UCS671:Embedded Vision

Jhilik Bhattacharya

January 16, 2024

Abstract

In this course you will learn to work on the Jetson Nano kit and the Jetbot. You will learn about Nvidia NGC docker images,how to use them and deploy different CV applications on the kit.

1 Prerequisites

- Mandatory: Dual boot with Ubuntu
- Mandatory: Your system should have atleast 1 USB-A to attach data cable, else get a converter (USB-C to USB-A)
- Preferred: Carry an ethernet cable
- Preferred: Carry a USB-C to USB-A cable (to be attached to your USB-A laptop port, alternative for power plug)

2 Lecture 1,2: Connecting with Jetson and Overview of dockers

In this session you will issue the Jetson Nano board. Some kits are already mounted on the Jetbot whereas some are available individually. We need the Jetson board, a micro-usb data cable and the power cable to connect. Make sure your jetson kits do not touch the table. Once you power on, open a linux terminal on your laptop and connect to the jetson using ssh.
ssh -X csed@192.168.55.1
csed is the user id created on the jetson. IP assigned due to the data cable is 192.168.55.1. The laptop is assigned 192.168.55.100. The $-X$ option is used to transfer graphical transfer rights to the host(your laptop). Once you are logged in, the terminal becomes your jetson terminal. You will be able to see the difference in your terminal prompt. If you connect with ssh without the username , it will work only if your laptop username is same as the jetson username. You can check whether the graphical transfer is working by running *gedit* on your jetson terminal.
Now that you are in the jetson machine you can use linux based commands(ls,cd etc.) to move around on your machine. Your jetson board is currently fully functional with installed OS as well as other docker images. We will in a minute learn more about docker images. Before that, you can take a quick look at the steps for installing OS on a new jetson board in case you have to start from scratch. You can refer the Nvidia developer page for instructions to burn the OS image on the micro SD card using any software like etcher and starting your jetson device. The first time you need to connect USB mouse,keyboard and a monitor (using the HDMI port) to the Jetson. Once the GUI starts, follow the steps to create user account and get get the board ready. To maintain consistency, use username as ”csed” and password as ”cslab768”.
Coming back to the concept of dockers, they are an executable package to run your application. They include all software dependencies within it and can be run on any system. A docker builds images and runs the instance of the image (called container). The nvidia NGC catalog hosts a number of docker images which can be pulled using the following command
sudo docker pull xxxx

$xxxx$ denotes the name of the image, for example $nvcr.io/nvidia/l4t - ml : r36.2.0 - py3$. In the $x : y$ format, x represents the repository and y gives the tag name. The tag name denotes the jetpack version number as well as the software version. The jetpack version and associated tag numbers are shown below:

- JetPack 5.1.1 (L4T R35.3.1)
- JetPack 5.1 (L4T R35.2.1)
- JetPack 5.0.2 (L4T R35.1.0)
- JetPack 5.0.1 Developer Preview (L4T R34.1.1)
- JetPack 5.0.0 Developer Preview (L4T R34.1.0)
- JetPack 4.6.1 (L4T R32.7.1)
- JetPack 4.6 (L4T R32.6.1)
- JetPack 4.5 (L4T R32.5.0)
- JetPack 4.4.1 (L4T R32.4.4)
- JetPack 4.4 (L4T R32.4.3)
- JetPack 4.4 Developer Preview (L4T R32.4.2)

Your jetpack version is 4.6. Recall this while downloading any docker image. At this point if a question pops in your mind- "Whats the advantage of using docker images over conda environments? ". Doesnt a conda environment do a similar thing? You can create an environment, activate it and have a list of packages(softwares) installed there. This does not clash with software versions in any other environment. If a docker image also offers us an environment with all softwares installed, whats the advantage of dockers over conda environment. Well the simplest answer is that docker image allows you a OS level virtualization. You can run a linux container from windows OS if you are using dockers.
You can view the list of docker images installed on your jetson by executing
sudo docker images
For checking the list of active containers you can use
sudo docker ps
For saving any container as an image you can commit the container using
sudo docker commit container_id repository:tagname
You run a docker image using
sudo docker run xxxx
where *xxxx* is the repository:tag for a docker image. Take a note of different options like *it*, *rm*, *network*, *runtime*, *volume*, *device* you can use
Now that you know how to run a container, you should be able to answer the following questions

- You have connected a device to your jetson nano but cannot access it from inside a container. Why?
- How to prevent loosing all your data saved while running the container?
- When is interactive mode (-it) not required?
- When is it important to not use the *rm* option?

Finally you can delete images and containers using
sudo docker rmi image_id
sudo docker rm container_id

Alternatively you may also use repository:tag for images or names for containers.

3 Lab 1: Getting Acquainted with dockers

In this session you will try the following

- Run the docker image nvcr.io/nvidia/l4t-ml:r36.2.0-py3 and check which version of pytorch, tensorflow, opencv, numpy are installed here
- Run the docker with rm option, install a software (say matplotlib or pillow) and exit. Run it again and check if you are able to access the softwares
- Repeat the experiment without using rm option.
- Save the container as a new image
- Delete the new image you just created
- Check whether the container you saved is still available and save an image again
- This time delete the container you are using and check if you can run the image you have saved.