JetPack, Scikit-learn Installation

RTLab Seongsu Keum

CONTENTS

1 JetPack Installation

- host install
- target install
- car detect sample

2 Scikit-learn Installation

- numpy, scipy, matplotlib
- scikit learn

3 NVIDIA Two Days to a Demo

- ...

JetPack

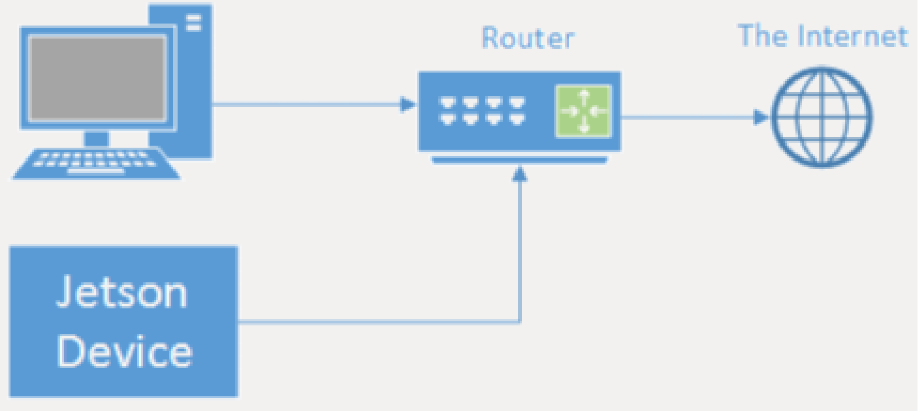
NVIDIA JetPack SDK is the most comprehensive solution for building AI applications. Use the JetPack installer to flash your Jetson Developer Kit with the latest OS image, install developer tools for both host PC and Developer Kit, and install the libraries and APIs, samples, and documentation needed to jumpstart your development environment.

Key Features in JetPack

- TensorRT
- cuDNN
- CUDA
- VisionWorks/OpenCV
- Multimedia API
- L4T
- Developer Tools

01 JetPack Installation

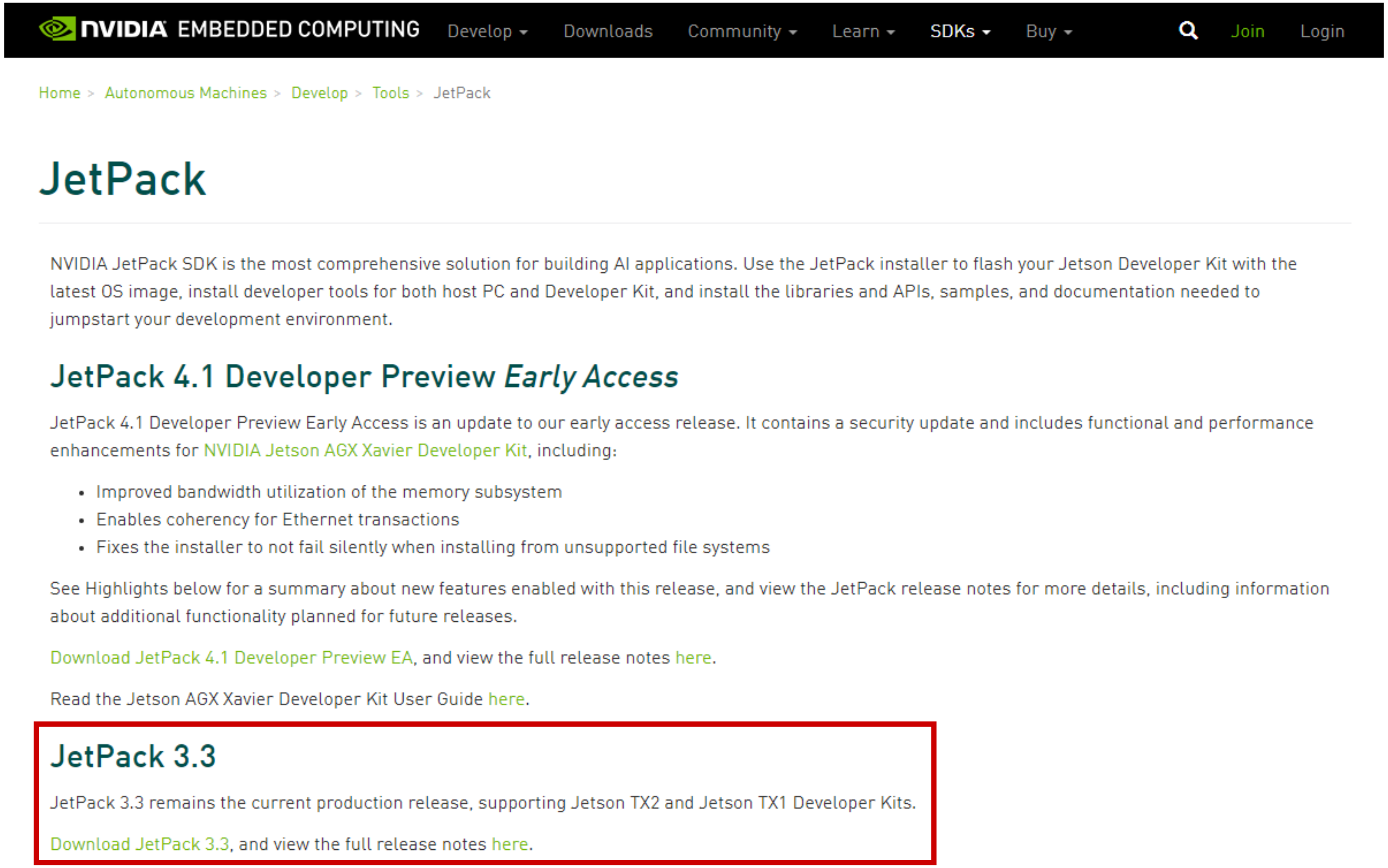
- host PC (Ubuntu 16.04)
- NVIDIA Jetson TX2
- Internet accessible router/switch



Jetpack (v3.0) Install Guide
https://www.youtube.com/watch?v=D7lkth34rgM

01 JetPack Installation

https://developer.nvidia.com/embedded/jetpack



Home > Autonomous Machines > Develop > Tools > JetPack

JetPack

NVIDIA JetPack SDK is the most comprehensive solution for building AI applications. Use the JetPack installer to flash your Jetson Developer Kit with the latest OS image, install developer tools for both host PC and Developer Kit, and install the libraries and APIs, samples, and documentation needed to jumpstart your development environment.

JetPack 4.1 Developer Preview *Early Access*

JetPack 4.1 Developer Preview Early Access is an update to our early access release. It contains a security update and includes functional and performance enhancements for NVIDIA Jetson AGX Xavier Developer Kit, including:

- Improved bandwidth utilization of the memory subsystem
- Enables coherency for Ethernet transactions
- Fixes the installer to not fail silently when installing from unsupported file systems

See Highlights below for a summary about new features enabled with this release, and view the JetPack release notes for more details, including information about additional functionality planned for future releases.

Download JetPack 4.1 Developer Preview EA, and view the full release notes here.

Read the Jetson AGX Xavier Developer Kit User Guide here.

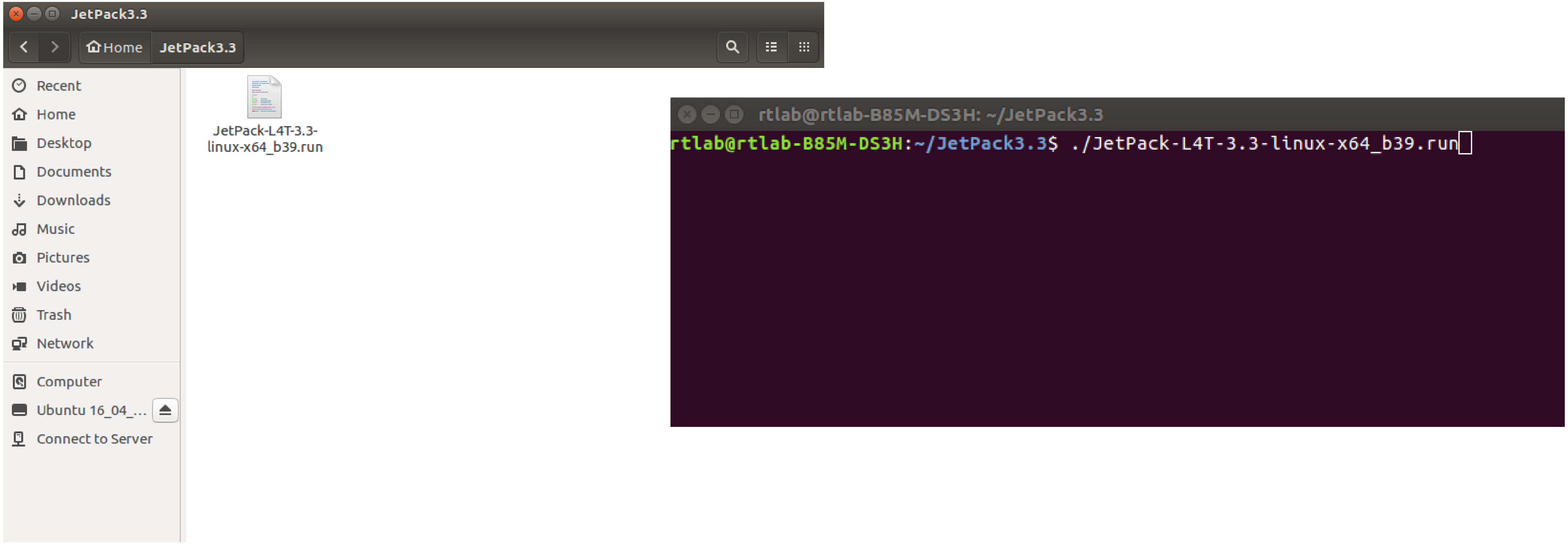
JetPack 3.3

JetPack 3.3 remains the current production release, supporting Jetson TX2 and Jetson TX1 Developer Kits.

Download JetPack 3.3, and view the full release notes here.

01 JetPack Installation

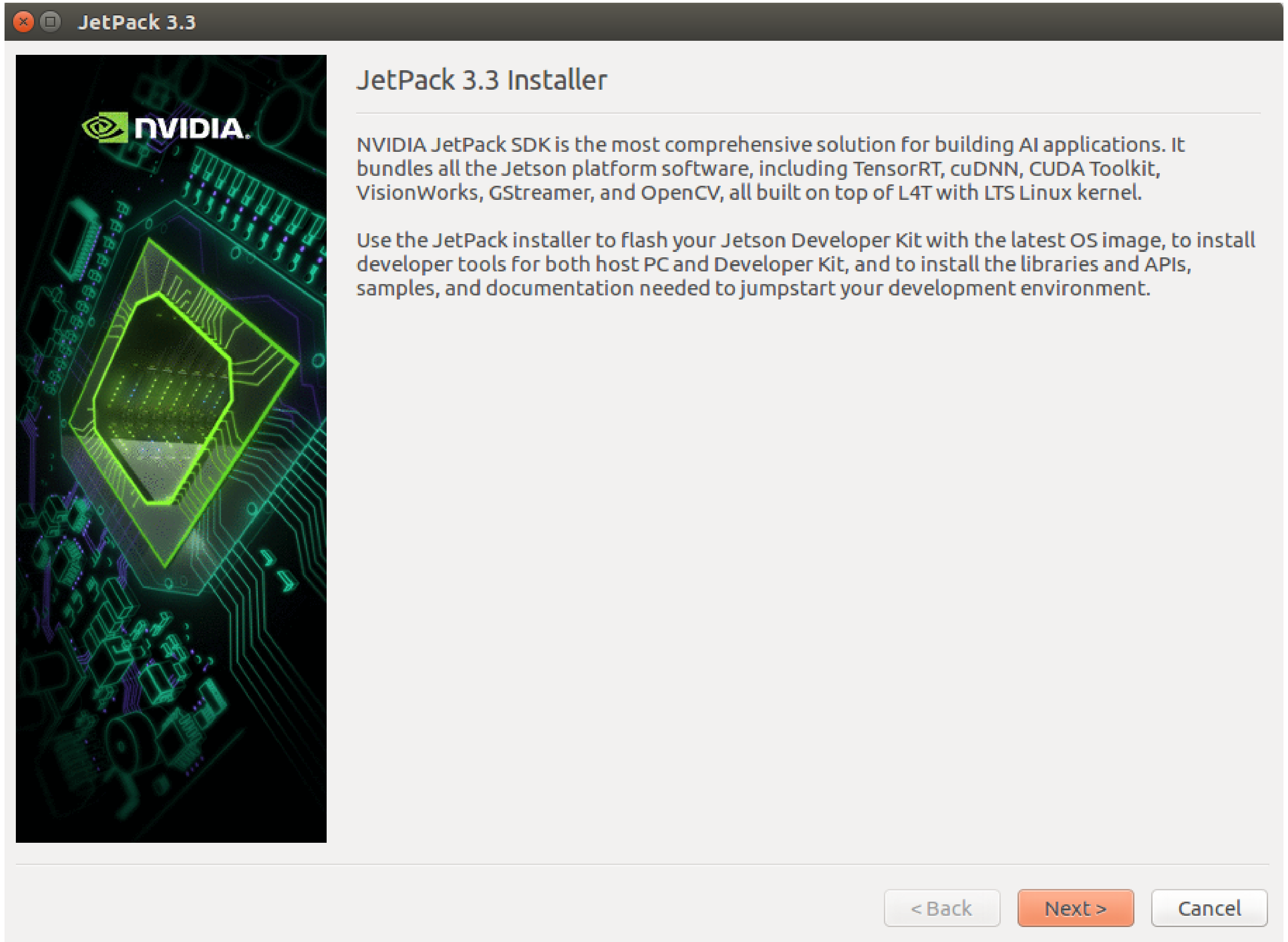
Download JetPack



```
rtlab@rtlab-B85M-DS3H:~/JetPack3.3$ ./JetPack-L4T-3.3-linux-x64_b39.run
```

01 JetPack Installation

Jetpack Installer



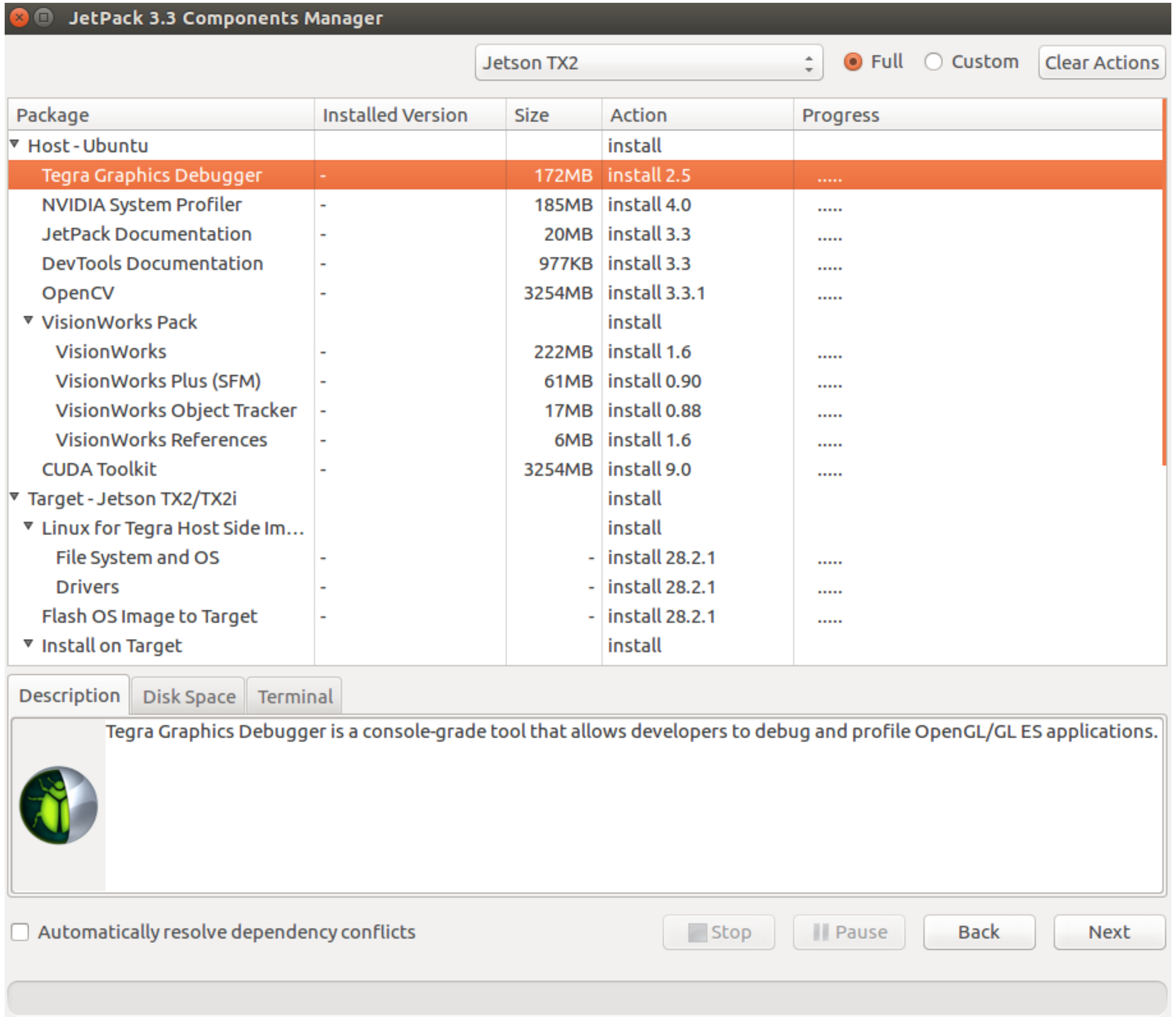
JetPack 3.3 Installer

NVIDIA JetPack SDK is the most comprehensive solution for building AI applications. It bundles all the Jetson platform software, including TensorRT, cuDNN, CUDA Toolkit, VisionWorks, GStreamer, and OpenCV, all built on top of L4T with LTS Linux kernel.

Use the JetPack installer to flash your Jetson Developer Kit with the latest OS image, to install developer tools for both host PC and Developer Kit, and to install the libraries and APIs, samples, and documentation needed to jumpstart your development environment.

01 JetPack Installation

Install Packages



JetPack 3.3 Components Manager

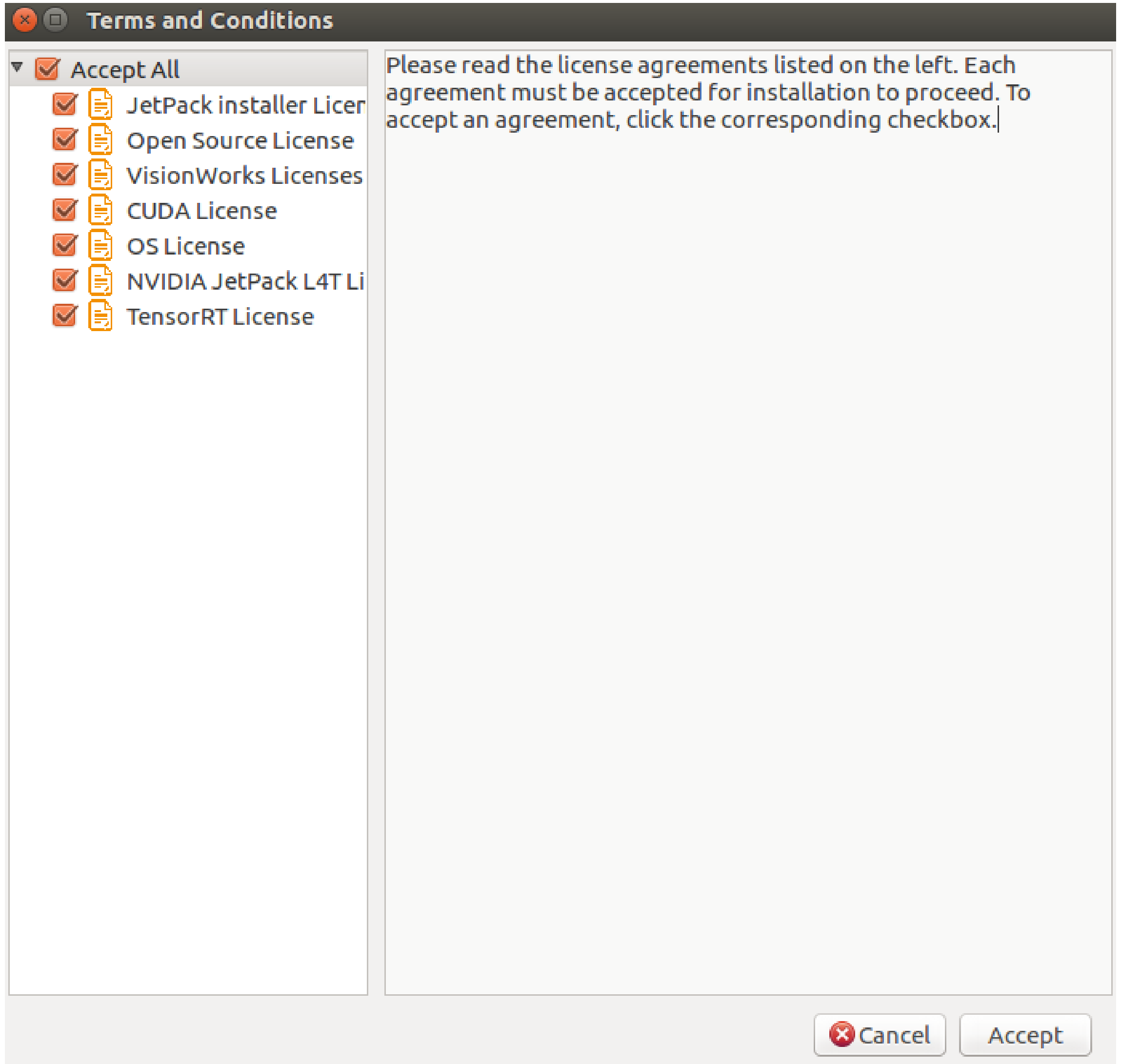
Jetson TX2 ● Full ○ Custom

Package	Installed Version	Size	Action	Progress
▾ Host - Ubuntu			install	
Tegra Graphics Debugger	-	172MB	install 2.5	.....
NVIDIA System Profiler	-	185MB	install 4.0	.....
JetPack Documentation	-	20MB	install 3.3	.....
DevTools Documentation	-	977KB	install 3.3	.....
OpenCV	-	3254MB	install 3.3.1	.....
▾ VisionWorks Pack			install	
VisionWorks	-	222MB	install 1.6	.....
VisionWorks Plus (SFM)	-	61MB	install 0.90	.....
VisionWorks Object Tracker	-	17MB	install 0.88	.....
VisionWorks References	-	6MB	install 1.6	.....
CUDA Toolkit	-	3254MB	install 9.0	.....
▾ Target - Jetson TX2/TX2i			install	
▾ Linux for Tegra Host Side Im...			install	
File System and OS	-	-	install 28.2.1	.....
Drivers	-	-	install 28.2.1	.....
Flash OS Image to Target	-	-	install 28.2.1	.....
▾ Install on Target			install	

Tegra Graphics Debugger is a console-grade tool that allows developers to debug and profile OpenGL/GL ES applications.

□ Automatically resolve dependency conflicts

Accept License



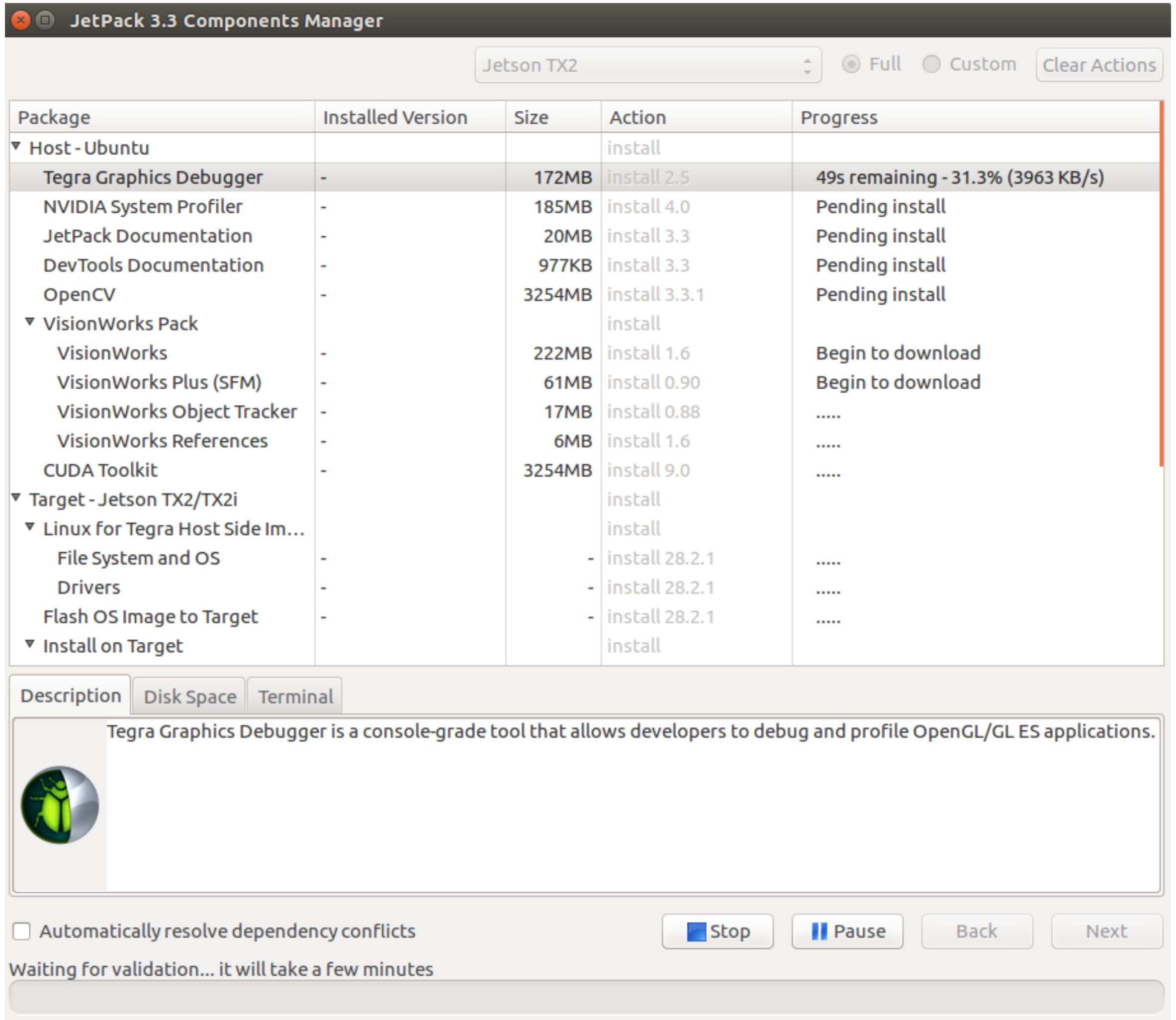
Terms and Conditions

- ▾ ☑ Accept All
  - ☑ JetPack installer Licer
  - ☑ Open Source License
  - ☑ VisionWorks Licenses
  - ☑ CUDA License
  - ☑ OS License
  - ☑ NVIDIA JetPack L4T Li
  - ☑ TensorRT License

Please read the license agreements listed on the left. Each agreement must be accepted for installation to proceed. To accept an agreement, click the corresponding checkbox.

01 JetPack Installation

Host Installation took ~15 minutes



JetPack 3.3 Components Manager

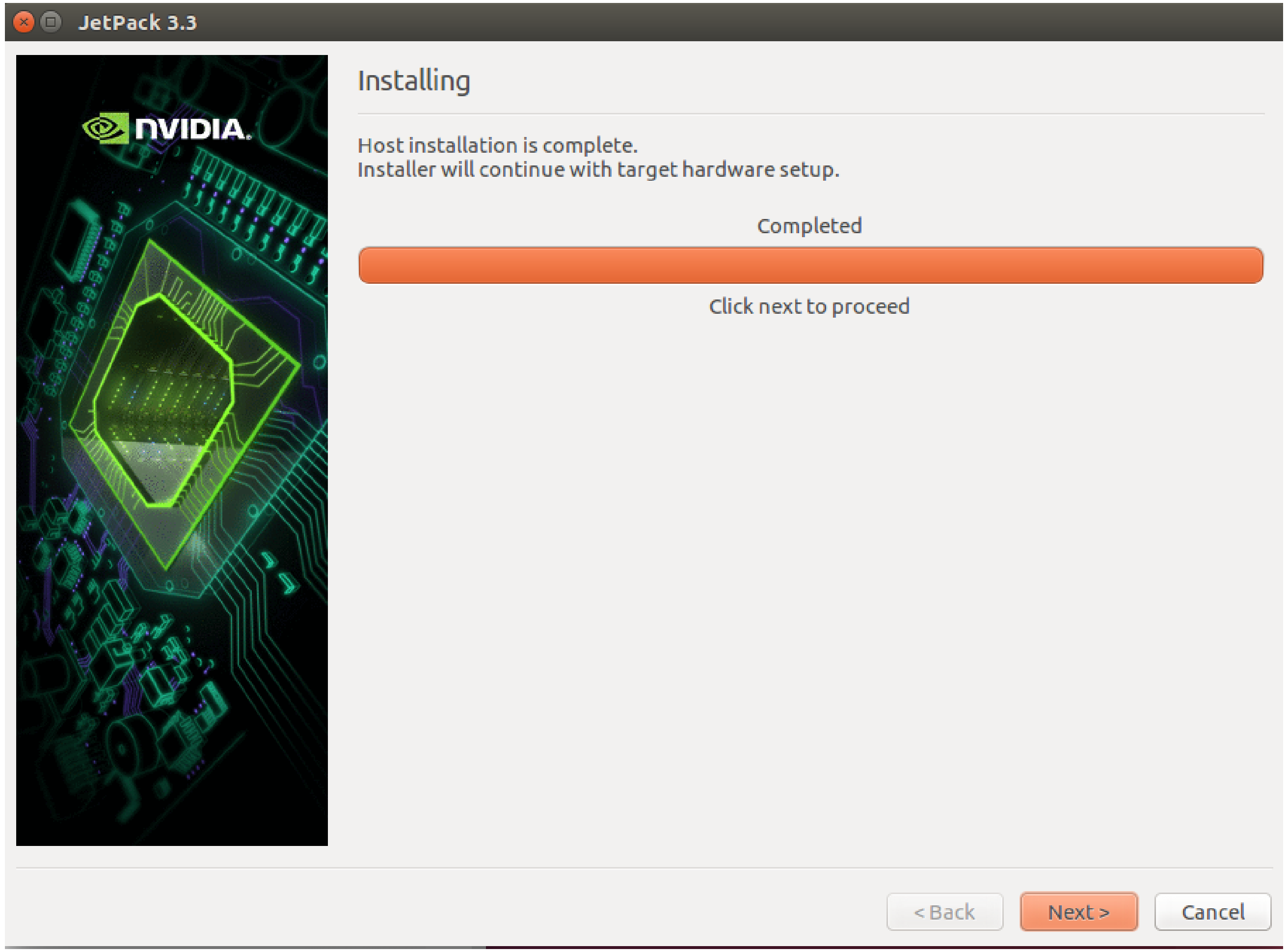
Package	Installed Version	Size	Action	Progress
▾ Host - Ubuntu			install	
Tegra Graphics Debugger	-	172MB	install 2.5	49s remaining - 31.3% (3963 KB/s)
NVIDIA System Profiler	-	185MB	install 4.0	Pending install
JetPack Documentation	-	20MB	install 3.3	Pending install
DevTools Documentation	-	977KB	install 3.3	Pending install
OpenCV	-	3254MB	install 3.3.1	Pending install
▾ VisionWorks Pack			install	
VisionWorks	-	222MB	install 1.6	Begin to download
VisionWorks Plus (SFM)	-	61MB	install 0.90	Begin to download
VisionWorks Object Tracker	-	17MB	install 0.88	.....
VisionWorks References	-	6MB	install 1.6	.....
CUDA Toolkit	-	3254MB	install 9.0	.....
▾ Target - Jetson TX2/TX2i			install	
▾ Linux for Tegra Host Side Im...			install	
File System and OS	-	-	install 28.2.1	.....
Drivers	-	-	install 28.2.1	.....
Flash OS Image to Target	-	-	install 28.2.1	.....
▾ Install on Target			install	

Tegra Graphics Debugger is a console-grade tool that allows developers to debug and profile OpenGL/GL ES applications.

□ Automatically resolve dependency conflicts

Waiting for validation... it will take a few minutes

Completion of Host Installation



JetPack 3.3

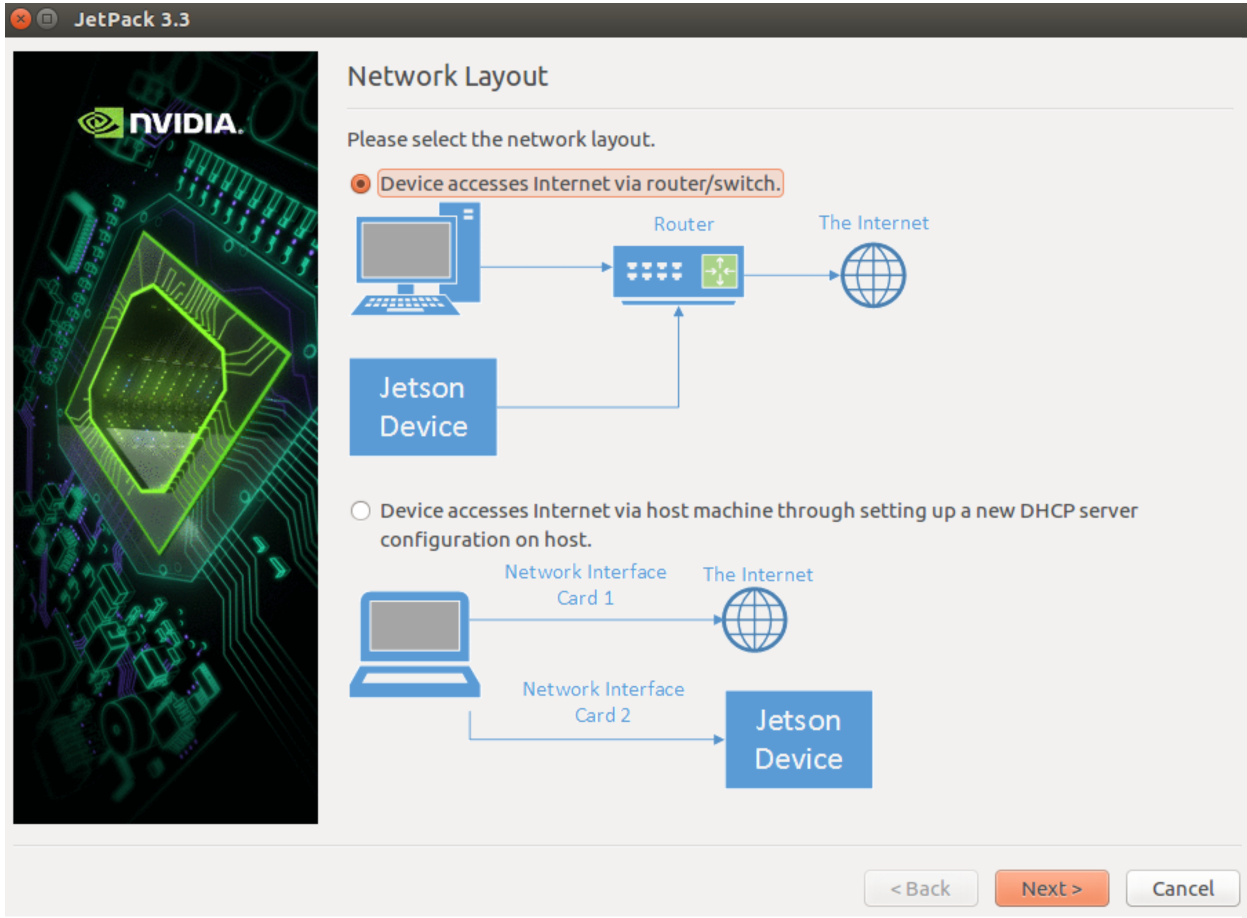
Installing

Host installation is complete.
Installer will continue with target hardware setup.

Completed

Click next to proceed

01 JetPack Installation

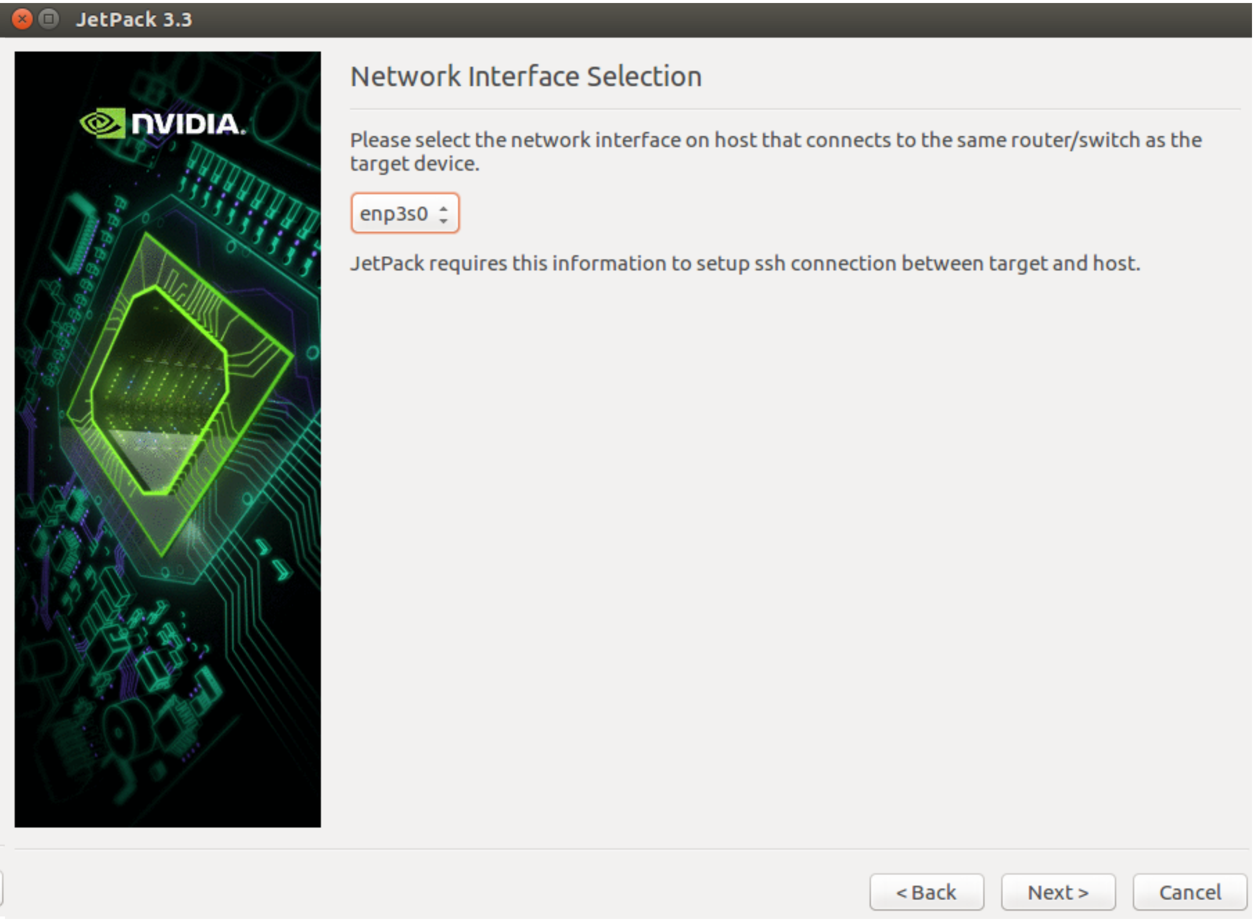


Network Layout

Please select the network layout.

● Device accesses Internet via router/switch.

○ Device accesses Internet via host machine through setting up a new DHCP server configuration on host.



Network Interface Selection

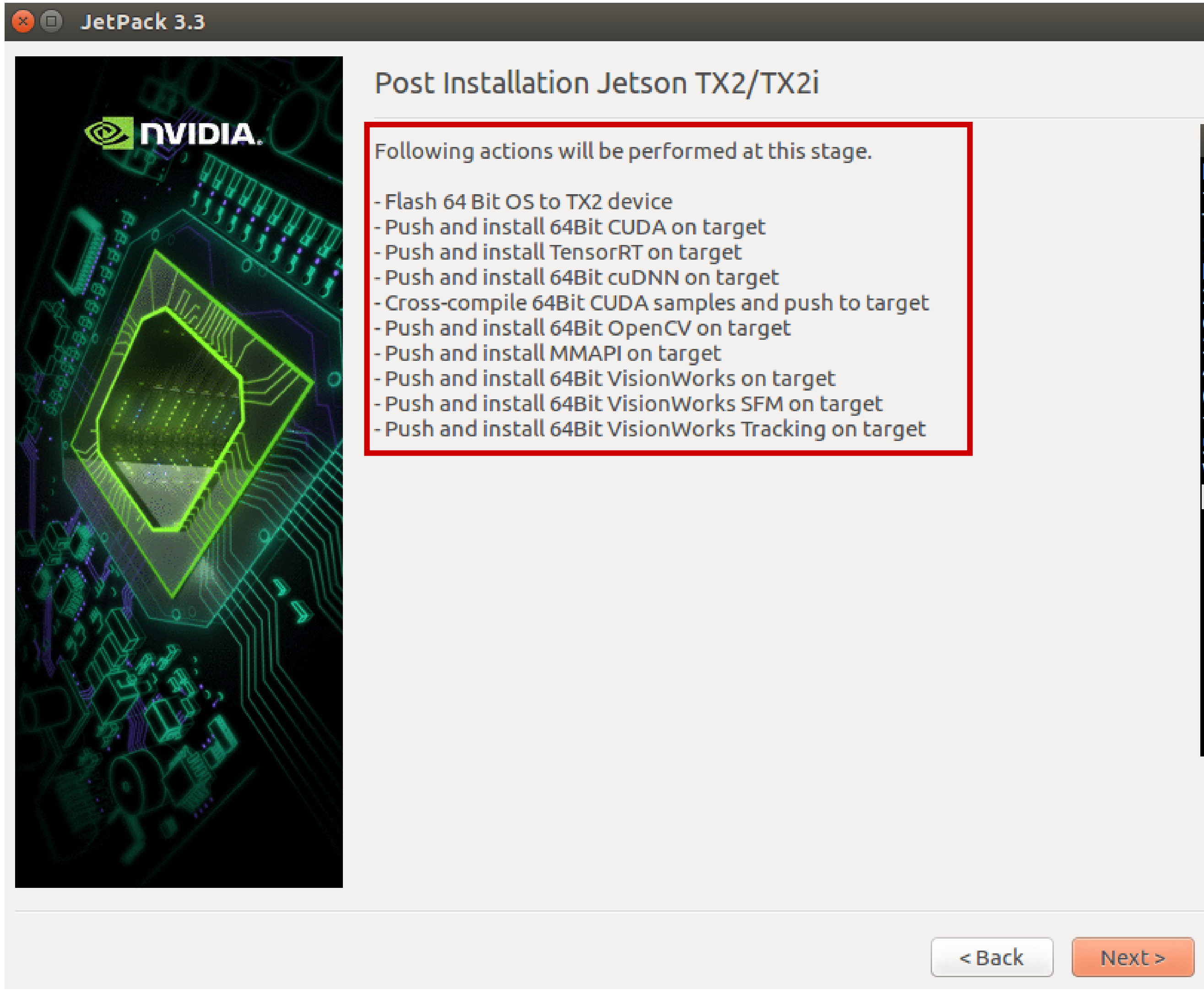
Please select the network interface on host that connects to the same router/switch as the target device.

enp3s0

JetPack requires this information to setup ssh connection between target and host.

01 JetPack Installation

Target Installation

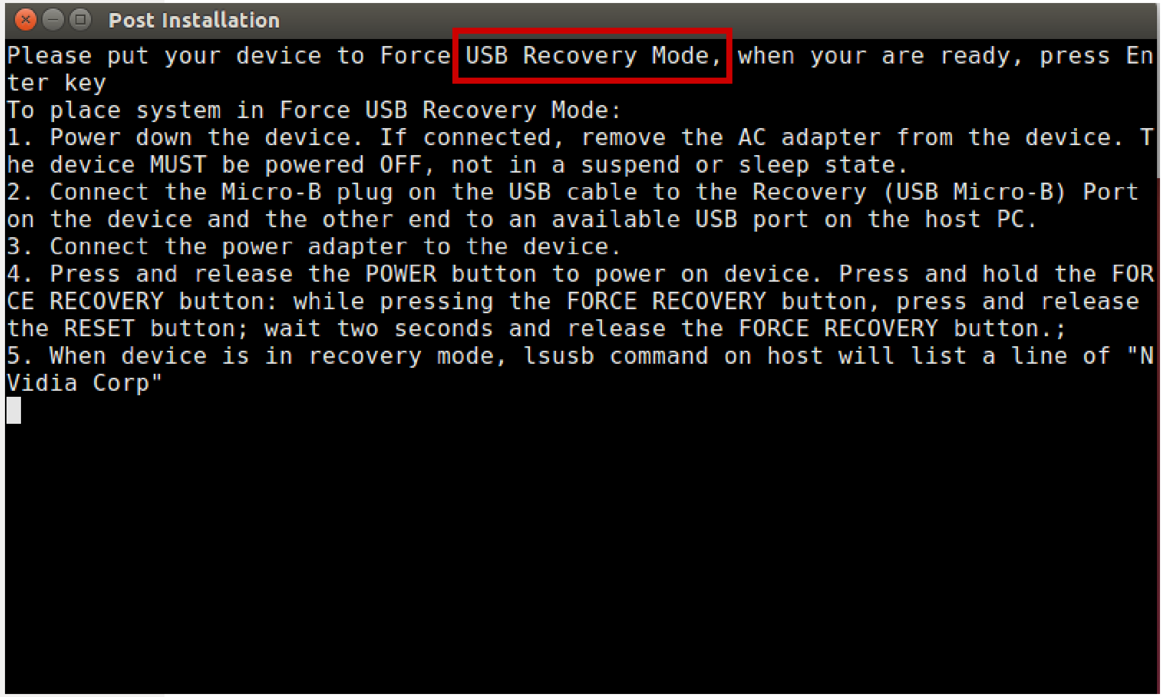


Post Installation Jetson TX2/TX2i

Following actions will be performed at this stage.

- Flash 64 Bit OS to TX2 device
- Push and install 64Bit CUDA on target
- Push and install TensorRT on target
- Push and install 64Bit cuDNN on target
- Cross-compile 64Bit CUDA samples and push to target
- Push and install 64Bit OpenCV on target
- Push and install MMAPI on target
- Push and install 64Bit VisionWorks on target
- Push and install 64Bit VisionWorks SFM on target
- Push and install 64Bit VisionWorks Tracking on target

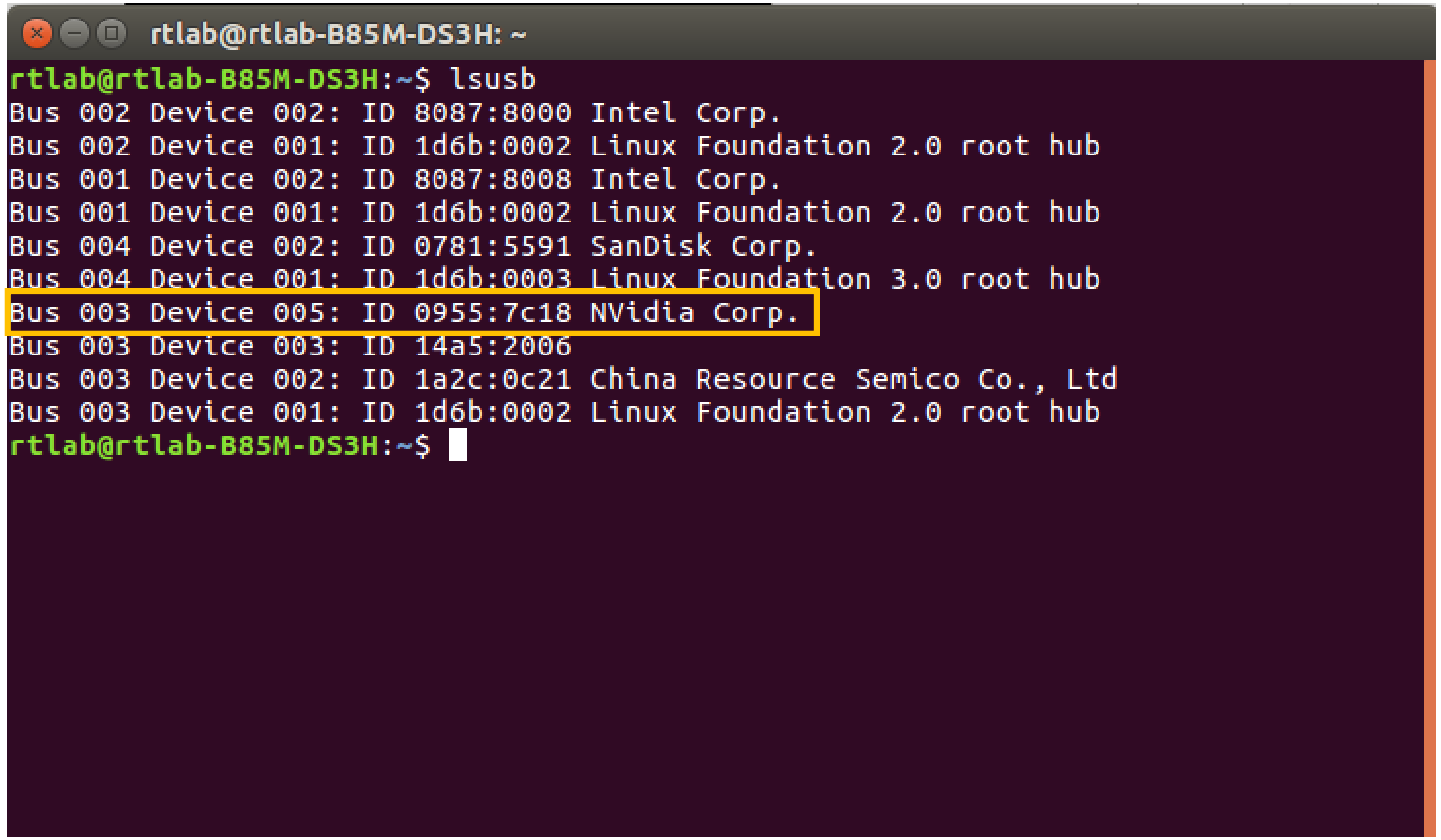
USB Recovery Mode



```
Please put your device to Force USB Recovery Mode, when your are ready, press En
ter key
To place system in Force USB Recovery Mode:
1. Power down the device. If connected, remove the AC adapter from the device. T
he device MUST be powered OFF, not in a suspend or sleep state.
2. Connect the Micro-B plug on the USB cable to the Recovery (USB Micro-B) Port
on the device and the other end to an available USB port on the host PC.
3. Connect the power adapter to the device.
4. Press and release the POWER button to power on device. Press and hold the FOR
CE RECOVERY button: while pressing the FORCE RECOVERY button, press and release
the RESET button; wait two seconds and release the FORCE RECOVERY button.;
5. When device is in recovery mode, lsusb command on host will list a line of "N
Vidia Corp"
```

01 JetPack Installation

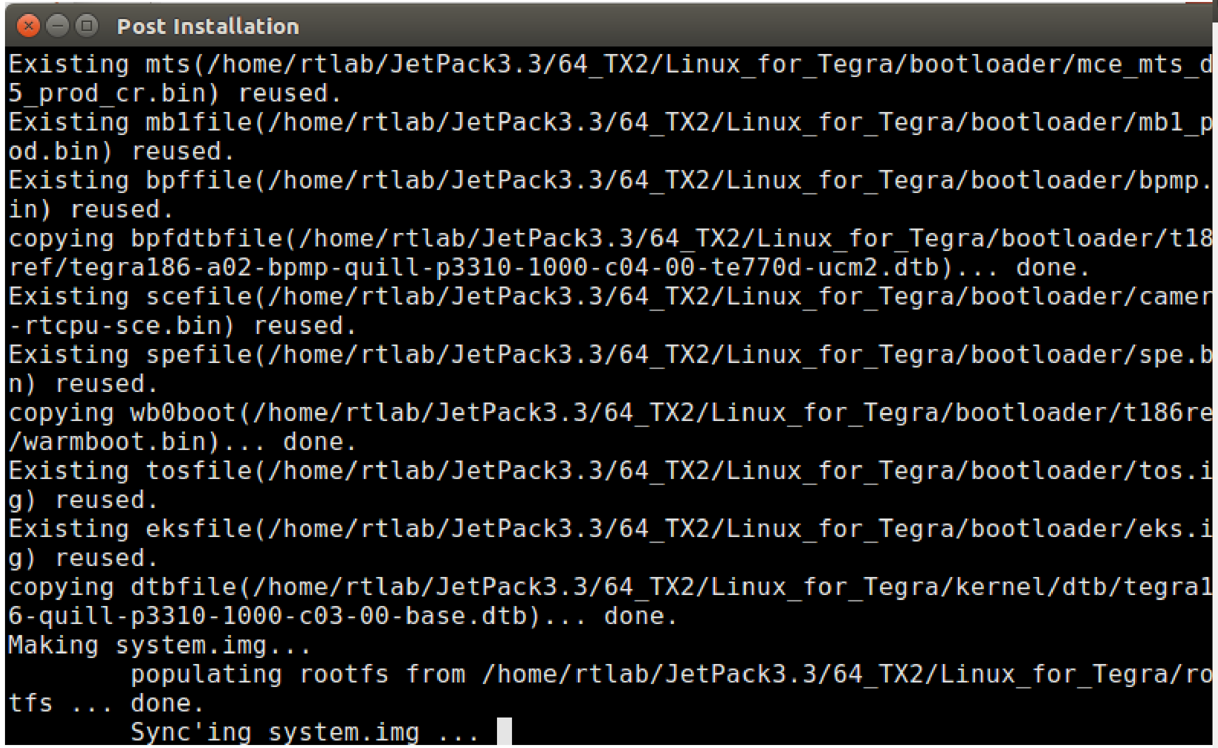
lsusb



```
rtlab@rtlab-B85M-DS3H:~$ lsusb
Bus 002 Device 002: ID 8087:8000 Intel Corp.
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 002: ID 8087:8008 Intel Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 004 Device 002: ID 0781:5591 SanDisk Corp.
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 005: ID 0955:7c18 NVidia Corp.
Bus 003 Device 003: ID 14a5:2006
Bus 003 Device 002: ID 1a2c:0c21 China Resource Semico Co., Ltd
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
rtlab@rtlab-B85M-DS3H:~$
```

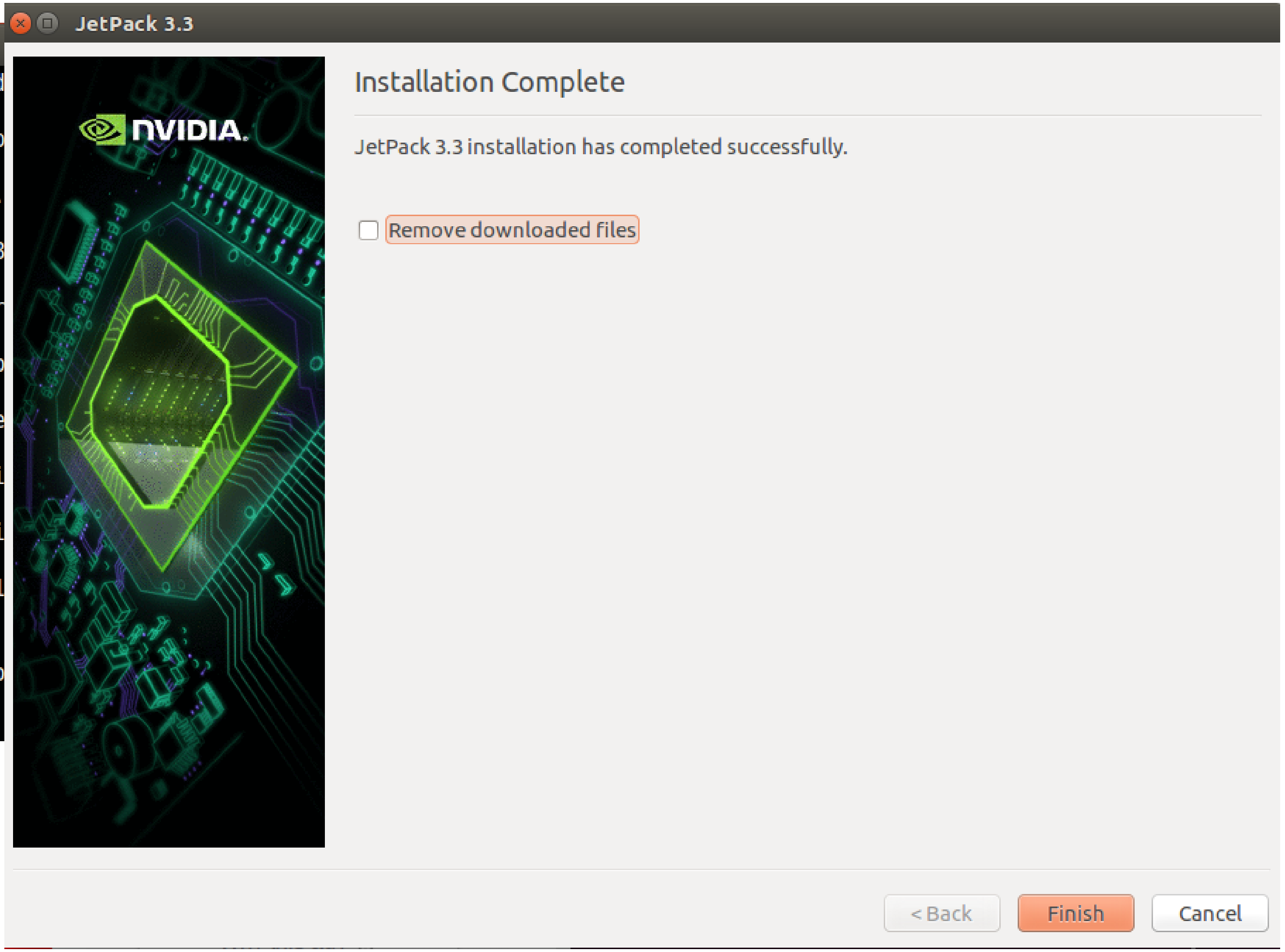
01 JetPack Installation

Building root and flashing.. ~20 minutes



```
Existing mts(/home/rtlab/JetPack3.3/64_TX2/Linux_for_Tegra/bootloader/mce_mts_d
5_prod_cr.bin) reused.
Existing mb1file(/home/rtlab/JetPack3.3/64_TX2/Linux_for_Tegra/bootloader/mb1_p
od.bin) reused.
Existing bpffile(/home/rtlab/JetPack3.3/64_TX2/Linux_for_Tegra/bootloader/bpmp.
in) reused.
copying bpfdtbfile(/home/rtlab/JetPack3.3/64_TX2/Linux_for_Tegra/bootloader/t18
ref/tegra186-a02-bpmp-quill-p3310-1000-c04-00-te770d-ucm2.dtb)... done.
Existing scefile(/home/rtlab/JetPack3.3/64_TX2/Linux_for_Tegra/bootloader/camer
-rtcpu-sce.bin) reused.
Existing spefile(/home/rtlab/JetPack3.3/64_TX2/Linux_for_Tegra/bootloader/spe.b
n) reused.
copying wb0boot(/home/rtlab/JetPack3.3/64_TX2/Linux_for_Tegra/bootloader/t186re
/warmboot.bin)... done.
Existing tosfile(/home/rtlab/JetPack3.3/64_TX2/Linux_for_Tegra/bootloader/tos.i
g) reused.
Existing eksfile(/home/rtlab/JetPack3.3/64_TX2/Linux_for_Tegra/bootloader/eks.i
g) reused.
copying dtbfile(/home/rtlab/JetPack3.3/64_TX2/Linux_for_Tegra/kernel/dtb/tegra1
6-quill-p3310-1000-c03-00-base.dtb)... done.
Making system.img...
        populating rootfs from /home/rtlab/JetPack3.3/64_TX2/Linux_for_Tegra/ro
tfs ... done.
        Sync'ing system.img ...
```

Completion of Target Installation



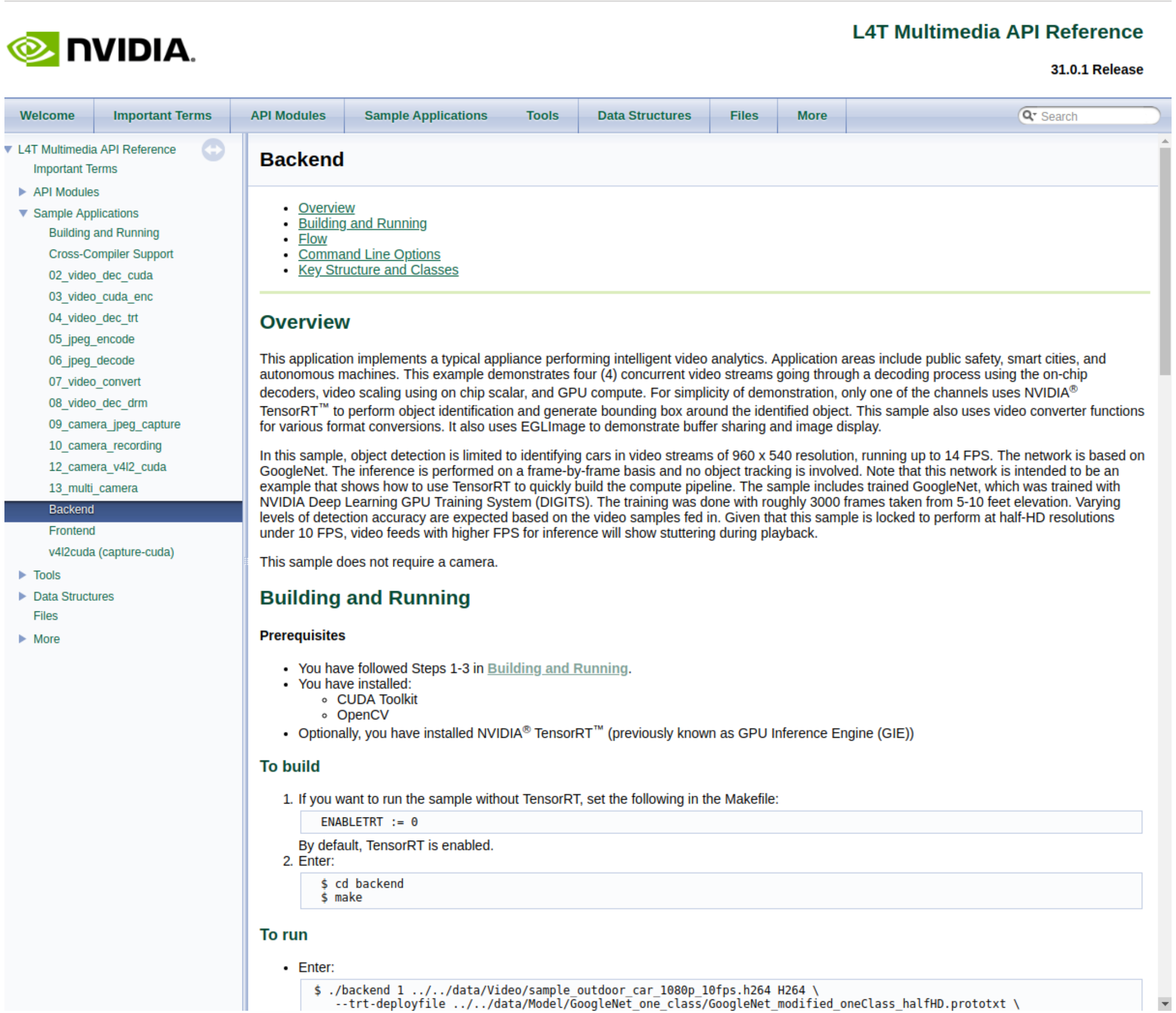
Installation Complete

JetPack 3.3 installation has completed successfully.

□ Remove downloaded files

01 JetPack Installation

file:///home/nvidia/L4T/nvl4t_docs/index.html



Backend

- Overview
- Building and Running
- Flow
- Command Line Options
- Key Structure and Classes

Overview

This application implements a typical appliance performing intelligent video analytics. Application areas include public safety, smart cities, and autonomous machines. This example demonstrates four (4) concurrent video streams going through a decoding process using the on-chip decoders, video scaling using on chip scalar, and GPU compute. For simplicity of demonstration, only one of the channels uses NVIDIA® TensorRT™ to perform object identification and generate bounding box around the identified object. This sample also uses video converter functions for various format conversions. It also uses EGLImage to demonstrate buffer sharing and image display.

In this sample, object detection is limited to identifying cars in video streams of 960 x 540 resolution, running up to 14 FPS. The network is based on GoogleNet. The inference is performed on a frame-by-frame basis and no object tracking is involved. Note that this network is intended to be an example that shows how to use TensorRT to quickly build the compute pipeline. The sample includes trained GoogleNet, which was trained with NVIDIA Deep Learning GPU Training System (DIGITS). The training was done with roughly 3000 frames taken from 5-10 feet elevation. Varying levels of detection accuracy are expected based on the video samples fed in. Given that this sample is locked to perform at half-HD resolutions under 10 FPS, video feeds with higher FPS for inference will show stuttering during playback.

This sample does not require a camera.

Building and Running

Prerequisites

- You have followed Steps 1-3 in Building and Running.
- You have installed:
  - CUDA Toolkit
  - OpenCV
- Optionally, you have installed NVIDIA® TensorRT™ (previously known as GPU Inference Engine (GIE))

To build

1. If you want to run the sample without TensorRT, set the following in the Makefile:

```
ENABLETRT := 0
```

   By default, TensorRT is enabled.
2. Enter:

```
$ cd backend
$ make
```

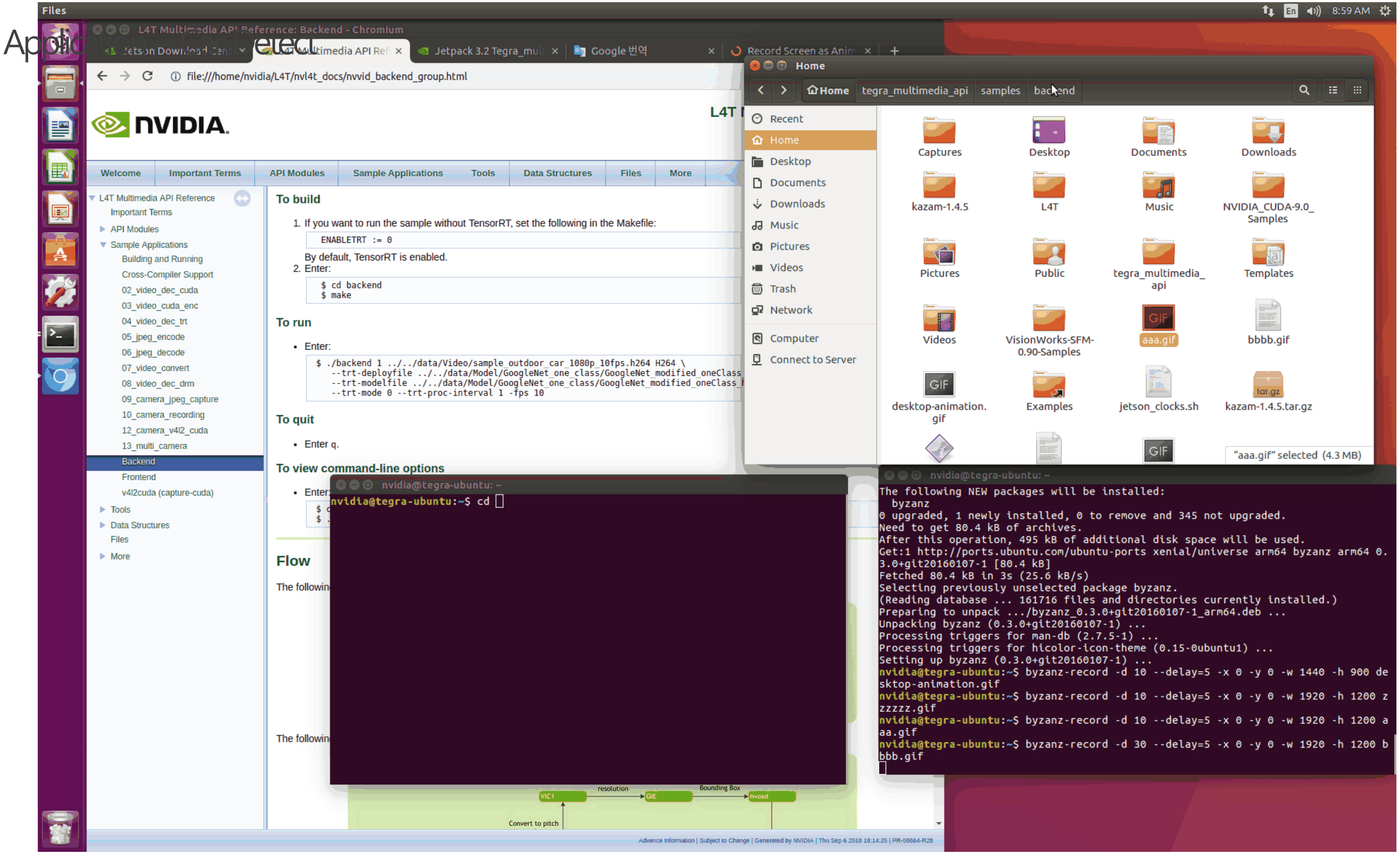
To run

- Enter:

```
$ ./backend 1 ../../data/Video/sample_outdoor_car_1080p_10fps.h264 H264 \
    --trt-deployfile ../../data/Model/GoogleNet_one_class/GoogleNet_modified_oneClass_halfHD.prototxt \
```

01 JetPack Installation

Sample Application – Car Detect



To build

1. If you want to run the sample without TensorRT, set the following in the Makefile:

```
ENABLETRT := 0
```

By default, TensorRT is enabled.

2. Enter:

```
$ cd backend
$ make
```

To run

- Enter:

```
$ ./backend 1 ../../data/Video/sample_outdoor_car_1080p_10fps.h264 H264 \
   --trt-deployfile ../../data/Model/GoogleNet_one_class/GoogleNet_modified_oneClass
   --trt-modelfile ../../data/Model/GoogleNet_one_class/GoogleNet_modified_oneClass_
   --trt-mode 0 --trt-proc-interval 1 -fps 10
```

To quit

- Enter q.

To view command-line options

- Enter:

Flow

The followin

The followin

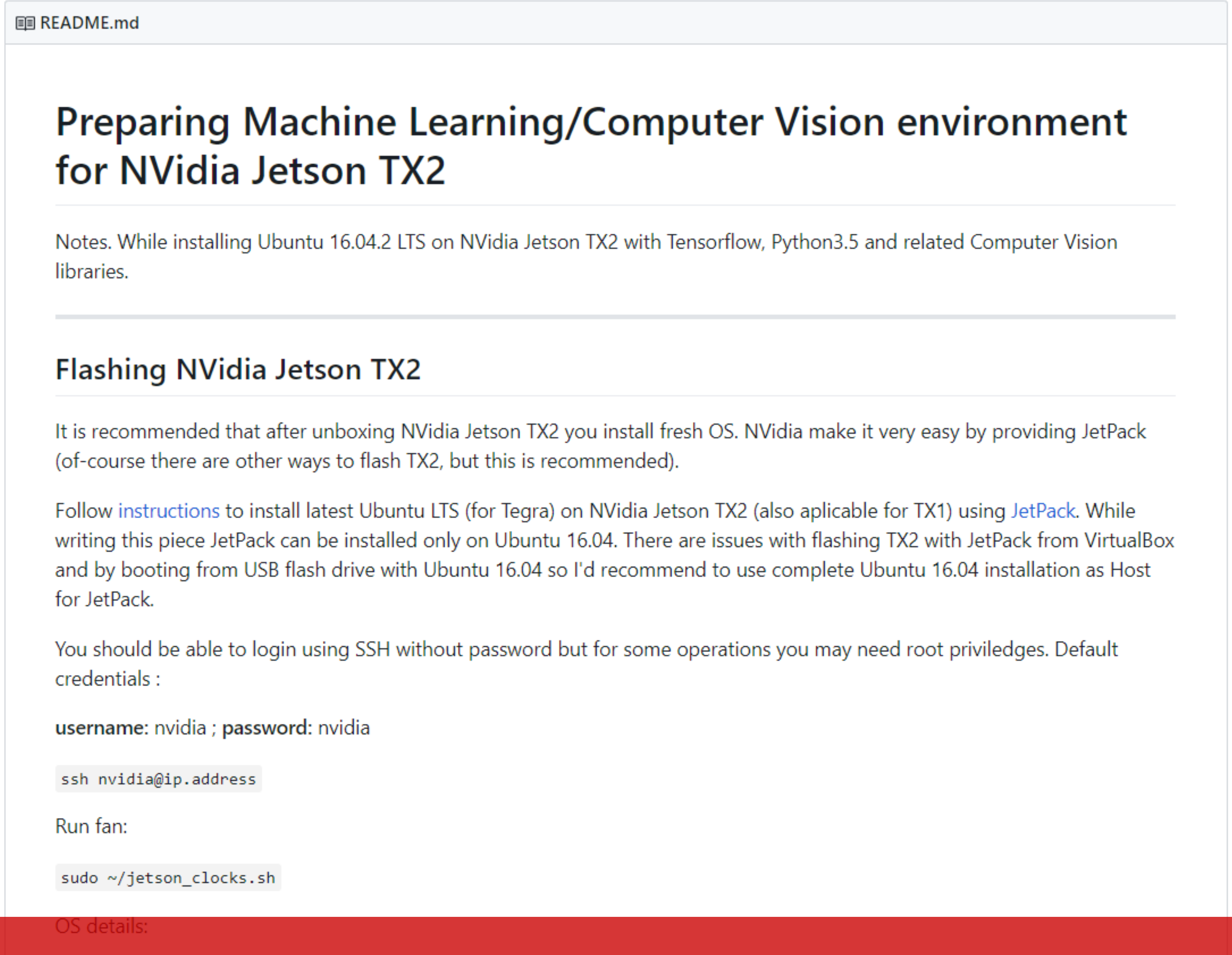
```
nvidia@tegra-ubuntu:~$ cd 
```

```
The following NEW packages will be installed:
  byzanz
0 upgraded, 1 newly installed, 0 to remove and 345 not upgraded.
Need to get 80.4 kB of archives.
After this operation, 495 kB of additional disk space will be used.
Get:1 http://ports.ubuntu.com/ubuntu-ports xenial/universe arm64 byzanz arm64 0.
3.0+git20160107-1 [80.4 kB]
Fetched 80.4 kB in 3s (25.6 kB/s)
Selecting previously unselected package byzanz.
(Reading database ... 161716 files and directories currently installed.)
Preparing to unpack .../byzanz_0.3.0+git20160107-1_arm64.deb ...
Unpacking byzanz (0.3.0+git20160107-1) ...
Processing triggers for man-db (2.7.5-1) ...
Processing triggers for hicolor-icon-theme (0.15-0ubuntu1) ...
Setting up byzanz (0.3.0+git20160107-1) ...
nvidia@tegra-ubuntu:~$ byzanz-record -d 10 --delay=5 -x 0 -y 0 -w 1440 -h 900 de
sktop-animation.gif
nvidia@tegra-ubuntu:~$ byzanz-record -d 10 --delay=5 -x 0 -y 0 -w 1920 -h 1200 z
zzzzz.gif
nvidia@tegra-ubuntu:~$ byzanz-record -d 10 --delay=5 -x 0 -y 0 -w 1920 -h 1200 a
aa.gif
nvidia@tegra-ubuntu:~$ byzanz-record -d 30 --delay=5 -x 0 -y 0 -w 1920 -h 1200 b
bbb.gif
```

02 Scikit-learn Installation

NVidia-Jetson-TX2-Install-Guide-Lines

https://github.com/Naurislv/NVidia-Jetson-TX2-Install-Guide-Lines



README.md

Preparing Machine Learning/Computer Vision environment for NVidia Jetson TX2

Notes. While installing Ubuntu 16.04.2 LTS on NVidia Jetson TX2 with Tensorflow, Python3.5 and related Computer Vision libraries.

Flashing NVidia Jetson TX2

It is recommended that after unboxing NVidia Jetson TX2 you install fresh OS. NVidia make it very easy by providing JetPack (of-course there are other ways to flash TX2, but this is recommended).

Follow instructions to install latest Ubuntu LTS (for Tegra) on NVidia Jetson TX2 (also aplicable for TX1) using JetPack. While writing this piece JetPack can be installed only on Ubuntu 16.04. There are issues with flashing TX2 with JetPack from VirtualBox and by booting from USB flash drive with Ubuntu 16.04 so I'd recommend to use complete Ubuntu 16.04 installation as Host for JetPack.

You should be able to login using SSH without password but for some operations you may need root priviledges. Default credentials :

username: nvidia ; **password**: nvidia

```
ssh nvidia@ip.address
```

Run fan:

```
sudo ~/jetson_clocks.sh
```

OS details:

02 Scikit-learn Installation

Upgrade Ubuntu

```
sudo apt-get update && sudo apt-get upgrade
```

```
sudo apt-get install python3-numpy swig python3-dev python3-pip python3-wheel -y
```

- Upgrade pip : `sudo pip3 install --upgrade pip`
- Other dependecies (replace 3.5 with the python version you are using). Order is important.:

i. `sudo pip3.5 install --upgrade azure.common` (If you will not use MS Azure ignore this)

ii. `sudo pip3.5 install --upgrade azure.servicebus` (If you will not use MS Azure ignore this)

iii. `sudo pip3.5 install --upgrade azure.storage` (If you will not use MS Azure ignore this)

iv. `sudo pip3.5 install --upgrade setuptools ez_setup`

v. `sudo pip3.5 install --upgrade matplotlib`

vi. `sudo pip3.5 install --upgrade configparser`

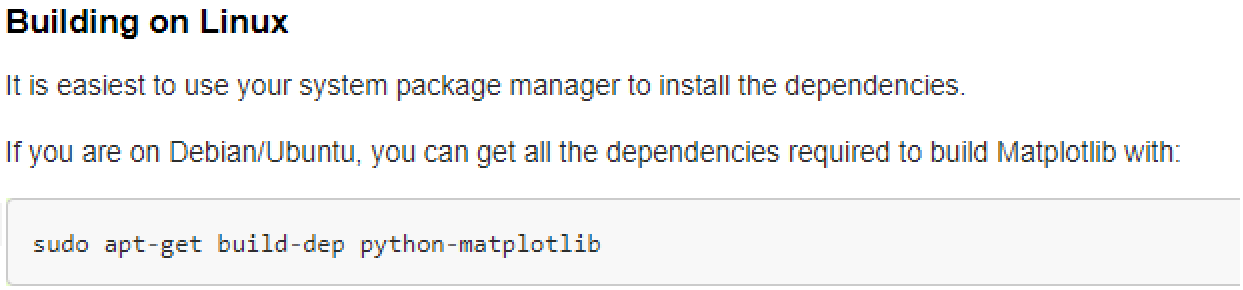
vii. `sudo pip3.5 install --upgrade numpy`

viii. `sudo apt-get install libblas-dev liblapack-dev libatlas-base-dev`

ix. `sudo pip3.5 install --upgrade scipy`

x. `sudo pip3.5 install --upgrade pandas`

xi. `sudo pip3.5 install --upgrade sklearn`



Building on Linux

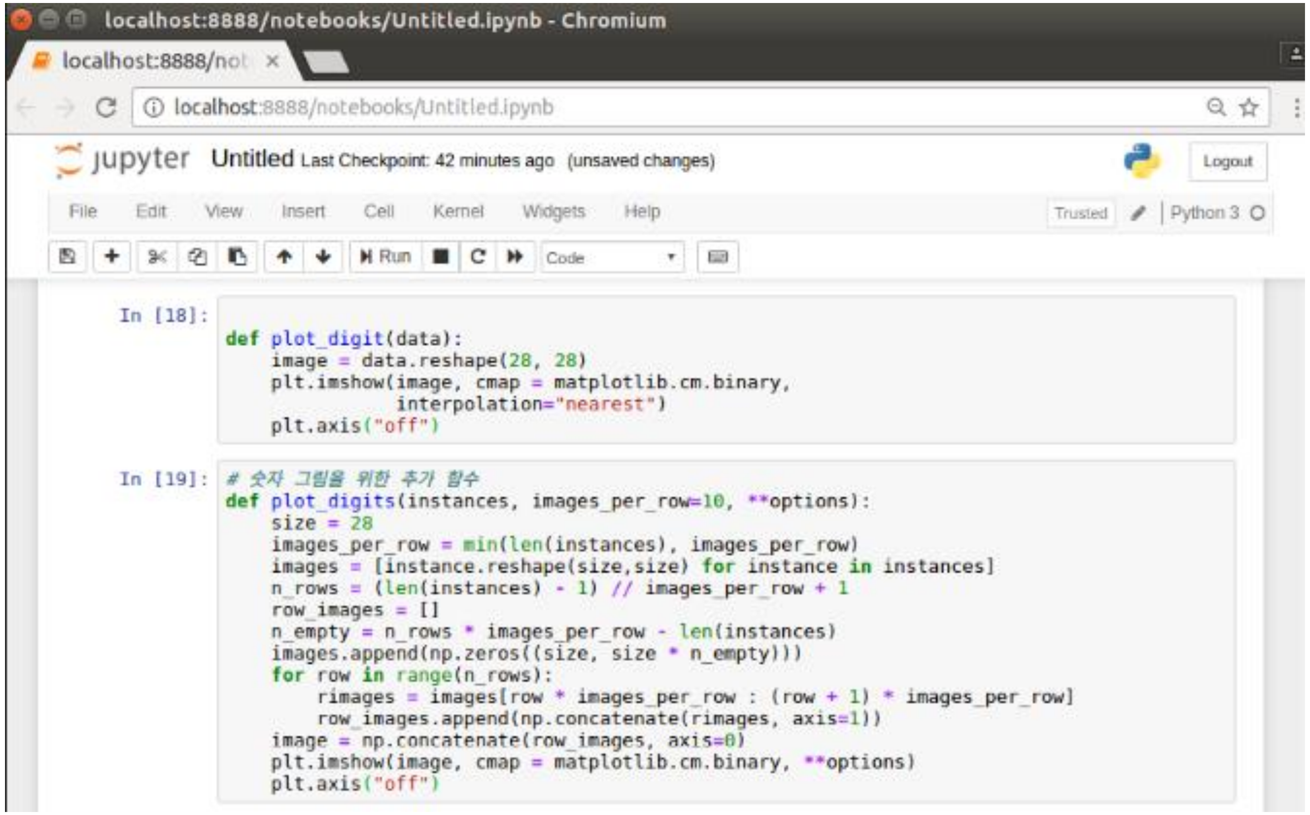
It is easiest to use your system package manager to install the dependencies.

If you are on Debian/Ubuntu, you can get all the dependencies required to build Matplotlib with:

```
sudo apt-get build-dep python-matplotlib
```

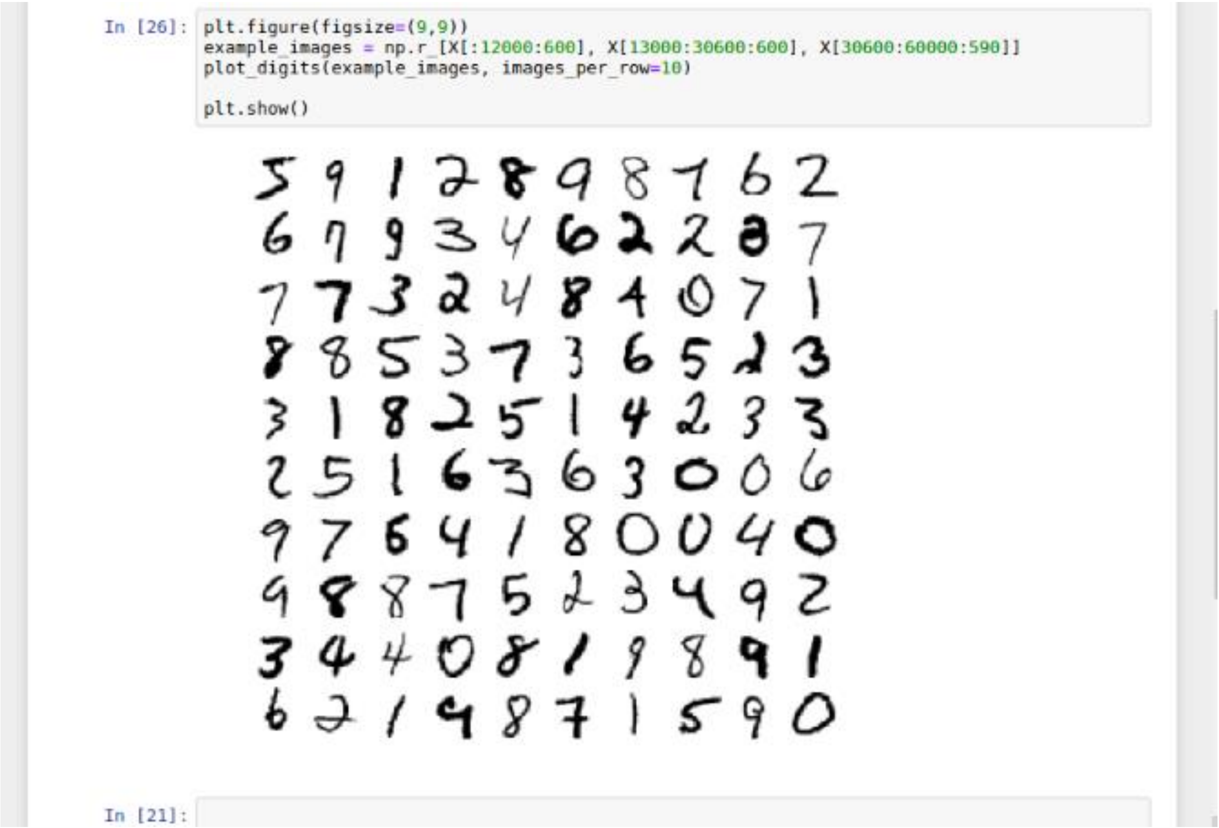
02 Scikit-learn Installation

chap3 example code


localhost:8888/notebooks/Untitled.ipynb - Chromium
localhost:8888/not
localhost:8888/notebooks/Untitled.ipynb
jupyter Untitled Last Checkpoint: 42 minutes ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

```
In [18]:
def plot_digit(data):
    image = data.reshape(28, 28)
    plt.imshow(image, cmap = matplotlib.cm.binary,
               interpolation="nearest")
    plt.axis("off")
```

```
In [19]:
# 숫자 그림을 위한 추가 함수
def plot_digits(instances, images_per_row=10, **options):
    size = 28
    images_per_row = min(len(instances), images_per_row)
    images = [instance.reshape(size,size) for instance in instances]
    n_rows = (len(instances) - 1) // images_per_row + 1
    row_images = []
    n_empty = n_rows * images_per_row - len(instances)
    images.append(np.zeros((size, size * n_empty)))
    for row in range(n_rows):
        rimages = images[row * images_per_row : (row + 1) * images_per_row]
        row_images.append(np.concatenate(rimages, axis=1))
    image = np.concatenate(row_images, axis=0)
    plt.imshow(image, cmap = matplotlib.cm.binary, **options)
    plt.axis("off")
```



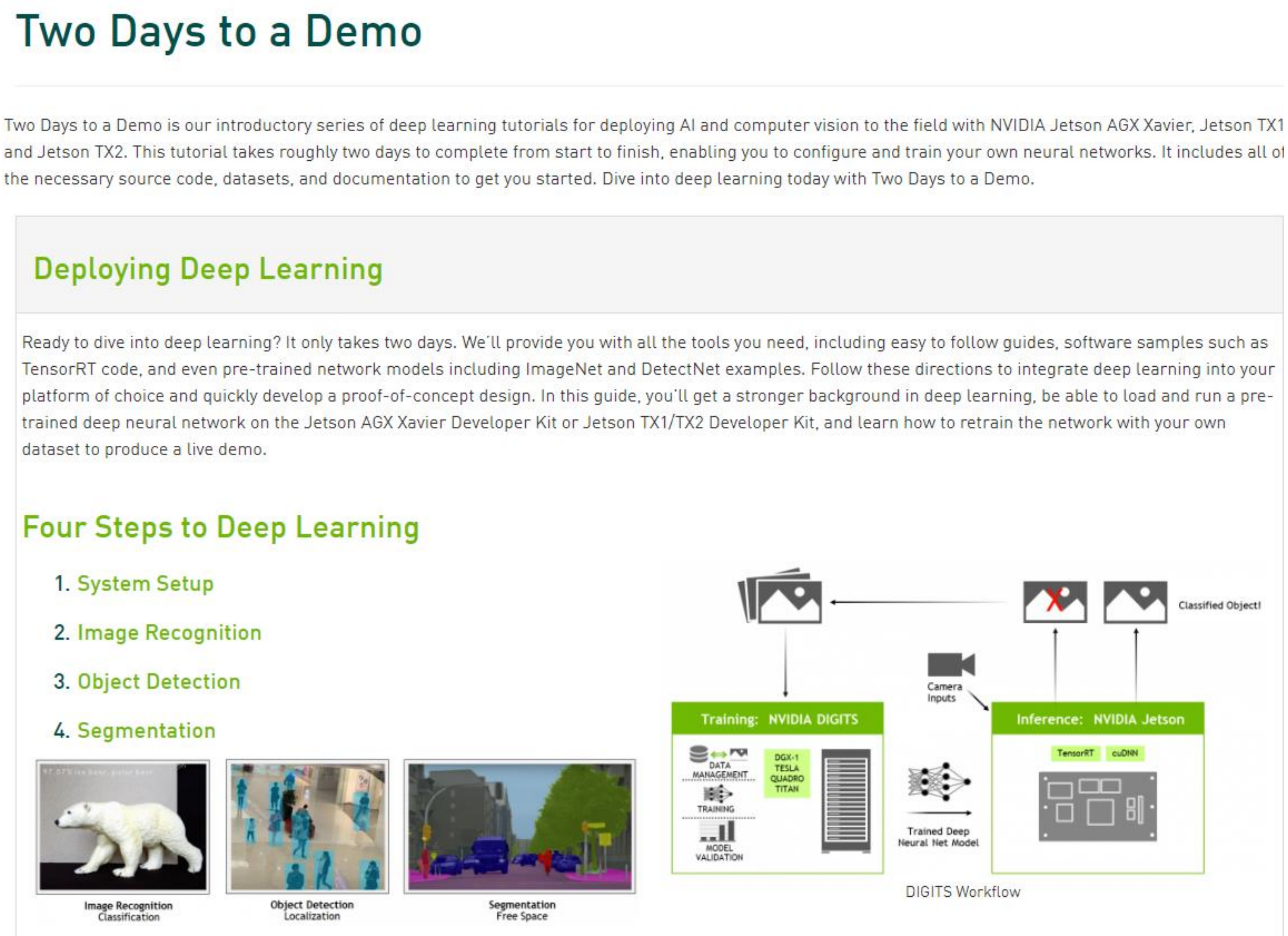
```
In [26]:
plt.figure(figsize=(9,9))
example_images = np.r_[X[:12000:600], X[13000:30600:600], X[30600:60000:590]]
plot_digits(example_images, images_per_row=10)

plt.show()
```

```
In [21]:
```

03 NVIDIA Two Days to a Demo

https://developer.nvidia.com/embedded/twodaystoademo



Two Days to a Demo

Two Days to a Demo is our introductory series of deep learning tutorials for deploying AI and computer vision to the field with NVIDIA Jetson AGX Xavier, Jetson TX1 and Jetson TX2. This tutorial takes roughly two days to complete from start to finish, enabling you to configure and train your own neural networks. It includes all of the necessary source code, datasets, and documentation to get you started. Dive into deep learning today with Two Days to a Demo.

Deploying Deep Learning

Ready to dive into deep learning? It only takes two days. We'll provide you with all the tools you need, including easy to follow guides, software samples such as TensorRT code, and even pre-trained network models including ImageNet and DetectNet examples. Follow these directions to integrate deep learning into your platform of choice and quickly develop a proof-of-concept design. In this guide, you'll get a stronger background in deep learning, be able to load and run a pre-trained deep neural network on the Jetson AGX Xavier Developer Kit or Jetson TX1/TX2 Developer Kit, and learn how to retrain the network with your own dataset to produce a live demo.

Four Steps to Deep Learning

1. System Setup
2. Image Recognition
3. Object Detection
4. Segmentation

Image Recognition
Classification

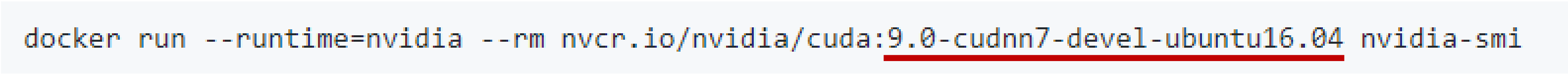
Object Detection
Localization

Segmentation
Free Space

DIGITS Workflow

03 NVIDIA Two Days to a Demo

1. Installing Ubuntu on the Host
2. Running JetPack on the Host
3. Installing the NVIDIA driver (v390.87)
4. Installing Docker



```
docker run --runtime=nvidia --rm nvcr.io/nvidia/cuda:9.0-cudnn7-devel-ubuntu16.04 nvidia-smi
```

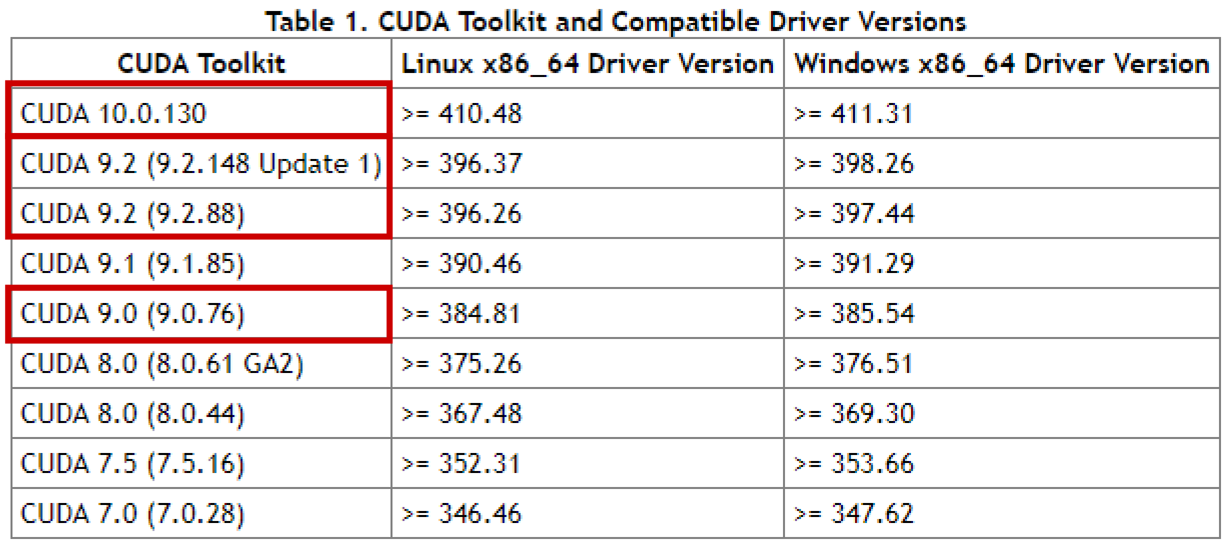


Table 1. CUDA Toolkit and Compatible Driver Versions

CUDA Toolkit	Linux x86_64 Driver Version	Windows x86_64 Driver Version
CUDA 10.0.130	>= 410.48	>= 411.31
CUDA 9.2 (9.2.148 Update 1)	>= 396.37	>= 398.26
CUDA 9.2 (9.2.88)	>= 396.26	>= 397.44
CUDA 9.1 (9.1.85)	>= 390.46	>= 391.29
CUDA 9.0 (9.0.76)	>= 384.81	>= 385.54
CUDA 8.0 (8.0.61 GA2)	>= 375.26	>= 376.51
CUDA 8.0 (8.0.44)	>= 367.48	>= 369.30
CUDA 7.5 (7.5.16)	>= 352.31	>= 353.66
CUDA 7.0 (7.0.28)	>= 346.46	>= 347.62

Thank You!

자세한 내용을 입력해 주세요

DAILY-C

데일리C 레몬워터

소중한 내 몸을 위한 특별한 물

레몬의 상큼함이 듬뿍!

물보다 맛있는 음료 데일리C 레몬워터

비타민C가 함유된 리프레시 비타민 음료

240ml캔, 500ml 페트

DAILY-C

데일리C 레몬1000C+

레몬 과즙이 들어있는 리얼 레몬 드링크

하루 한 병으로 비타민C를 상큼하게 충전하는

리얼 레몬 드링크

레몬 32개 분량에 해당하는 비타민C 1,0000mg

을

가볍게 충전해 보세요

140ml 병

그래서 우리 발표

아이디어를

이렇게 글씨로

포인트 살려서 예쁘게

적어 주세요