window, launch SDK Manager with the command:

```
$ sdkmanager
```

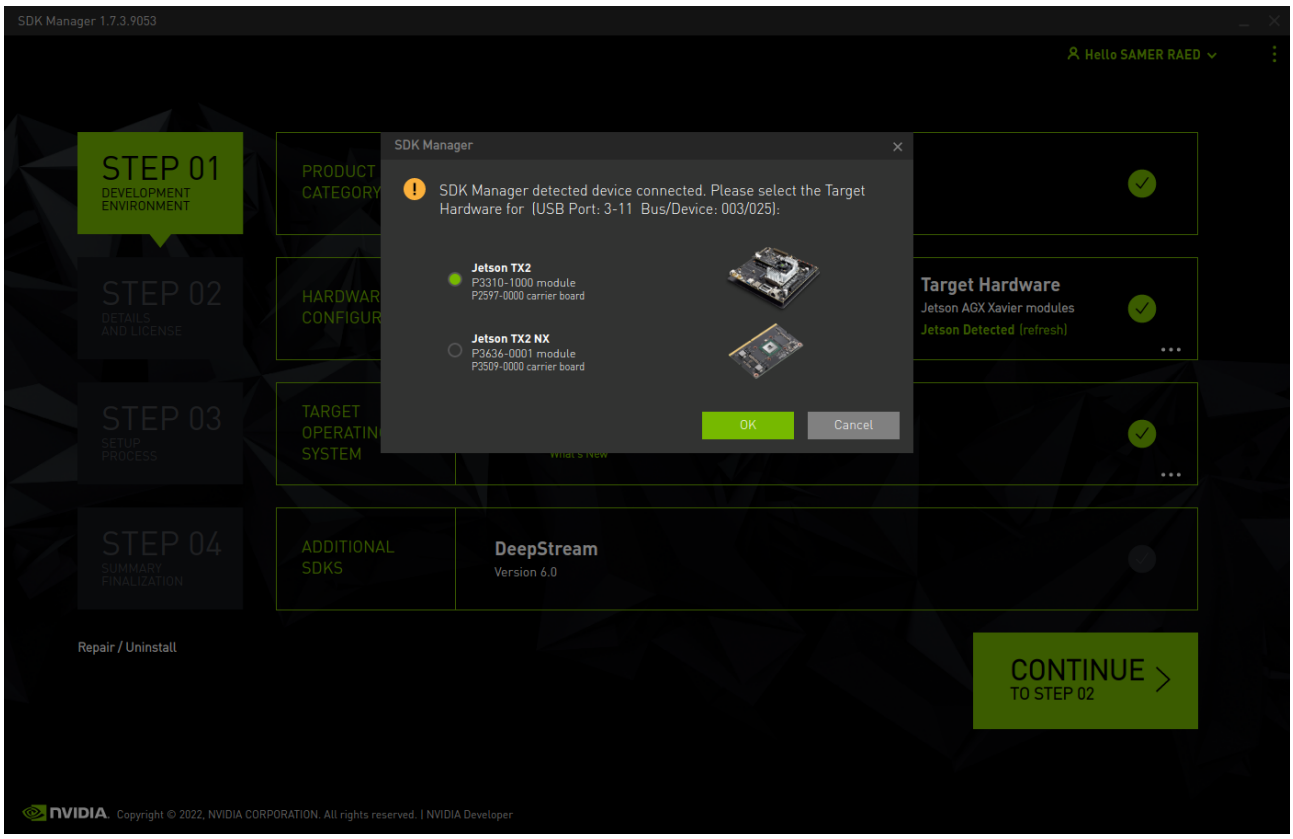
-From the SDK Manager launch screen, select the appropriate login tab for your account type then select the Product Category (Jetson) to install and follow the steps in the following section to complete the installation.

2.1 Installation Steps

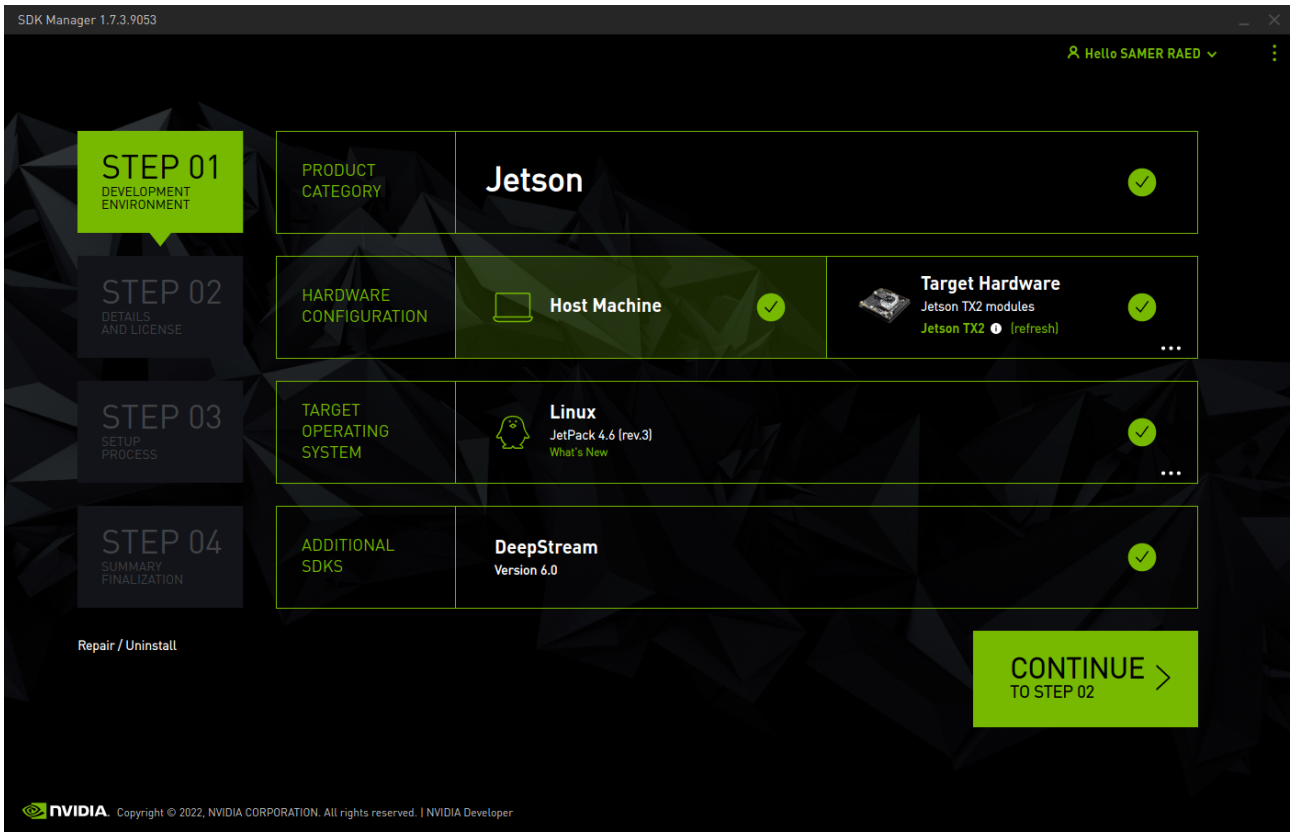
The figure below shows the needed hardware. The host device on the left and TX2 connected to its own monitor, keyboard and mouse on the right.



If the recovery mode was set correctly, the following must show up on the host computer.

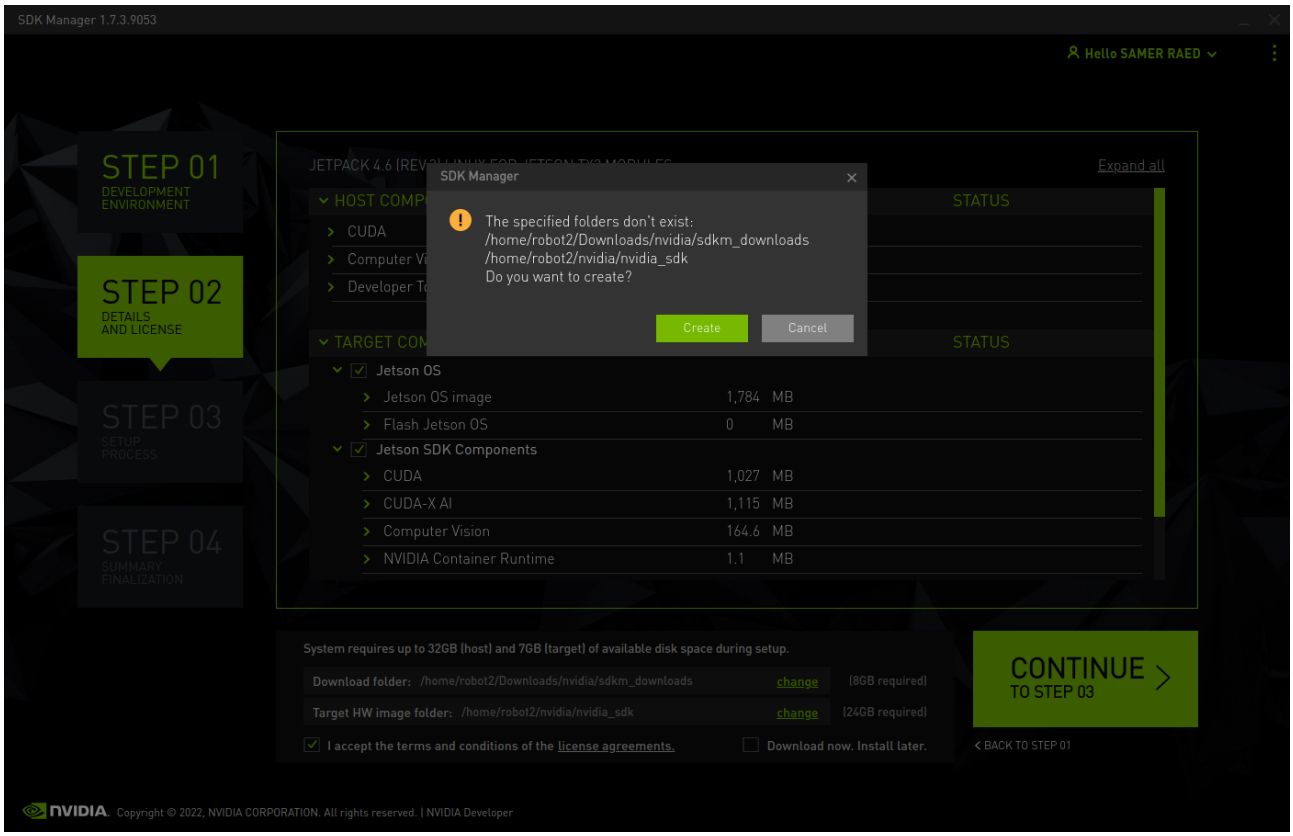


After selecting the the TX2 developing kit as target hardware, you are ready to start installation



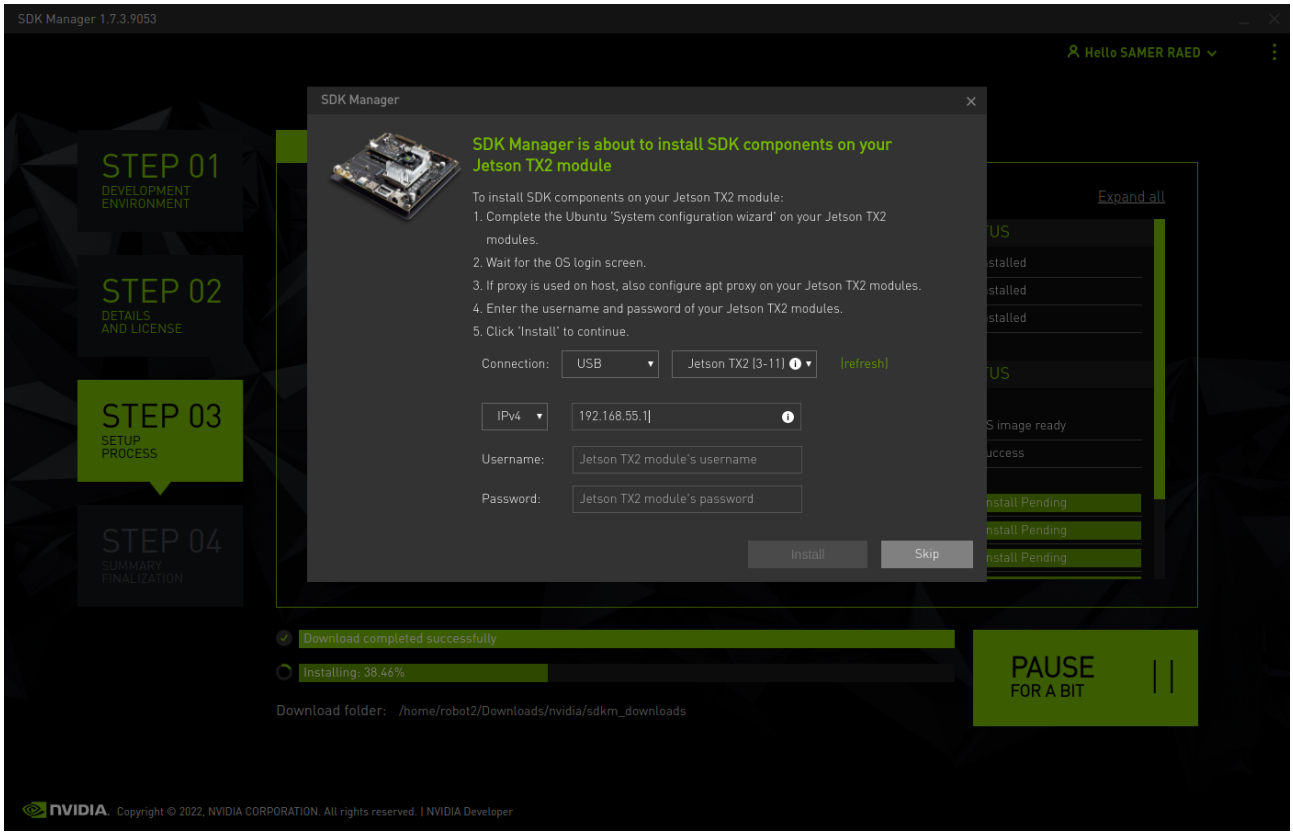
The installation requires around 32 GB free space on the host machine. Accept the agreement and continue.

SDK Manager will create the files were the OS image will be downloaded in.

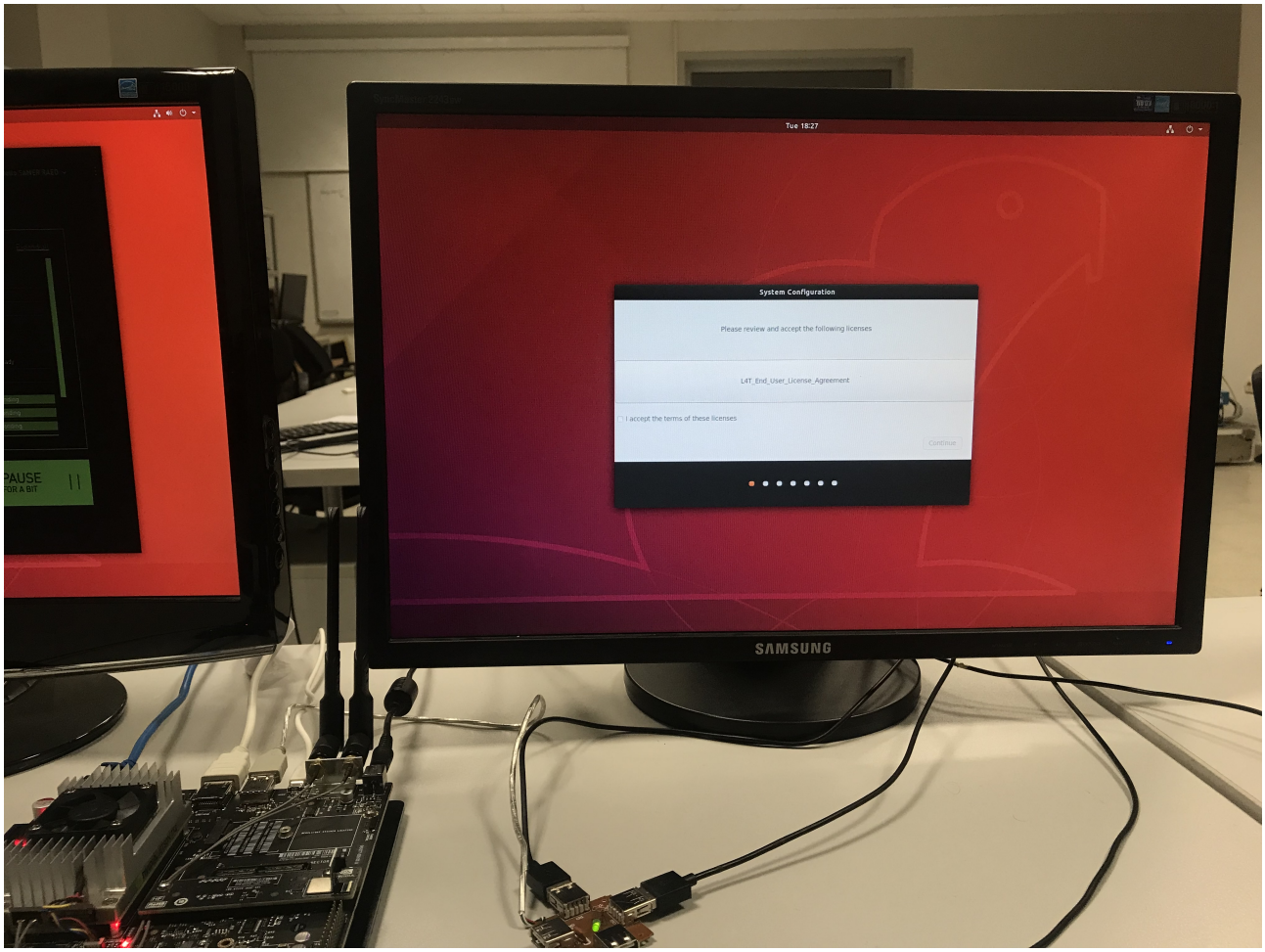


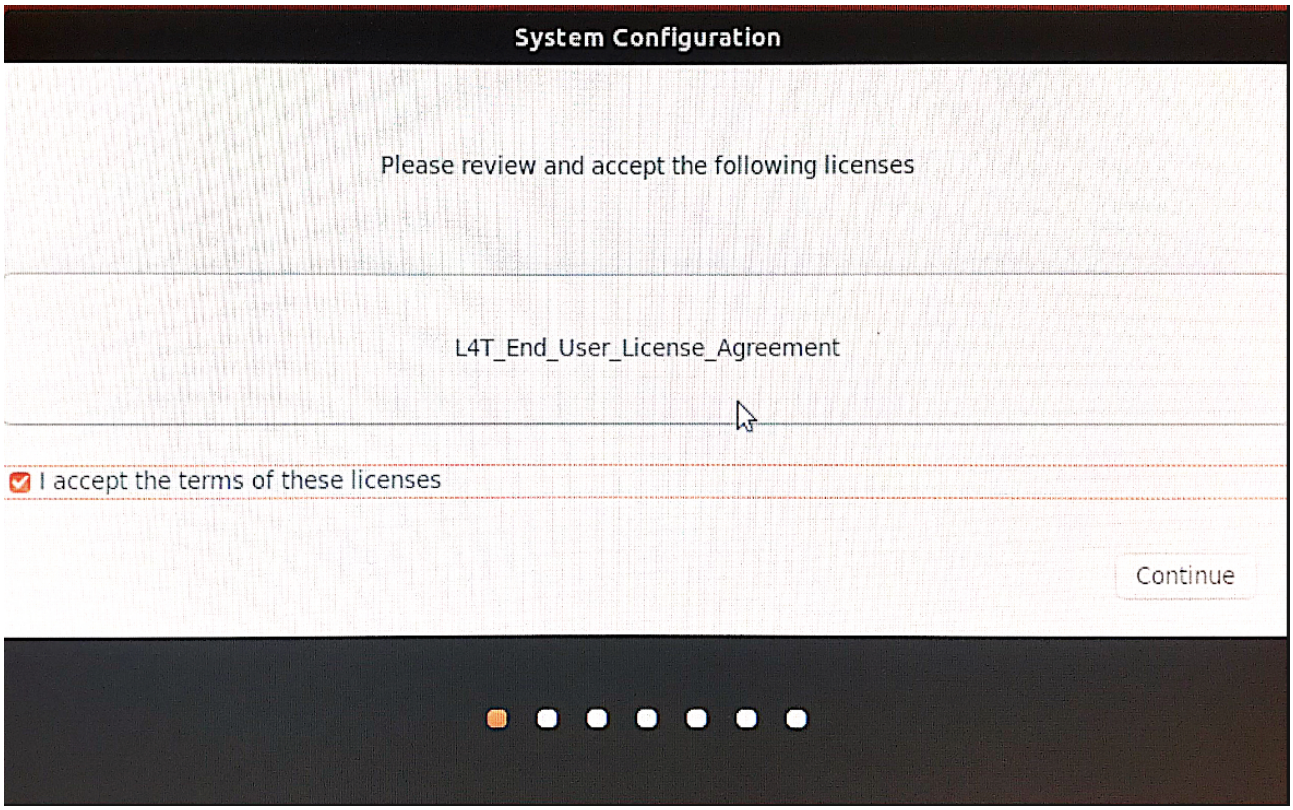
Now wait for the download to be done. It takes about 2 hours at an average internet speed.

When download is done, at some point of the installation, the following screen will pop up and the TX2 will launch. Keep it open and setup the TX2 first.

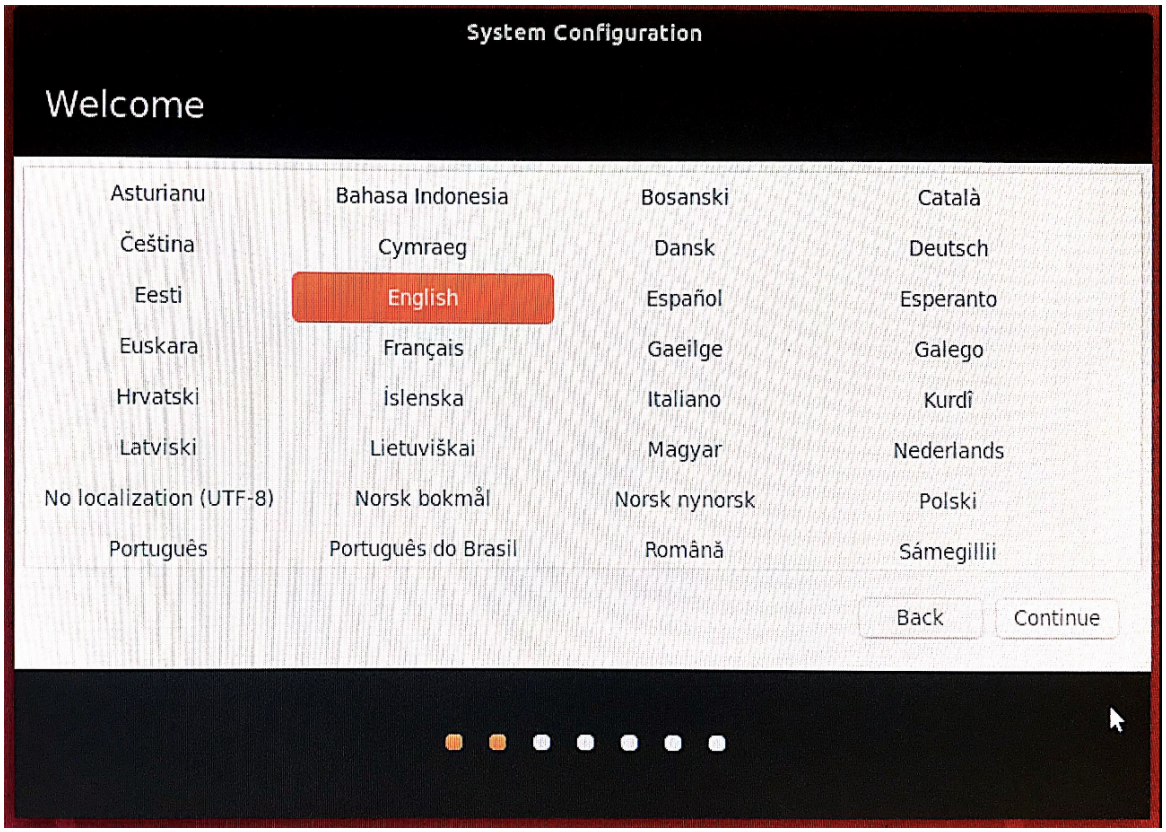


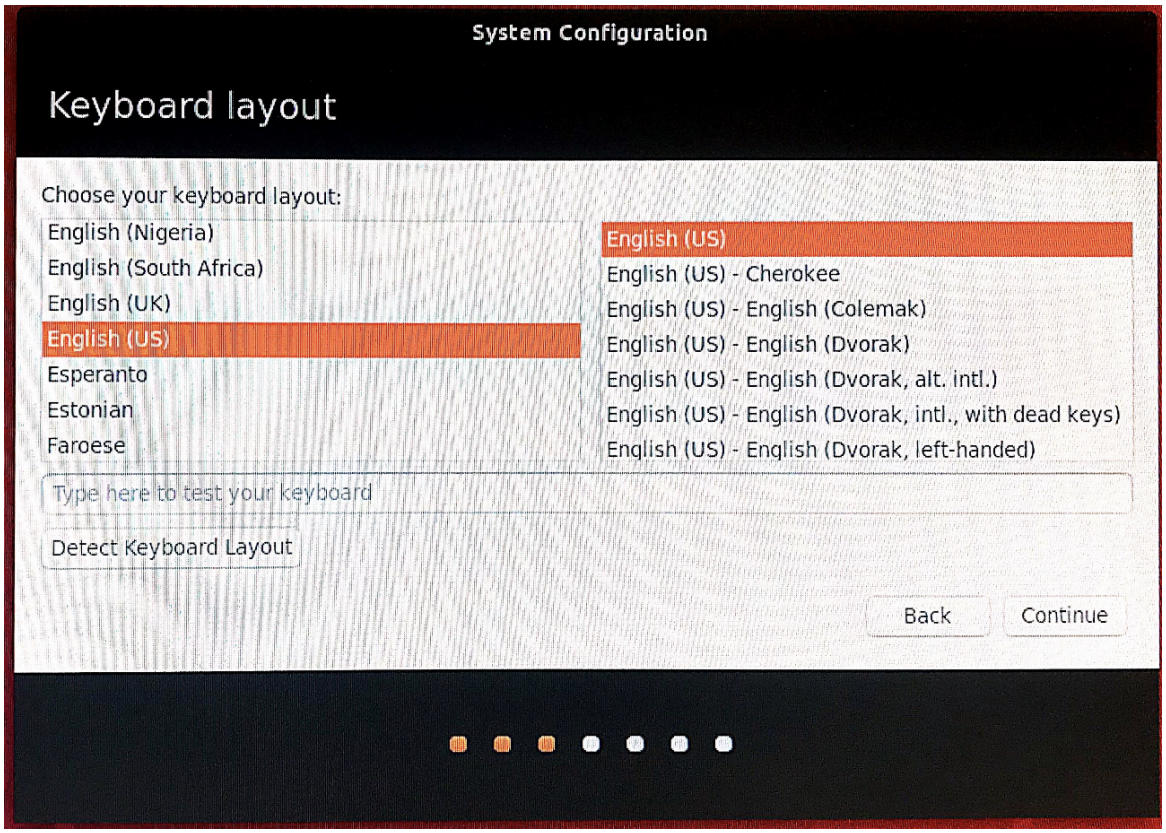
The figure below shows the TX2 launching. Follow the instructions as shown.

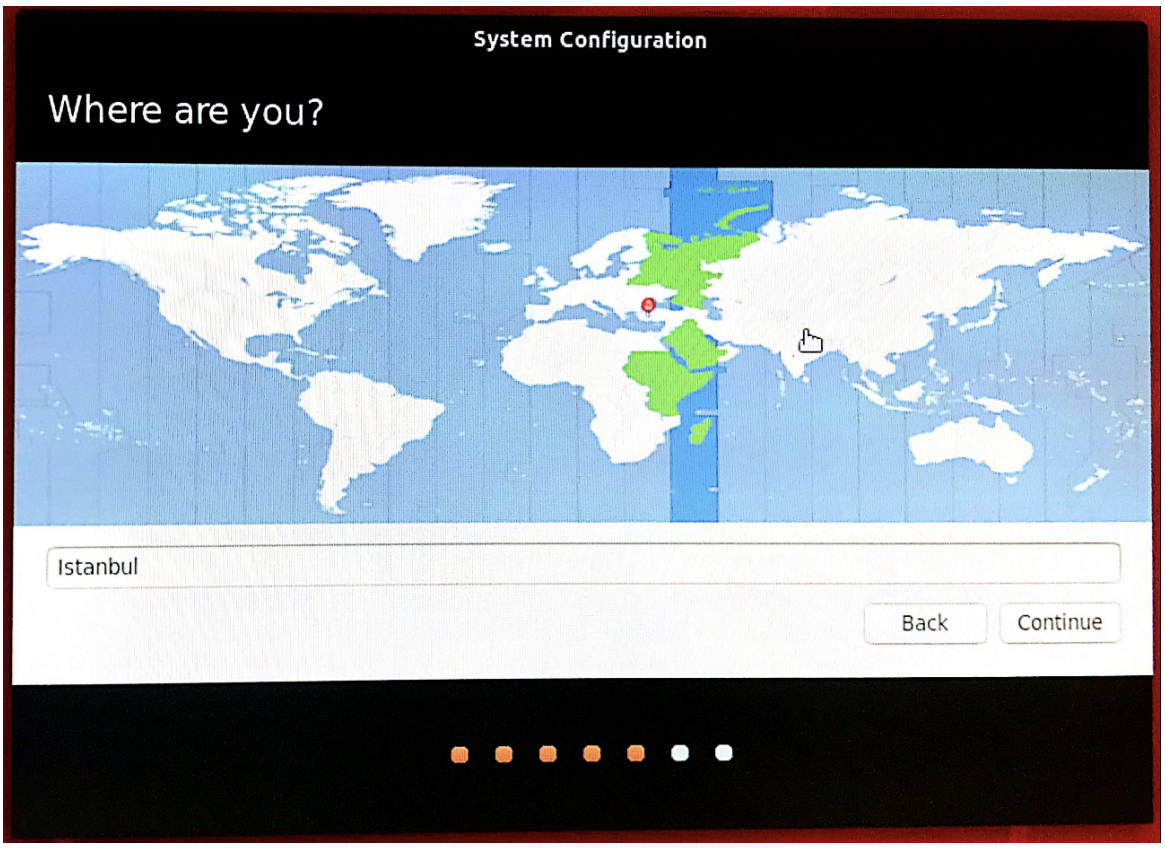




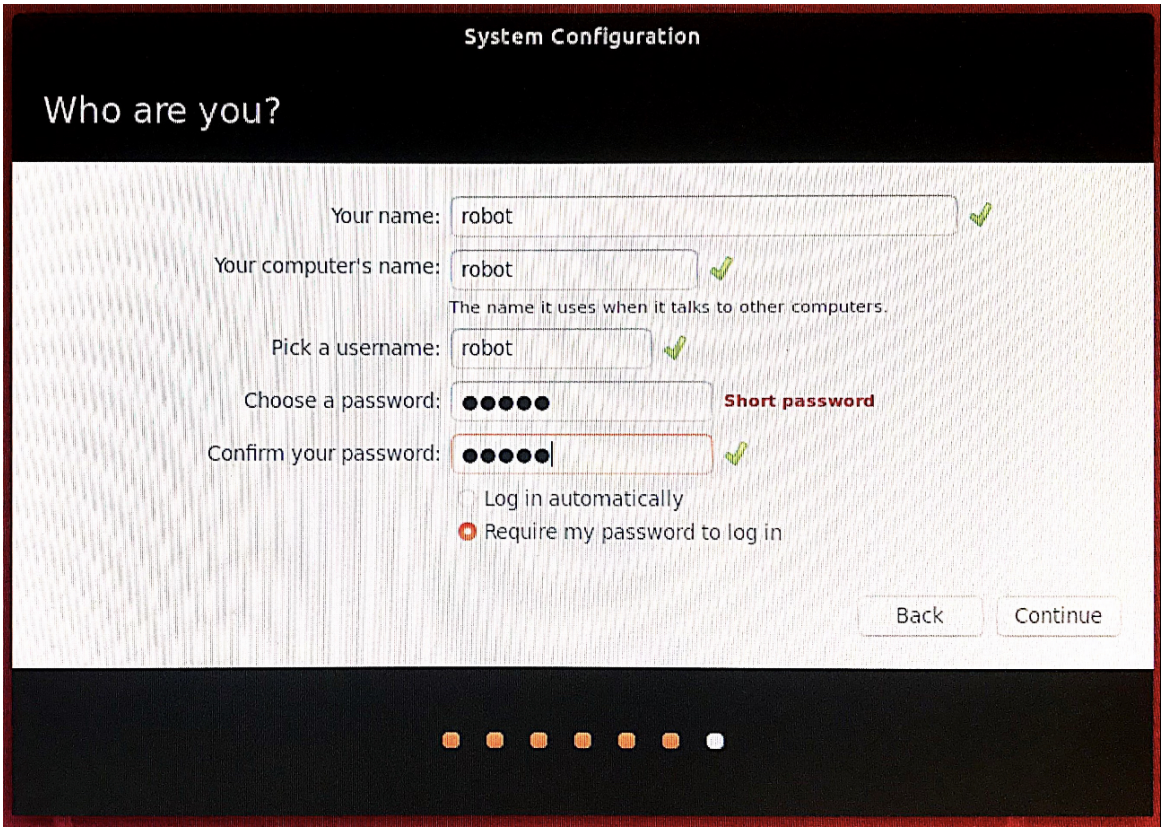
Select the language and location according to your preferences.



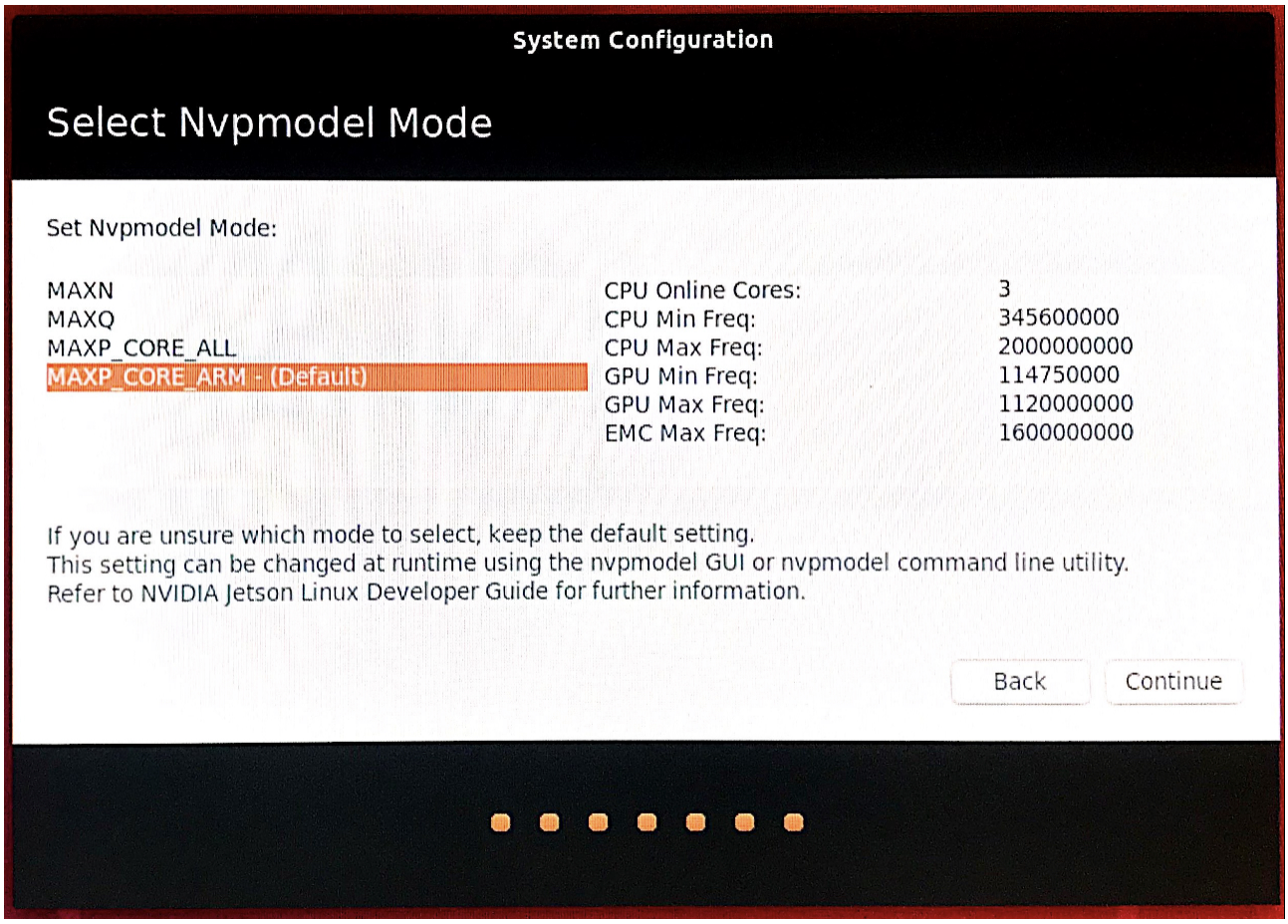




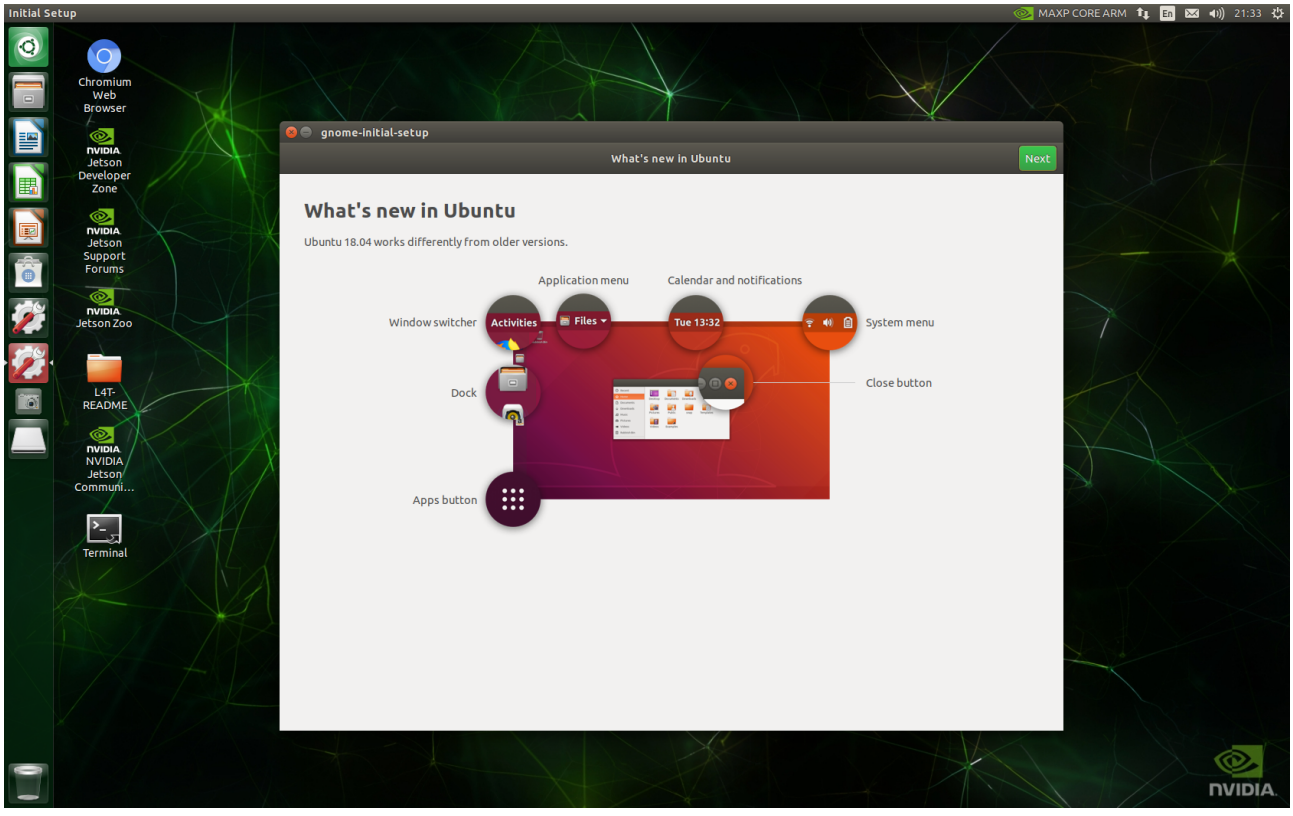
Type in the desired username and password. These are the same username and password that the host machine is waiting for!



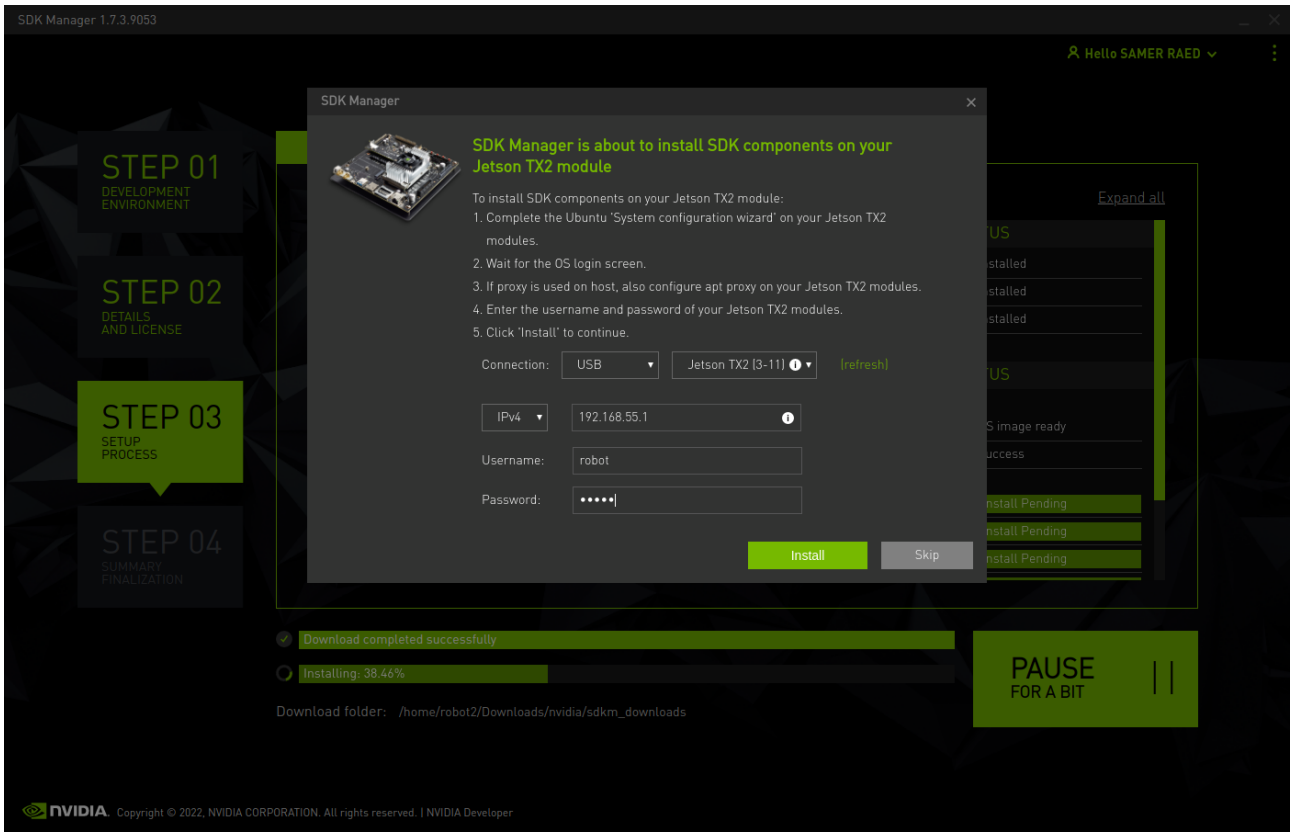
Select the default Nvpmodel mode. You can change it later.

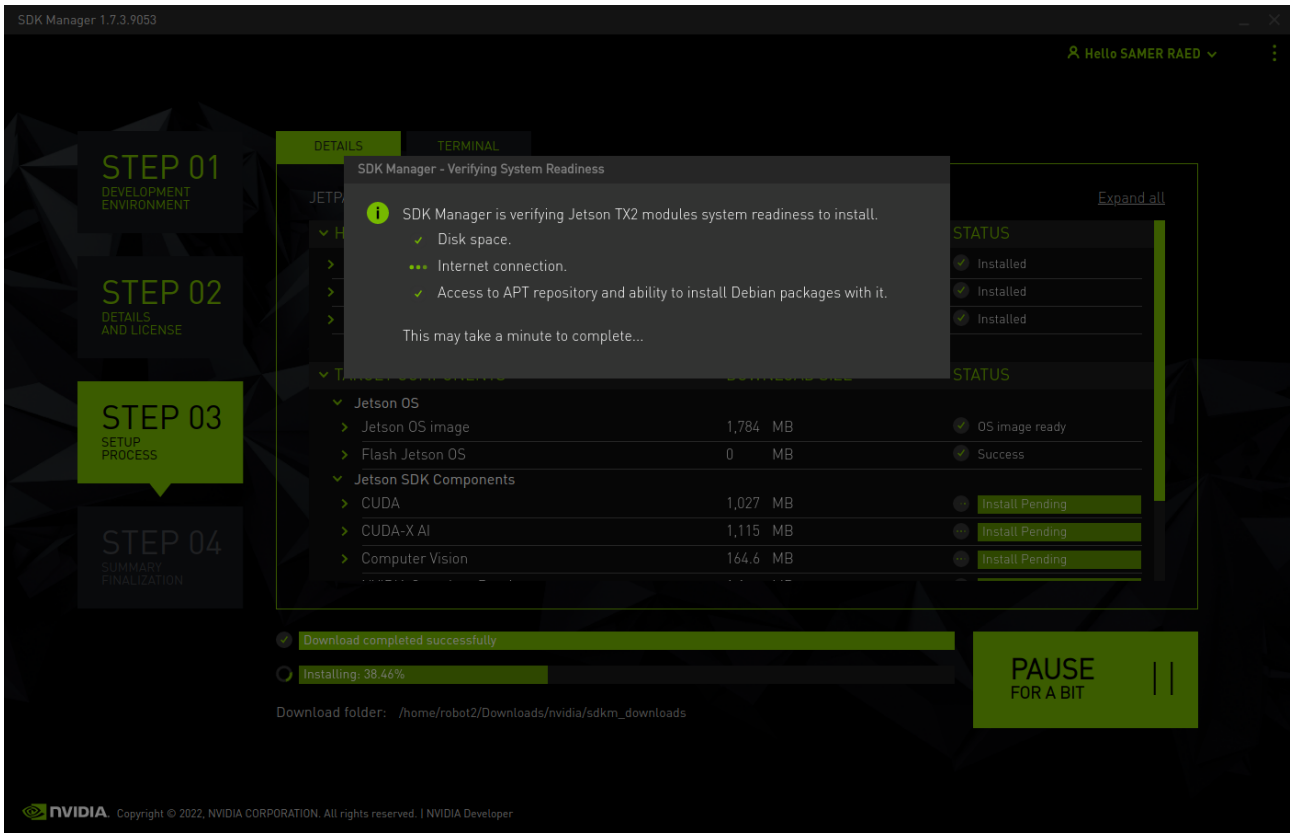


Now, the TX2 is almost ready! One more step remaining.

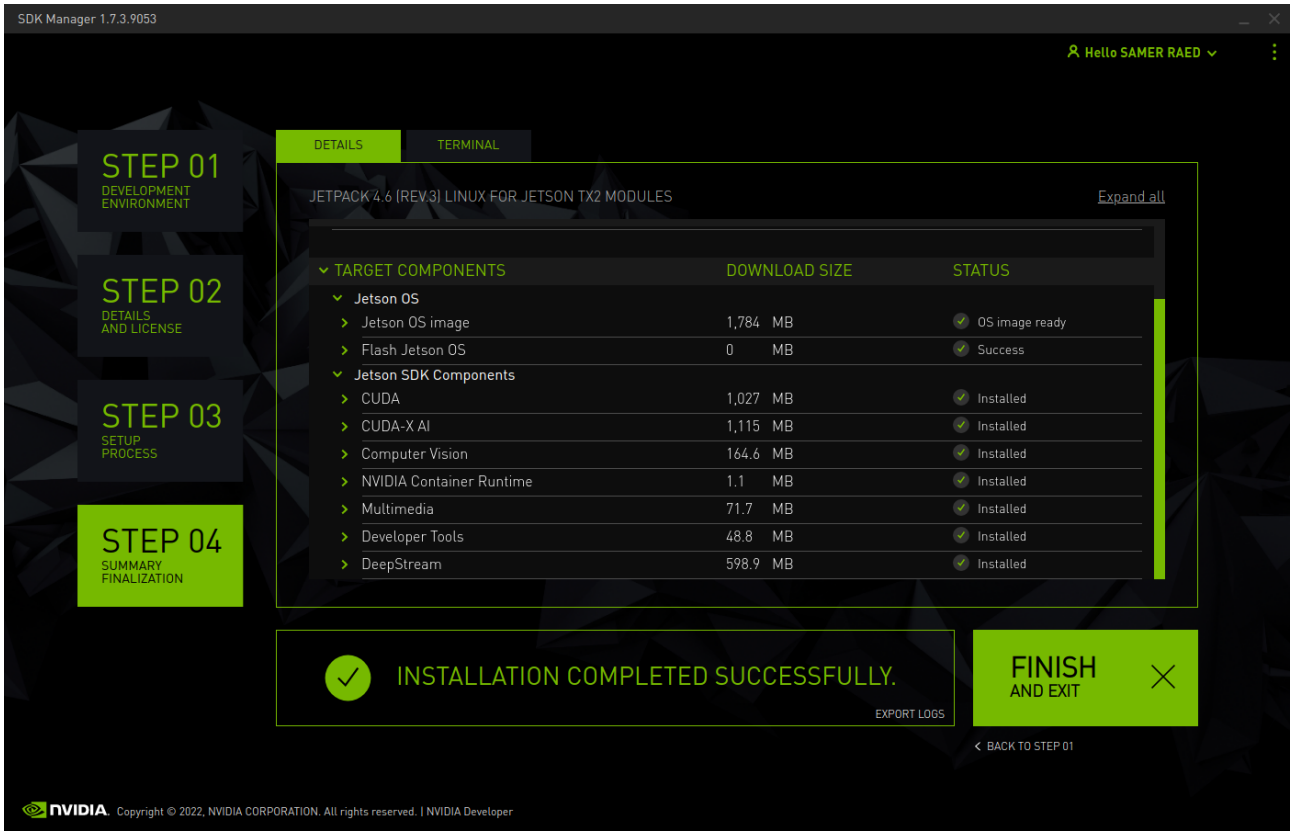


Type in your username and password to complete the installation.





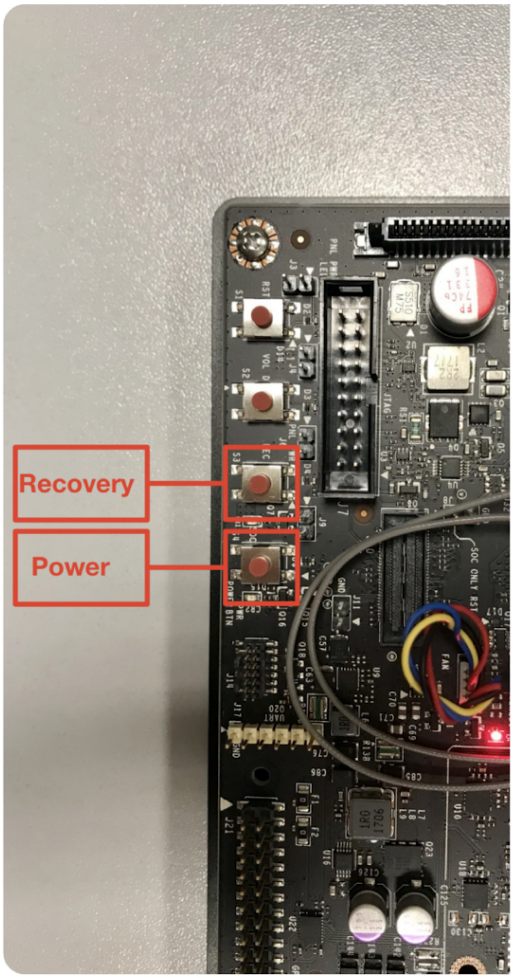
Now, you are fully ready to use one of the most powerful and energy-efficient embedded AI computer device.

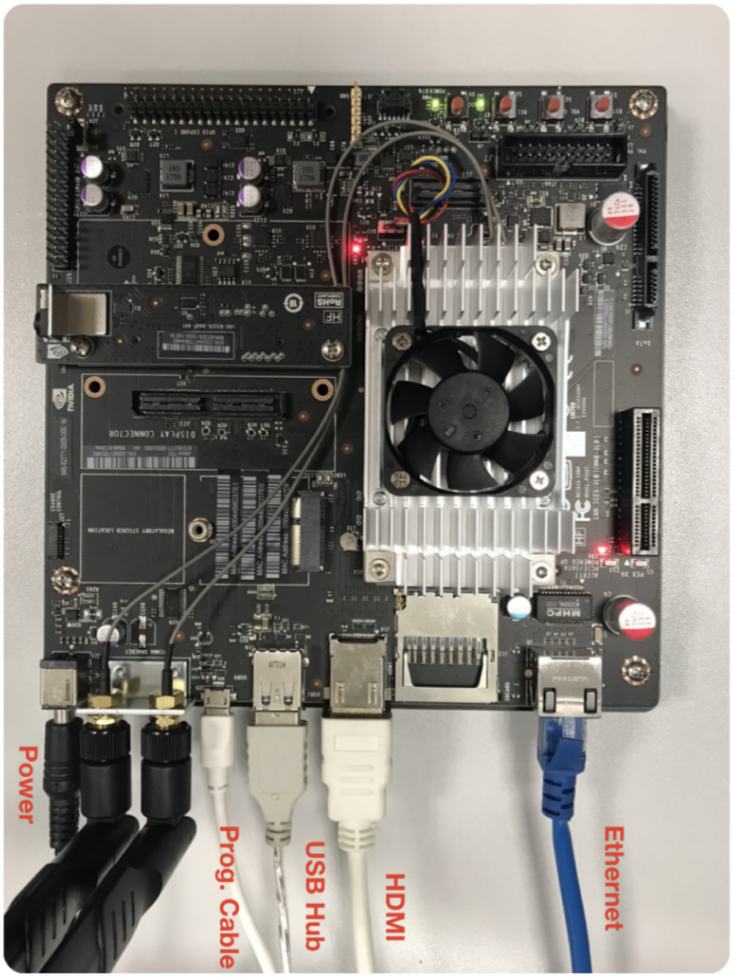


2.2 Recovery Mode

TX2 Force Recovery Mode:

1. Starting with the device powered off.
2. Establish the connections present in the figures below (Ethernet is optional).
3. USB Hub is used for keyboard and mouse connection.
4. Press and hold down the "Recovery" button.
5. Simultaneously press and hold down the "Power" button.
6. Release the Power button, then release the Force Recovery button after two seconds.





Return to the SDK Manager launching

3 Robot Operating System

The Robot Operating System (ROS) is a suite of open-source robotics middleware. Although ROS is not an operating system, but rather a collection of software frameworks for robot software development, it provides services for a heterogeneous computer cluster such as hardware abstraction, low-level device control, implementation of commonly used functionality, message-passing between processes, and package management.

3.1 Installation of ROS

Although we recommend as a starter to follow the steps from the original website in order to understand the commands printed, we will include the important commands that you will just copy-paste into your terminal for healthy installation.

```
$ sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu
  $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'

$ sudo apt install curl

$ curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.a
  c | sudo apt-key add -

$ sudo apt update

$ sudo apt install ros-melodic-desktop-full

$ echo "source /opt/ros/melodic/setup.bash" >> ~/.bashrc

$ source ~/.bashrc

$ sudo apt install python-rosdep python-rosinstall
  python-rosinstall-generator python-wstool build-essential

$ sudo apt install python-rosdep

$ sudo rosdep init

$ rosdep update
```

3.2 Creation of ROS Workspace

```
$ mkdir -p ~/catkin_ws/src

$ cd ~/catkin_ws/

$ catkin_make

$ echo "source ~/catkin_ws/devel/setup.bash" >> ~/.bashrc

$ source ~/.bashrc
```

4 Orbbec Astra Pro Camera Setup & Application

Step 1) Install ROS driver for Orbbec 3D cameras.

Again you can simply follow the steps from the source, however, for simplicity the code is presented below:

```
$ sudo apt install ros-*-rgbd-launch ros-*-libuvc
  ros-*-libuvc-camera ros-*-libuvc-ros

$ cd ~/catkin_ws/src

$ git clone https://github.com/orbbec/ros_astra_camera

$ roscd astra_camera

$ ./scripts/create_udev_rules

$ cd ~/catkin_ws/

$ catkin_make --pkg astra_camera -DFILTER=ON
```

In order to connect the camera to ROS succesfully you must connect the Camera with the main USB port (since /camera/driver/device_id: #1) hence we switched into a Bluetooth compatible keyboard and mouse. Then launch ROS:

```
$ roscore
```

And on another terminal (CTRL+ALT+T) launch the ROS driver:

```
$ roslaunch astra_camera astra.launch
```

Voila! The Astra Camera is now publishing ROS_msgs into ROS_topics. Exit the program by CTRL+C. **Note**: If you get a calibration warning you may follow the steps on this site to perform calibration. [sudo apt install ros-melodic-freenect-launch for Kinect 360 Sensor]

Step 2) Install OpenCV from the Source.

Itseez maintains the OpenCV image processing library, which was built by Intel and then sponsored by Willow Garage. OpenCV is an abbreviation for Intel® Open Source Computer Vision Library. It is a set of C functions and a few C++ classes that implement various well-known Image Processing and Computer Vision methods. OpenCV is available for Mac OS X, Windows, and Linux (Terminal environment).

Note: The following command lines are modified for our version compatibility.
-Line 4: Change "python3.5-dev" to "python3.7-dev" or any new version of python.
-Line 5: Change "libjasper-dev" to "jasper" since Jetson products are not compatible with "libjasper-dev" package.
-Line 13: Change "WITH_CUDA=OFF" to "WITH_CUDA=ON" to allow the applications be run on the GPU Processing unit instead of CPU's.
-Line 19: Change "opencv" to "opencv4" since latest linux installations does not support old versions of OpenCV.

```
$ sudo apt-get update

$ sudo apt-get upgrade

$ sudo apt-get install build-essential cmake git libgtk2.0-dev pkg-config
  libavcodec-dev libavformat-dev libswscale-dev

$ sudo apt-get install python3.7-dev python3-numpy libtbb2 libtbb-dev

$ sudo apt-get install libjpeg-dev libpng-dev libtiff5-dev jasper
  libdc1394-22-dev libeigen3-dev libtheora-dev libvorbis-dev
  libxvidcore-dev libx264-dev sphinx-common libtbb-dev yasm libfaac-dev
  libopencore-amrnb-dev libopencore-amrwb-dev libopenexr-dev
  libgstreamer-plugins-base1.0-dev libavutil-dev libavfilter-dev
  libavresample-dev

$ sudo -s

# cd ~/opt

# git clone https://github.com/Itseez/opencv.git

# git clone https://github.com/Itseez/opencv_contrib.git

# cd opencv

# mkdir release

# cd release

# cmake -D BUILD_TIFF=ON -D WITH_CUDA=ON -D ENABLE_AVX=OFF -D
  WITH_OPENGL=OFF -D WITH_OPENCL=OFF -D WITH_IPP=OFF -D WITH_TBB=ON -D
  BUILD_TBB=ON -D WITH_EIGEN=OFF -D WITH_V4L=OFF -D WITH_VTK=OFF -D
  BUILD_TESTS=OFF -D BUILD_PERF_TESTS=OFF -D CMAKE_BUILD_TYPE=RELEASE -D
  CMAKE_INSTALL_PREFIX=/usr/local -D
  OPENCV_EXTRA_MODULES_PATH=/opt/opencv_contrib/modules /opt/opencv/

# make -j4

# make install

# ldconfig

# exit

$ cd ~

$ pkg-config --modversion opencv4 #TO GET VERSION OF OpenCV
```

The process could take up to 8 hours! So patience is the key...

Step 3) Install cv_bridge.

Packages for interfacing ROS with OpenCV, a library of programming functions for real time computer vision.

Simply apply:

```
$ cd ~/catkin_ws/src

$ git clone https://github.com/ITUROBLAB/vision_opencv-melodic.git

$ cd ..

$ catkin_make
```

The non-modified repository was sourced from here.

4.1 OpenCV_Apps

opencv_apps provides various nodes that run internally OpenCV's functionalities and publish the result as ROS topics. With opencv_apps, you can skip writing OpenCV application codes for a lot of its functionalities by simply running a launch file that corresponds to OpenCV's functionality you want.
Some of the features covered by opencv_apps are explained in the wiki.

The following should be done for installation:

```
$ cd ~/catkin_ws/src

$ git clone https://github.com/ITUROBLAB/opencv_apps-contrib.git

$ cd ..

$ catkin_make
```

let us learn how to run these applications...
Firstly:

```
$ gnome-terminal --tab -- roscore

$ gnome-terminal --tab -- roslaunch astra_camera astra.launch
```

After that we need to forward the camera's info and image topics to "/camera_info" and "/image" topics respectively. For Orbbec Astra Pro the command would be:

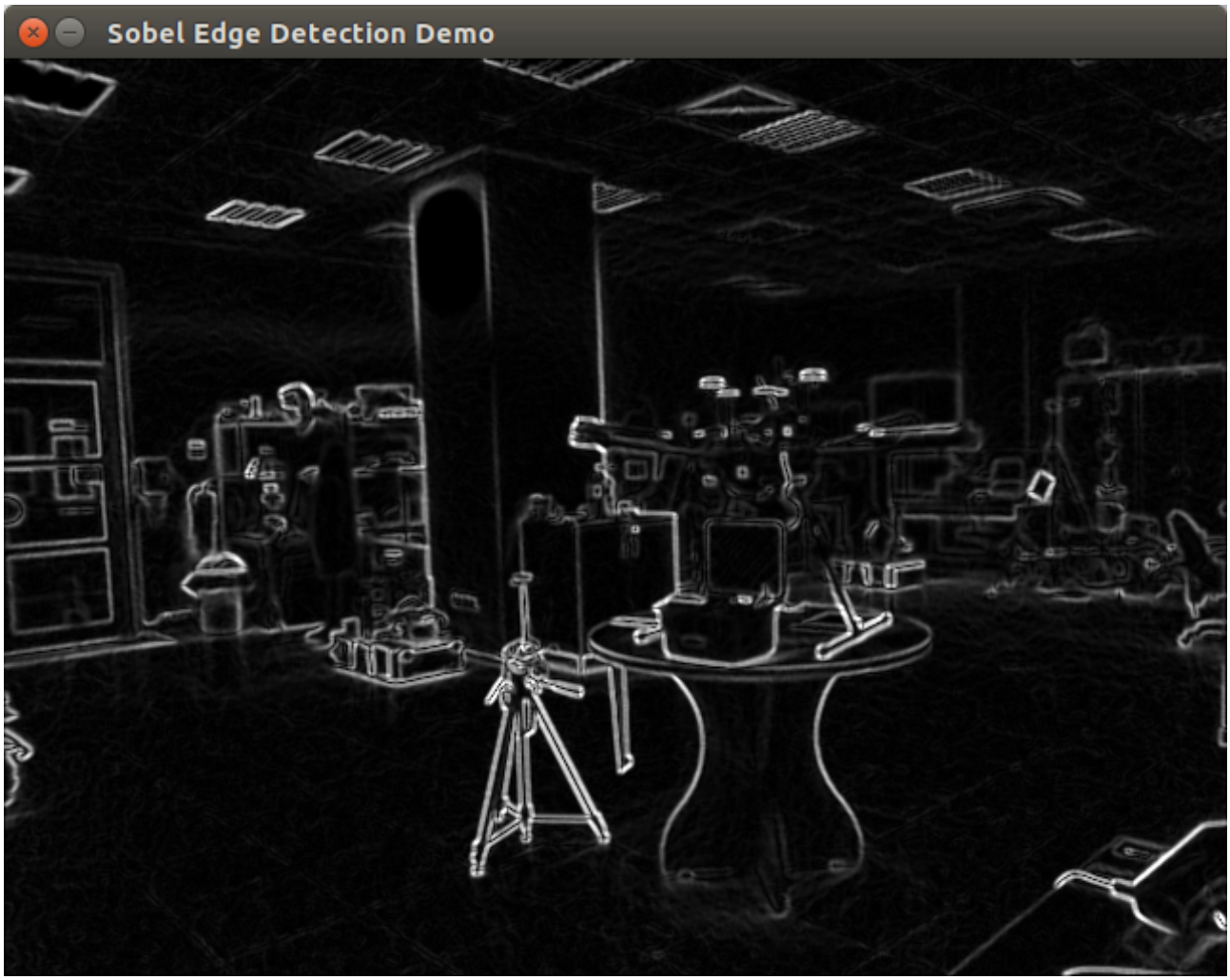
```
$ gnome-terminal --tab -- rosrun topic_tools relay
  /camera/rgb/camera_info /camera_info

$ gnome-terminal --tab -- rosrun topic_tools relay
  /camera/rgb/image_raw /image
```

Let us try the edge detection as an example:

```
$ gnome-terminal --tab -- roslaunch opencv_apps edge_detection.launch
```

Output Screen:



4.2 YOLO: Real-Time Object Detection

This is a ROS package developed for object detection in camera images. You only look once (YOLO) is a state-of-the-art, real-time object detection system. In the following ROS package you are able to use YOLO (V3) on GPU and CPU. The pre-trained model of the convolutional neural network is able to detect pre-trained classes including the data set from VOC and COCO, or you can also create a network with your own detection objects. For more information about YOLO, Darknet, available training data and training YOLO see the following link: YOLO: Real-Time Object Detection.

In order to install darknet_ros, clone the latest version using SSH (see how to set up an SSH key) from this repository into your catkin workspace and compile the package using ROS.

```
$ cd ~/catkin_ws/src

$ git clone --recursive git@github.com:leggedrobotics/darknet_ros.git

$ cd ..

$ catkin_make -DCMAKE_BUILD_TYPE=Release
```

Let us launch YOLO on the Astra Camera:

```
$ gnome-terminal --tab -- roscore

$ gnome-terminal --tab -- roslaunch astra_camera astra.launch

$ gnome-terminal --tab -- roslaunch darknet_ros darknet_ros.launch
```

Output Screen: