ITAI 2374-01 Experience
James H Phelan
2022.01.16 - present

Link to course:
https://courses.nvidia.com/courses/course-v1:DLI+S-RX-02+V2/about
Link to syllabus:
https://developer.nvidia.com/edge-ai-robotics-teaching-kit-syllabus
Set up the Nano:
https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit#intro

First boot went on forever and seemed to get stuck.
Redownloaded the .zip & reflashed the drive.

2022.01.17
Second boot prompt.
Accept EULA
Setup configuration completed
log in as ubuntu@NANO
password “ubuntu”
having username other than “ubuntu” makes SSH difficult as it wants to default to ubuntu
require login password
accept defaults
$ **ip a**
to display IP addresses
Set up PuTTY for login

look at L4T-README/README-vnc.txt
[I’ve had a LOT of trouble getting VNC working on Ubuntu]
ubuntu@NANO:~$ **sudo apt update**
ubuntu@NANO:~$ **sudo apt install vino**
ubuntu@NANO:~$ **mkdir -p ~/.config/autostart**
ubuntu@NANO:~$ **cp /usr/share/applications/vino-server.desktop ~/.config/autostart**
ubuntu@NANO:~$ **gsettings set org.gnome.Vino prompt-enabled false**
ubuntu@NANO:~$ **gsettings set org.gnome.Vino require-encryption false**
ubuntu@NANO:~$ **gsettings set org.gnome.Vino authentication-methods "['vnc']"**
ubuntu@NANO:~$ **gsettings set org.gnome.Vino vnc-password $(echo -n 'ubuntu'|base64)**
ubuntu@NANO:~$ **sudo reboot**

launch VNCviewer
was able to log in ok w/ acknowledgement of cautions.
Success!

2022.01.18
The recommended Logitec C270 HD WEBCAM camera arrived today:
https://www.amazon.com/Logitech-Widescreen-designed-Calling-Recording/dp/B004FHO5Y6/ref=sr_1_2?gclid=Cj0KCQjwi43oBRDBARIsAExSRQHtMypbRxcsZLLG2O91x0rEosiP8DcfcNzNWsjdQlfzSMCaU3J7_bUaAqEiEALw_wcB&hvadid=321859566873&hvdev=c&hvlocphy=9032151&hvnetw=g&hvpos=1t1&hvqmt=b&hvrand=4551797936993747797&hvtargid=aud-676677759524%3Akwd-297700966697&hydadcr=18004_9429434&keywords=c270+hd+webcam&qid=1560580432&refinements=p_85%3A2470955011&rnid=2470954011&rps=1&s=gateway&sr=8-2

tried to run with cheese:
$ **cheese**
Only black black display with inactive buttons

There is help here, but will do the setup in order of the course to avoid confusion:
https://forums.developer.nvidia.com/t/jetson-nano-faq/82953#5390621

Back to the course track:

https://courses.nvidia.com/courses/course-v1:DLI+S-RX-02+V2/courseware/b2e02e999d9247eb8e33e893ca052206/63a4dee75f2e4624afbc33bce7811a9b/?child=first

Initial Setup Headless Mode
To complete setup when no display is attached to the developer kit, you'll need to connect the developer kit to another computer and then communicate with it via a terminal application (e.g., PuTTY) to handle the USB serial communication on that other computer.

Not much help here. Got additional help from: https://everythingwhat.com/how-do-i-use-putty-with-usb-to-serial-adapter

In short, use system manager to find USB-serial device. In my case COM3. Open PuTTY, start new connection, select "serial" and set port to COM3 {or yours}, connect. Voila! Termial opens to NANO login. I'll still use over ethernet.

Additional Notes About Setup
The course runs best if there is a "swap" file of size 4GB, so that if the Jetson Nano is a little short of RAM it can extend a bit by swapping with some of the (slower) disk space. After setting up the microSD card and booting the system, check your memory and swap values with this command, which shows the values in megabytes. You should see **4071** as the size of the swap if you have 4GB configured:

```
ubuntu@NANO:~$ free -m
              total        used        free      shared  buff/cache
available
Mem:           3956        1582        1307          40        1065
2168
Swap:          1978           0        1978
```

If you don't have the right amount of swap, or want to change the value, use the following procedure to do so (from a terminal):

```
# Disable ZRAM:
sudo systemctl disable nvzramconfig
ubuntu@NANO:~$ sudo systemctl disable nvzramconfig
Removed /etc/systemd/system/multi-user.target.wants/nvzramconfig.service.

# Create 4GB swap file
sudo fallocate -l 4G /mnt/4GB.swap
sudo chmod 600 /mnt/4GB.swap
sudo mkswap /mnt/4GB.swap
ubuntu@NANO:~$ sudo fallocate -l 4G /mnt/4GB.swap
ubuntu@NANO:~$ sudo chmod 600 /mnt/4GB.swap
ubuntu@NANO:~$ sudo mkswap /mnt/4GB.swap
Setting up swapspace version 1, size = 4 GiB (4294963200 bytes)
no label, UUID=8eb4eb58-0b96-496c-8e76-7ee430410311
```

```
# Append the following line to /etc/fstab
sudo su
echo "/mnt/4GB.swap swap swap defaults 0 0" >> /etc/fstab
exit

ubuntu@NANO:~$ sudo su
root@NANO:/home/ubuntu# echo "/mnt/4GB.swap swap swap defaults 0
0" >> /etc/fstab
root@NANO:/home/ubuntu# exit
ubuntu@NANO:~$

# REBOOT!
ubuntu@NANO:~$ sudo reboot
```

Enough for tonight. Send email to classmates with advice re setup.

[2022.01.22 late footnote]
Notice when you reboot the Nano when connected via USB that a new drive appears on the laptop: L4T-README with the contents of the L4T-README file from the Nano desktop.

2022.01.20
Ok, looks like this is the next step to load the Jupyter notebooks and run the camera:
https://courses.nvidia.com/courses/course-v1:DLI+S-RX-02+V2/courseware/b2e02e999d9247eb8e33e893ca052206/63a4dee75f2e4624afbc33bce7811a9b/?child=first
Headless Device Mode
Download Docker And Start JupyterLab
... already set up
6 Add a data directory for the course with the following command in the Jetson Nano terminal you've logged into:
ubuntu@NANO:~$ **mkdir -p ~/nvdli-data**
7 Run the Docker container with the following command, where <tag> is a combination of the course version and Jetson Nano JetPack L4T operating system version (form is <tag> = <course_version>-<L4T_version>). A list of tags can be found in the course NVIDIA NGC cloud page :
https://catalog.ngc.nvidia.com/orgs/nvidia/teams/dli/containers/dli-nano-ai

Which says: "**Latest Tag v2.0.2-r32.6.1kr**"
Modified December 18, 2021

Clicking the "Pull tag" button cc this to the clipboard:
docker pull nvcr.io/nvidia/dli/dli-nano-ai:v2.0.2-r32.6.1kr

```
sudo docker run --runtime nvidia -it --rm --network host \
    --volume ~/nvdli-data:/nvdli-nano/data \
    --device /dev/video0 \
    nvcr.io/nvidia/dli/dli-nano-ai:<tag>
```
Inserting <tag> as **v2.0.2-r32.6.1kr** makes it

ubuntu@NANO:~$ **sudo docker run --runtime nvidia -it --rm --network host **
**   --volume ~/nvdli-data:/nvdli-nano/data **
**   --device /dev/video0 **
**   nvcr.io/nvidia/dli/dli-nano-ai:v2.0.2-r32.6.1kr**
long list (41) of
12 random hex characters:
     Waiting >>
     Downloading *[byte counter]* >>
     Download complete >>
     Extracting *[byte counter]*
     Pull complete
many minutes to complete. We shall see....

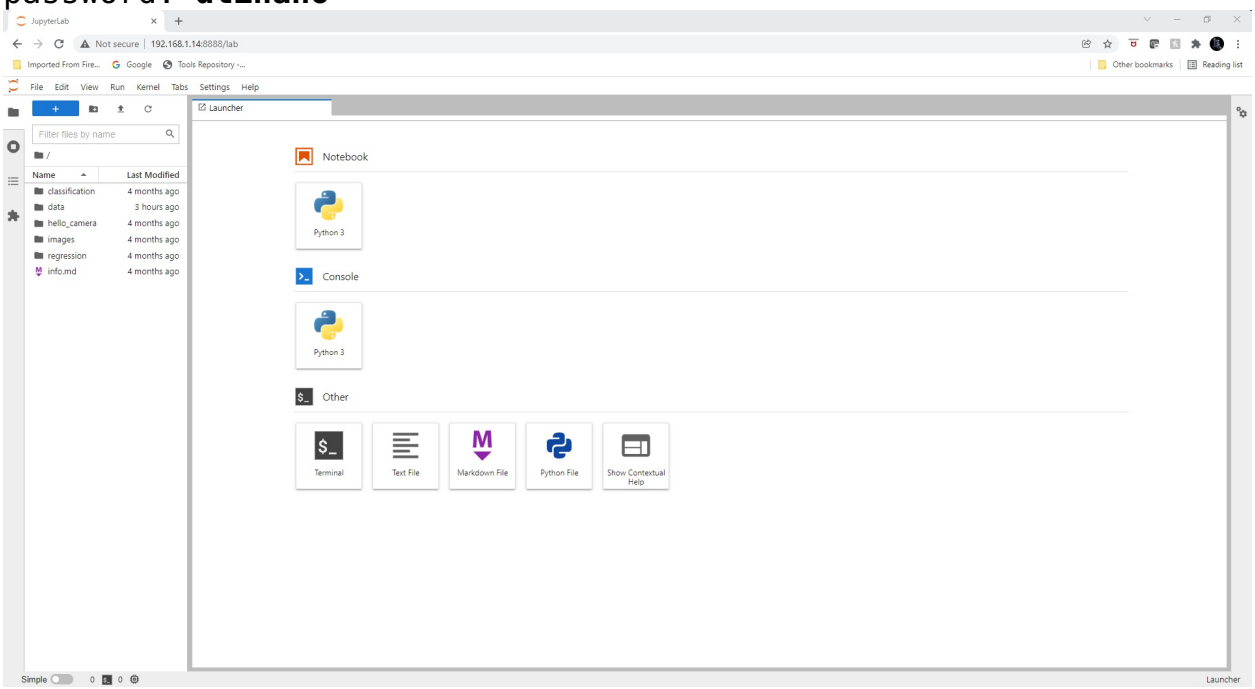
To create and run a reusable script for this step try the following (example tag shown):
Why would I want to do that?

If using the alternate CSI camera instead of the USB webcam, add --volume /tmp/argus_socket:/tmp/argus_socket to your docker run command. For example:
This could come in handy if I want to use the Picamera later.

Logging Into The JupyterLab Server
Open the following link address : 192.168.55.1:8888
I question this IP address!
The JupyterLab server running on the Jetson Nano will open up with a login prompt the first time.
Enter the password: **dlinano**
You will see this screen. Congratulations!

Status: Downloaded newer image for nvcr.io/nvidia/dli/dli-nano-ai:v2.0.2-r32.6.1kr
allow 10 sec for JupyterLab to start @ **http://192.168.1.14:8888** (password dlinano)
JupterLab logging location: /var/log/jupyter.log (inside the container)

Open Chrome web browser: http://192.168.1.14:8888
password: **dlinano**



Enough for today

2022.01.21
To create and run a reusable script for this step try the following (example tag shown):

```
# create a reusable script
echo "sudo docker run --runtime nvidia -it --rm --network host \
    --volume ~/nvdli-data:/nvdli-nano/data \
    --device /dev/video0 \
    nvcr.io/nvidia/dli/dli-nano-ai:v2.0.1-r32.6.1" > docker_dli_run.sh
# make the script executable
chmod +x docker_dli_run.sh
# run the script
./docker_dli_run.sh
```

If using the alternate CSI camera instead of the USB webcam, add --volume /tmp/argus_socket:/tmp/argus_socket to your docker run command. For example:

```
# create a reusable script
echo "sudo docker run --runtime nvidia -it --rm --network host \
    --volume ~/nvdli-data:/nvdli-nano/data \
    --volume /tmp/argus_socket:/tmp/argus_socket \
    --device /dev/video0 \
    nvcr.io/nvidia/dli/dli-nano-ai:v2.0.1-r32.6.1" > docker_dli_run.sh
# make the script executable
chmod +x docker_dli_run.sh
# run the script
./docker_dli_run.sh
```

Similar list as above. Top 3/4 “already exist”, rest downloaded & extracted as above.
Now Jupyter notebook won’t launch at
192.168.55.1:8888 nor
192.168.1.14:8888

Try reloading the original script
ubuntu@NANO:~$ **sudo docker run --runtime nvidia -it --rm --network host **
**--volume ~/nvdli-data:/nvdli-nano/data **
**--device /dev/video0 **
nvcr.io/nvidia/dli/dli-nano-ai:v2.0.2-r32.6.1kr
No download but an immediate, then
Status: Downloaded newer image for nvcr.io/nvidia/dli/dli-nano-ai:v2.0.2-r32.6.1kr
allow 10 sec for JupyterLab to start @ **http://192.168.1.14:8888** (password dlinano)
JupterLab logging location: /var/log/jupyter.log (inside the container)
Jupyter notebook launches & logs in as described.

usb_camera.ipynb runs ok
csi_camera.ipynb runtime errors, possibly because I realoaded the original script w/o the Picamera code.

Try the bash script
ubuntu@NANO:~$ **./docker_dli_run.sh**
Note it comes out in root:
root@NANO:/nvdli-nano#

Still no-go on csi-camera.
Reboot
exit
$ **sudo reboot**

I note the Jupyter notebook won't open unless I relaunch the docker script. So that's why they offer a reusable script! Mine should have the commands for the Picamera:

sudo docker run --runtime nvidia -it --rm --network host
--volume ~/nvdli-data:/nvdli-nano/data
--volume/tmp/argus_socket:/tmp/argus_socket
--device /dev/video0
nvcr.io/nvidia/dli/dli-nano-ai:v2.0.1-r32.6.1

Still no-go. Diane reports that the Nano A0 can only handle ONE camera, so never mind. The B1 is preferred but my A0 will "barely" work for the class.

No adeq response from HCA re registration.
Class will be virtual first 2 weeks. COVID
Cannot access unless registered.
D' not sure if campus will be open 6pm Monday.
Will let us know.

2022.01.24
Managed to slog through the horrible hccs.edu website. Determined from my ToDo list that I needed to pass some readines test. I got into line for the virtual advisor which went surprisingly well. Advisor informed I needed prior transcript (from 40 yrs ago??) to defer the test. I requested and got email for supervisor admin and sent her a photo of my Baylor College of Medicine diploma. She ok'd it and even forwarded me forms for a Senior discount which I $550 credit for the 3 hrs course! Got email link from Raymond and was able to join the class. There were over a dozen students of all kinds and interests when we introduced ourselves. R' gave slideshow intro to NVIDIA course, syllabus and schedule. Equipment to be provided when avail from

NVIDIA. Was placed in 1/3 lab groups for project who formed a GroupMe.

2022.01.25
Texted and called info to Roberto who will attempt to join class tho registration “closed” yesterday.

Want to try to get Picamera to run on Nano by itself without the USB camera:

ubuntu@NANO:~$ **ls /dev/video***
ls: cannot access '/dev/video*': No such file or directory

https://forums.developer.nvidia.com/t/trying-to-use-the-noir-raspberry-pi-camera-with-my-jetson-nano/175105/2

ubuntu@NANO:~$ **gst-launch-1.0 nvarguscamerasrc ! nvoverlaysink**
```
Setting pipeline to PAUSED ...
Pipeline is live and does not need PREROLL ...
Setting pipeline to PLAYING ...
New clock: GstSystemClock
Error generated.
/dvs/git/dirty/git-master_linux/multimedia/nvgstreamer/gst-nvarguscamera/gstnvarguscamerasrc.cpp, execute:725 No cameras available
Got EOS from element "pipeline0".
Execution ended after 0:00:00.001111698
Setting pipeline to PAUSED ...
Setting pipeline to READY ...
Setting pipeline to NULL ...
Freeing pipeline ...
```

https://www.okdo.com/getting-started/get-started-with-jetson-nano-2gb-and-csi-camera/
Clone the CSI-Camera repository
ubuntu@NANO:~$ **git clone**
https://github.com/JetsonHacksNano/CSI-Camera.git
Change directory to CSI-Camera.
ubuntu@NANO:~$ **cd CSI-Camera**
ubuntu@NANO:~/CSI-Camera$ **ls**
```
dual_camera.py  LICENSE    simple_camera.cpp
face_detect.py  README.md  simple_camera.py
```

Test the camera, this is a single GStreamer pipeline command.
Ctrl + C to exit.
ubuntu@NANO:~/CSI-Camera$ **gst-launch-1.0 nvarguscamerasrc sensor_id=0 ! **
2' ! \deo/x-raw(memory:NVMM),width=3280, height=2464, framerate=21/1, format=NV12
**> nvvidconv flip-method=2 ! 'video/x-raw, width=816, height=616' ! **
> nvvidconv ! nvegltransform ! nveglglessink -e
Setting pipeline to PAUSED ...

Using winsys: x11
Pipeline is live and does not need PREROLL ...
Got context from element 'eglglessink0': gst.egl.EGLDisplay=context, display=(GstEGLDisplay)NULL;
Setting pipeline to PLAYING ...
New clock: GstSystemClock
Error generated.
/dvs/git/dirty/git-master_linux/multimedia/nvgstreamer/gst-nvarguscamera/gstnvarguscamerasrc.cpp, execute:725 **No cameras available**
Got EOS from element "pipeline0".
Execution ended after 0:00:00.020712393
Setting pipeline to PAUSED ...
Setting pipeline to READY ...
Setting pipeline to NULL ...
Freeing pipeline ...

Wonder if camera has failed? Try another?
ubuntu@NANO:~/CSI-Camera$ **sudo shutdown -h now**
Replaced w/ another Picamera.
REBOOT
ubuntu@NANO:~$ **ls /dev/video***
/dev/video0
So, now at least there’s a camera.
$ **cheese**
opens but “No device found”

ubuntu@NANO:~$ **gst-launch-1.0 nvarguscamerasrc ! nvoverlaysink**

```
Setting pipeline to PAUSED ...
Pipeline is live and does not need PREROLL ...
Setting pipeline to PLAYING ...
New clock: GstSystemClock
GST_ARGUS: Creating output stream
CONSUMER: Waiting until producer is connected...
GST_ARGUS: Available Sensor modes :
GST_ARGUS: 3264 x 2464 FR = 21.000000 fps Duration = 47619048 ; Analog Gain range min 1.000000,
max 10.625000; Exposure Range min 13000, max 683709000;

GST_ARGUS: 3264 x 1848 FR = 28.000001 fps Duration = 35714284 ; Analog Gain range min 1.000000,
max 10.625000; Exposure Range min 13000, max 683709000;

GST_ARGUS: 1920 x 1080 FR = 29.999999 fps Duration = 33333334 ; Analog Gain range min 1.000000,
max 10.625000; Exposure Range min 13000, max 683709000;

GST_ARGUS: 1640 x 1232 FR = 29.999999 fps Duration = 33333334 ; Analog Gain range min 1.000000,
max 10.625000; Exposure Range min 13000, max 683709000;

GST_ARGUS: 1280 x 720 FR = 59.999999 fps Duration = 16666667 ; Analog Gain range min 1.000000,
max 10.625000; Exposure Range min 13000, max 683709000;

GST_ARGUS: 1280 x 720 FR = 120.000005 fps Duration = 8333333 ; Analog Gain range min 1.000000,
max 10.625000; Exposure Range min 13000, max 683709000;

GST_ARGUS: Running with following settings:
   Camera index = 0
   Camera mode  = 2
   Output Stream W = 1920 H = 1080
   seconds to Run    = 0
   Frame Rate = 29.999999
GST_ARGUS: Setup Complete, Starting captures for 0 seconds
GST_ARGUS: Starting repeat capture requests.
CONSUMER: Producer has connected; continuing.
```

Video appears on monitor attached to Nano even tho command run from PC over serial terminal!
Now let’s try launching Jupyter notebook and trying again.
ubuntu@NANO:~$ **./docker_dli_run.sh**

Now try this:

```
ubuntu@NANO:~$ gst-launch-1.0 nvarguscamerasrc sensor_id=0 ! \
2' ! \deo/x-raw(memory:NVMM),width=3280, height=2464,
framerate=21/1, format=NV12
>  nvvidconv flip-method=2 ! 'video/x-raw, width=816, height=616'
! \
>  nvvidconv ! nvegltransform ! nveglglessink -e
Setting pipeline to PAUSED ...

Using winsys: x11
Pipeline is live and does not need PREROLL ...
Got context from element 'eglglessink0': gst.egl.EGLDisplay=context, display=(GstEGLDisplay)NULL;
Setting pipeline to PLAYING ...
New clock: GstSystemClock
GST_ARGUS: Creating output stream
CONSUMER: Waiting until producer is connected...
GST_ARGUS: Available Sensor modes :
GST_ARGUS: 3264 x 2464 FR = 21.000000 fps Duration = 47619048 ; Analog Gain range min 1.000000,
max 10.625000; Exposure Range min 13000, max 683709000;

GST_ARGUS: 3264 x 1848 FR = 28.000001 fps Duration = 35714284 ; Analog Gain range min 1.000000,
max 10.625000; Exposure Range min 13000, max 683709000;

GST_ARGUS: 1920 x 1080 FR = 29.999999 fps Duration = 33333334 ; Analog Gain range min 1.000000,
max 10.625000; Exposure Range min 13000, max 683709000;

GST_ARGUS: 1640 x 1232 FR = 29.999999 fps Duration = 33333334 ; Analog Gain range min 1.000000,
max 10.625000; Exposure Range min 13000, max 683709000;

GST_ARGUS: 1280 x 720 FR = 59.999999 fps Duration = 16666667 ; Analog Gain range min 1.000000,
max 10.625000; Exposure Range min 13000, max 683709000;

GST_ARGUS: 1280 x 720 FR = 120.000005 fps Duration = 8333333 ; Analog Gain range min 1.000000,
max 10.625000; Exposure Range min 13000, max 683709000;

GST_ARGUS: Running with following settings:
   Camera index = 0
   Camera mode  = 0
   Output Stream W = 3264 H = 2464
   seconds to Run    = 0
   Frame Rate = 21.000000
GST_ARGUS: Setup Complete, Starting captures for 0 seconds
GST_ARGUS: Starting repeat capture requests.
CONSUMER: Producer has connected; continuing.
ERROR: from element /GstPipeline:pipeline0/GstNvArgusCameraSrc:nvarguscamerasrc0: Internal data
stream error.
Additional debug info:
gstbasesrc.c(3055): gst_base_src_loop ():
/GstPipeline:pipeline0/GstNvArgusCameraSrc:nvarguscamerasrc0:
streaming stopped, reason error (-5)
EOS on shutdown enabled -- waiting for EOS after Error
Waiting for EOS...
^C
```

ubuntu@NANO:~$ **cheese**
(cheese:11524) Gtk-WARNING **: 20:43:39.611: These parsing error: cheese.css:7:35: The style property GtkScrollbar:min-slider-length is deprecated and shouldn't be used anymore. It will be removed in a future version Opening in BLOCKING MODE

https://forums.developer.nvidia.com/t/jetpack-4-5-usb-camera-does-not-work-in-cheese-and-opencv-4-1/166908

Not helpful. It talks about a JetPack 4.5 patch.
Not sure if I have JetPack & reluctant to add random patches to course computer. Will try update.

ubuntu@NANO:~$ **sudo apt update**
ubuntu@NANO:~$ **sudo apt upgrade**

ubuntu@NANO:~$ **sudo apt-cache show nvidia-jetpack**
```
Package: nvidia-jetpack
Version: 4.6-b199
Architecture: arm64
Maintainer: NVIDIA Corporation
Installed-Size: 194
Depends: nvidia-cuda (= 4.6-b199), nvidia-opencv (= 4.6-b199), nvidia-cudnn8 (= 4.6-b199),
nvidia-tensorrt (= 4.6-b199), nvidia-visionworks (= 4.6-b199), nvidia-container (= 4.6-b199),
nvidia-vpi (= 4.6-b199), nvidia-l4t-jetson-multimedia-api (>> 32.6-0),
nvidia-l4t-jetson-multimedia-api (<< 32.7-0)
Homepage: http://developer.nvidia.com/jetson
Priority: standard
Section: metapackages
Filename: pool/main/n/nvidia-jetpack/nvidia-jetpack_4.6-b199_arm64.deb
Size: 29368
SHA256: 69df11e22e2c8406fe281fe6fc27c7d40a13ed668e508a592a6785d40ea71669
SHA1: 5c678b8762acc54f85b4334f92d9bb084858907a
MD5sum: 1b96cd72f2a434e887f98912061d8cfb
Description: NVIDIA Jetpack Meta Package
Description-md5: ad1462289bdbc54909ae109d1d32c0a8
```

2022.01.29
ubuntu@NANO:~$ **ls /dev/video***
/dev/video0 /dev/video1

root@NANO:/home/ubuntu# ls /dev/video*
/dev/video0 /dev/video1

But when I run the python script in the Jupyter notebook for the class it can only find video0 in root!?

!ls -ltrh /dev/video*
```
crw-rw---- 1 root video 81, 0 Jan 29 16:51 /dev/video0
```

Per Diane, the Nano A2 [which I have] can handle only ONE video stream due to some video buffer limitation. The B1 can handle 2 video streams and has 2 CSI ports. This course doesn't require stereo cameras, so I'm ok at least that far. The A2 may have other limitations, however. So unplugging the Picamera should solve the problem. It did.

I didn't like the 224x224 picture window, so changed it to 640x480:
```
from jetcam.usb_camera import USBCamera
#TODO change capture_device if incorrect for your system
camera = USBCamera(width=640, height=480, capture_width=640,
capture_height=480, capture_device=0)
```

P01 Puzzle #1 Due 2022.02.01;23:59
"What is CUDA? Why is it needed?"
According to https://en.m.wikipedia.org/wiki/CUDA
CUDA stands for "Compute Unified Device Architecture" but nobody cares or uses this anymore, including NVIDIA its inventor, because that's pretty nebulous. It's a "parallel computing platform and application programming interface (API) that allows software to use certain types of graphics processing unit (GPU) for general purpose processing". So CUDA is a software layer that lets you use your computer's powerful, often multi-core, Graphics Processing Unit for something other than graphics. Tasks that perform the same computation over and over again, just changing the data, such as matrix calculations used in neural networks for image recognition and classification or simulations. Naturally it interfaces with the common programming languages and multi-processor frameworks. Without it, AI would not be possible on general purpose computers such as PCs and single board systems such as the Jeston Nano or Raspberry Pi.

Team meeting tonight via Microsoft Teams.
“Robotic_Insight_2374"

A01 Assignment #1
Describe the arrangements you made for Team communication. Did all the assigned people participate? State the key ability of the team, and how you expect members to contribute.

We have *too many* means of communication:
phone
text
personal email
HCC email
HCC ViewMyClasses
Canvas
GroupMe
Google Drive
Microsoft Teams
Cisco Webex

Fortunately we’ve settled on Canvas for class announcements, HCC e-mail for general team announcements, Google Drive to share documents, GroupMe for quick questions and Microsoft Teams for live discussions.

We’ve begun to delegate (verb) tasks by show of interest and area of expertise with the delegate (noun) assimilating and submitting the input of the whole team. All are contributing and participating as able.

We’ve only had one meeting and are just getting to know each other. It’s too soon to determine our key ability, but so far it’s a spirit of cooperation.

Lab 01 Due 2022.02.02;23:59 *[PHELAN contribution]*
Describe the activities you and your Team performed in response to the site [Jetson Community Projects], any results you obtained, and what you learned from the exercise. Include links and/or screenshots that document your results. Were you able to select a Team favorite project? Which ones did you find most interesting?

Not sure what is expected here. In only 8 days since the assignment there's not much activity we can do of any significance that would obtain any results to document. In our live discussion we reviewed the site, got a sampling of the possibilities of AI on the Jetson and expressed personal interest in several offerings. We're far from selecting a Team Project but discussed finding something between trivial and too difficult to accomplish during the course. What that will be is unclear. Perhaps we're reading too much into this?

Since the site suggests "Get Started with These Projects", it narrows it down to 4 choices: Jetbot, Hello AI World, Jetracer, and Pose Estimation. Having just resumed Tai Chi after over a decade of neglect, the Pose Estimation appeals to me but leans to the trivial. I've already done it with depthai examples using the OAK-D so there's not much there. My personal interest is getting the NASA/JPL Open Source Rover to be autonomous, so the Jetbot appeals to me as a toddler step in that direction. According to 3d party suppliers, the Jetracer requires more "advance programming skills" and fancier hardware than the Jetbot so should be deferred til later. There was some concern among team members about the availability of the necessary Jetbot hardware. Since I already have a Nano A2 and Roberto can 3D print the chassis, I think that hurdle is surmountable. That leaves Hello AI World to explore. The link gives no video examples but jumps right to github for code. I think there is an example program in the class materials so will explore that:

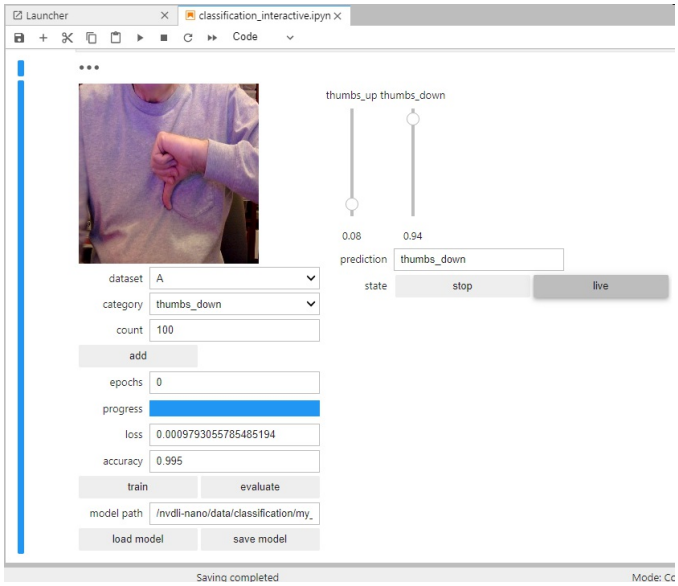
Jupyter notebook/classification/classification_interactive.ipynb

Does fine up until where it says "*Execute the cell below to create and display the full interactive widget.* ***Follow the instructions in the online DLI course pages to build your project***" which, after fruitlessly stumbling around the course material, I Googled "NVIDIA thumbs up thumbs down" and found here: https://www.youtube.com/watch?v=rSqIvLQ8Meg

Following the online tutorial I added 30 each of thumbs up and thumbs down images from the camera using my plain gray sweatshirt as a background. Set epochs to 10. Clicked “train” and waited. Finally after several minutes the progress bar appeared and the loss (initially 0.69..) and accuracy (0.75) began to appear with the first epoch. During training the camera latency was about 18 seconds as the CPU/GPU hogged all the processing power.

First epoch took several minutes then the epoch counter started counting down quickly as the progress bar marched across w/ each epoch. Finally loss 0.0047, accuracy 1 on the last epoch but each one varied.

Now “live”. First attempt wasn’t very accurate, missing several thumbs-up gestures. Deleted all the training images and started over. Second time trained 50 images each against a white towel and did much better. Mistakenly put some images in wrong category and had to delete and redo. Per instructors example, changed the background and the accuracy fell apart. Added 20 more images to each = 70. Some better but not good. Added 40 images each and retrained again for 10 epochs. Finally deleted all files and trained again with 100 poses each in front of towel. Seems the default vote is thumbs up, even in front of blank background. Thumbs down wasn’t perfect, but pretty good considering only 100 samples each. In front of shirt thumbs down was hard to recognize. In front of white closed blinds was pretty good despite training in front of different background.



The tools presented in this exercise made training an image classifier MUCH easier than reputed with other systems. Would be fun to try other classifiers in the course example.

2022.02.07

HCC West Loop Rm 152 in person meeting

Meet & greet. Good to be face to face! Edge vs Data Center computing. Advantages of computing on the edge. Powerpoint of microcontrollers (Arduino), microprocessors (Raspberry Pi) and Jetson Nano & its siblings. Discussion of possible class projects.

2022.02.08

Jetbot kit arrived today!! $100 jetbot, $100 Nano dev kit.

There was NO instruction manual so Googled “Waveshare jetbot assembly manual” and found this:

https://www.waveshare.com/wiki/JetBot_AI_Kit_Assemble_Manual

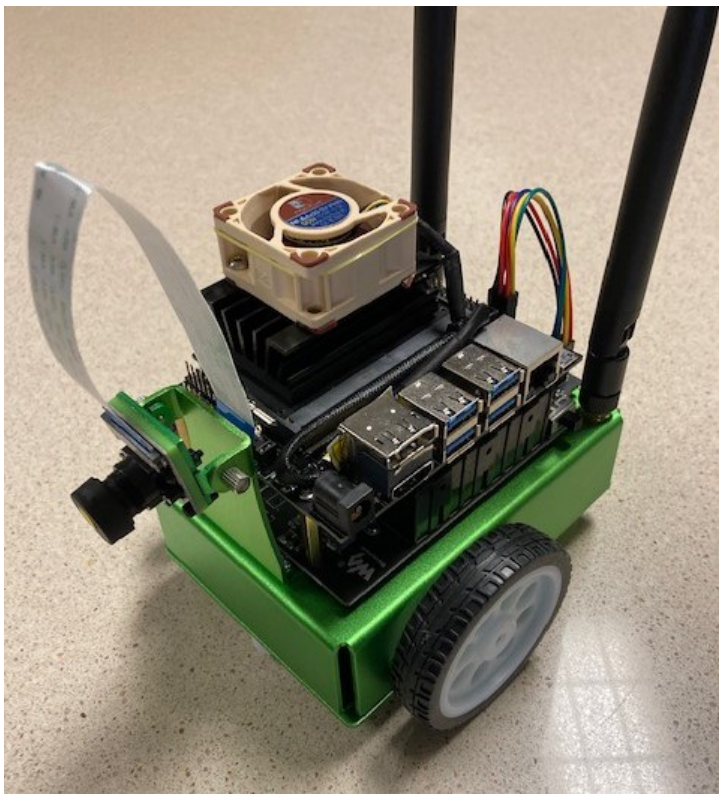
It’s TERRIBLE - incomplete.

Google then found me this:

https://www.youtube.com/watch?v=YfqKWalPpOw

Which did a MUCH better job through the mechanical part.

Now I need the software part.



2022.02.10

This is the home of the Jetbot:

https://jetbot.org/master/

which takes you to “Get Started”:

https://jetbot.org/master/getting_started.html

which takes you to “Software Setup” where I chose the 4GB image:

https://jetbot.org/master/software_setup/sd_card.html

New SD card formatted with SD Card Formatter and flashed with the extracted .img using Balena Etcher.

Booted SD card in Jetbot using battery power & locally configured.

BEFORE NEXT STEP TO $sudo apt update && sudo apt upgrade

Step 3 - Clone JetBot repo

ubuntu@JETBOT:~$ **git clone http://github.com/NVIDIA-AI-IOT/jetbot.git**

Step 4 - Configure System

ubuntu@JETBOT:~$ **cd jetbot**

ubuntu@JETBOT:~/jetbot$ **./scripts/configure_jetson.sh**

Hint

configure_jetson.sh also disables the GUI for the interest of saving system memory (DRAM) consumption.

If you want to re-enable the GUI, you can execute the following command.

sudo systemctl set-default graphical.target

Optionally, you can execute this script (./scripts/re_enable_gui.sh).

This is proven true with reboot.

ubuntu@JETBOT:~$ **./scripts/re_enable_gui.sh**

-bash: ./scripts/re_enable_gui.sh: No such file or directory

ubuntu@JETBOT:~$ **sudo systemctl set-default graphical.target**

Step 5 - Enable all containers

ubuntu@JETBOT:~/jetbot$ **cd docker**

./enable.sh $HOME   # we'll use home directory as working directory, set this as you please. SKIP TO BOTTOM OF PAGE

ubuntu@JETBOT:~/jetbot/docker$ **./enable.sh $HOME**

JETBOT_BASE_IMAGE not found for 32.6.1. Please manually set the JETBOT_BASE_IMAGE environment variable. (ie: export JETBOT_BASE_IMAGE=...)

[sudo] password for ubuntu:

Restarting docker daemon ...

Synchronizing state of docker.service with SysV service script with /lib/systemd/systemd-sysv-install.

Executing: /lib/systemd/systemd-sysv-install enable docker

Unable to find image 'jetbot/jetbot:display-0.4.3-32.6.1' locally

docker: Error response from daemon: manifest for jetbot/jetbot:display-0.4.3-32.6.1 not found: manifest unknown: manifest unknown.

See 'docker run --help'.

Unable to find image 'jetbot/jetbot:jupyter-0.4.3-32.6.1' locally

docker: Error response from daemon: manifest for jetbot/jetbot:jupyter-0.4.3-32.6.1 not found: manifest unknown: manifest unknown.

See 'docker run --help'.

ubuntu@JETBOT:~/jetbot/docker$ **sudo find / -name JETBOT_BASE_IMAGE**

nothing

Googling error message found this:

https://github.com/NVIDIA-AI-IOT/jetbot/issues/406

which advises change in configure.sh:

export L4T_VERSION="$L4T_RELEASE.$L4T_REVISION"   to

export L4T_VERSION="32.5.0"

After this and rerunning

ubuntu@JETBOT:~/jetbot/docker$ **./enable.sh $HOME**

a whole lot of files downloaded incl some unneeded files.

Advised Use 'sudo apt autoremove' to remove them.

ubuntu@JETBOT:~/jetbot/docker$ **sudo apt autoremove**

Now you can go to https://<jetbot_ip>:8888 from a web browser and start programming JetBot!

You can do this from any machine on your local network. The password to log in is **jetbot**.

http://192.168.1.14:8888/login?next=%2Flab%3F

jetbot

/jetbot/notebooks/basic_motion

This is the Jetbot camera FYI:

https://www.waveshare.com/imx219-160-camera.htm

Basic Motion

Welcome to JetBot's browser based programming interface! This document is called a Jupyter Notebook, which combines text, code, and graphic display all in one! Pretty neat, huh? If you're unfamiliar with Jupyter we suggest clicking the Help drop down menu in the top toolbar. This has useful references for programming with Jupyter.

In this notebook, we'll cover the basics of controlling JetBot.

Importing the Robot class

To get started programming JetBot, we'll need to import the Robot class. This class allows us to easily control the robot's motors! This is contained in the jetbot package.

If you're new to Python, a package is essentially a folder containing code files. These code files are called modules.

To import the Robot class, highlight the cell below and press ctrl + enter or the play icon above. This will execute the code contained in the cell

from jetbot import Robot

Now that we've imported the Robot class we can initialize the class instance as follows.

robot = Robot()

Commanding the robot

Now that we've created our Robot instance we named "robot", we can use this instance to control the robot. To make the robot spin counterclockwise at 30% of it's max speed we can call the following

WARNING: This next command will make the robot move! Please make sure the robot has clearance.

robot.left(speed=0.3)

Cool, you should see the robot spin counterclockwise!

If your robot didn't turn left, that means one of the motors is wired backwards! Try powering down your robot and swapping the terminals that the red and black cables of the incorrect motor.

REMINDER: Always be careful to check your wiring, and don't change the wiring on a running system!

Now, to stop the robot you can call the stop method.

robot.stop()

The robot didn’t spin correctly because the drive wheels flex upward just enough that the casters keep it from having good traction. I noted there’s a little spacer on the casters that might be removed. Let’s try that. The cover of the caster has a circle that corrals the 4 ball barrings that support the caster ball. Removing it doesn’t seem to inhibit the caster. It does lower the chassis about a millimeter which seems enough.

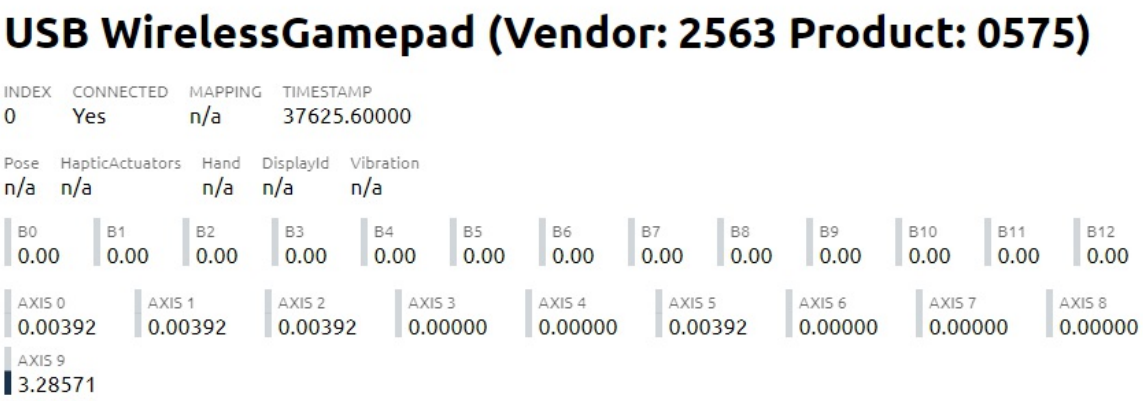
Teleoperation

Put 2x AA batteries in the Gamepad & place the dongle found in the battery compartment into a USB slot the machine running the notebook. This example seems to work best running the Jupyter notebook directly from the Nano desktop, rather than headlessly. Clicking the link in Jupyter will open a browser on the Nano and open the gamepad app. The first entry is the "index".

In this example we'll control the Jetbot remotely with a gamepad controller connected to our web browser machine.

The first thing we want to do is create an instance of the Controller widget, which we'll use to drive our robot. The Controller widget takes a index parameter, which specifies the number of the controller. This is useful in case you have multiple controllers attached, or some gamepads appear as multiple controllers. To determine the index of the controller you're using,

Visit http://html5gamepad.com.



Press buttons on the gamepad you're using*Remember the index* **[index = 0]** *of the gamepad that is responding to the button pressesNext, we'll create and display our controller using that index.*import ipywidgets.widgets as widgets

```
controller = widgets.Controller(index=0)  # replace with index of your controller

display(controller)
```

With some futzing, this works ok.

Later, when trying to display the camera, it doesn't find one, even tho

ubuntu@JETBOT:~$ **ls /dev/video***

/dev/video0

$ **cheese**

from the terminal command line or the desktop menu launches but fails to find the camera. Connections double checked. I think some camera configuration is missing. Pity, I was hoping to run the Jetbot around the house by teleop. Maybe the course Jupyter notebook examples to a better job? No such Notebook in the course. At least not at present. But the camera works in the HCC Jupyter notebook though the color is reddish (no cover on lens).

2022.02.12

Googled "jetbot teleoperation can't find camera" and found this:

https://developer.nvidia.com/blog/training-your-jetbot-in-isaac-sim/

which says after launching Jetbot docker (which seems automatic in latest iteration) to do this:

[this has to be done from **ubuntu@JETBOT:~/jetbot$**]

./scripts/enable_swap.sh

./docker/camera/enable.sh

ubuntu@JETBOT:~/jetbot$ **./scripts/enable_swap.sh**

```
### No swap memory currently configured on the system.
[sudo] password for ubuntu: ******
Setting up swapspace version 1, size = 4 GiB (4294963200 bytes)
no label, UUID=11c5c7be-b9d5-4f6e-ada9-3f8738c83fa3
# /etc/fstab: static file system information.
#
# These are the filesystems that are always mounted on boot, you can
# override any of these by copying the appropriate line from this file into
# /etc/fstab and tweaking it as you see fit.  See fstab(5).
#
# <file system> <mount point>            <type>          <options>        <dump> <pass>
/dev/root            /                    ext4           defaults              0 1
/swfile none swap sw 0 0
ubuntu@JETBOT:~/jetbot$ free -m
              total        used        free      shared  buff/cache   available
Mem:           3956        2007         584          56        1364        1722
Swap:          4095           0        4095
```

ubuntu@JETBOT:~/jetbot$ **./docker/camera/enable.sh**
docker: invalid reference format.
See 'docker run --help'.
If you see docker: invalid reference format, set your environment variables again by calling source configure.sh
ubuntu@JETBOT:~/jetbot$ **source configure.sh**
-bash: configure.sh: No such file or directory
ubuntu@JETBOT:~/jetbot/docker$ **source configure.sh**
JETBOT_BASE_IMAGE not found for 32.5.0. Please manually set the JETBOT_BASE_IMAGE environment variable. (ie: export JETBOT_BASE_IMAGE=...)

Synchronizing state of docker.service with SysV service script with /lib/systemd/systemd-sysv-install.

Executing: /lib/systemd/systemd-sysv-install enable docker

ubuntu@JETBOT:~/jetbot$ **./docker/camera/enable.sh**

Unable to find image 'jetbot/jetbot:camera-0.4.3-32.5.0' locally

camera-0.4.3-32.5.0: Pulling from jetbot/jetbot

[long list of docker elements “already exists”]

Digest: ...

Status: Downloaded newer image for jetbot/jetbot:camera-0.4.3-32.5.0 ...

ubuntu@JETBOT:~/jetbot$ **./docker/camera/enable.sh**

docker: Error response from daemon: Conflict. The container name "/jetbot_camera" is already in use by container ...

You have to remove (or rename) that container to be able to reuse that name.

See 'docker run --help'.

This is getting messy! I think I’ll do a clean install of the Jetbot.

ubuntu@JETBOT:~/jetbot/docker$ **sudo shutdown -h now**

SD card reader on PC is acting flakey!! REBOOT

DETOUR from Jetbot back to class assignments --

https://eagleonline.hccs.edu/courses/188486/assignments/3596711

which goes to

https://github.com/dusty-nv/jetson-inference

Go down to Hello AI World

JetPack already installed

go to Running to Docker Container

https://github.com/dusty-nv/jetson-inference/blob/master/docs/aux-docker.md

Choose 32.6.1 according to version in the L4T-Readme

cc the pull command & past into terminal command line:

ubuntu@NANO:~$ **docker pull dustynv/jetson-inference:r32.6.1**

Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post http://%2Fvar%2Frun%2Fdocker.sock/v1.24/images/create?fromImage=dustynv%2Fjetson-inference&tag=r32.6.1: dial unix /var/run/docker.sock: connect: permission denied

Even signing up and signing in on the Nano still FAILS!

2022.02.13

New day, new perspective.

Google “Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock:”

https://www.digitalocean.com/community/questions/how-to-fix-docker-got-permission-denied-while-trying-to-connect-to-the-docker-daemon-socket

“I’ve just installed docker but I have to run it with sudo every time. If I don’t add sudo I get the following error:”

Ok, so just add “sudo”. Bingo! NO! This is NOT the answer.

... Already exists

... Pull complete

Launching the Container

Due to various mounts and devices needed to run the container, it's recommended to use the docker/run.sh script to run the container:

DO “sudo apt update”, “sudo apt upgrade” again here first

ubuntu@NANO~:$ **git clone --recursive https://github.com/dusty-nv/jetson-inference**

long download...

ubuntu@NANO~:$ **cd jetson-inference**

ubuntu@NANO:~/jetson-inference$ **docker/run.sh**

Get:1-39

Fetched 7,825 kB in 10s

long pause

Running Applications

Once the container is up and running, you can then run example programs from the tutorial like normal inside the container:

cd build/aarch64/bin

./video-viewer /dev/video0

app didn’t discover any v4l2 device to when with dev/video0

seemed to do ok until...

[gstreamer] gstCamera -- end of stream (EOS)

[gstreamer] gstreamer v4l2src0 ERROR Internal data stream error.

[gstreamer] gstreamer Debugging info: gstbasesrc.c(3055): gst_base_src_loop ():

/GstPipeline0/GstV4l2Src:v4l2src0: streaming stopped,

reason not-negotiated (-4)

[gstreamer] gstreamer changes state from READY to PAUSED ==> mysink

video-viewer: failed to capture video frame

repeats......

Try again with others:

ubuntu@NANO:~/jetson-inference$ **docker/run.sh**

cd build/aarch64/bin

./imagenet images/jellyfish.jpg images/test/jellyfish.jpg

also FAIL

Let's go back and reinstall:

ubuntu@NANO~:$ **rm -rf jetson-inference**

ubuntu@NANO~:$ **git clone --recursive https://github.com/dusty-nv/jetson-inference**

ubuntu@NANO~:$ **cd jetson-inference**

ubuntu@NANO:~/jetson-inference$ **docker/run.sh**

Maybe don't do

./video-viewer /dev/video0

but instead go back to

Inference

Classifying Images with ImageNet

Using the ImageNet Program on Jetson

$ cd jetson-inference/build/aarch64/bin

has to be

$ **cd /jetson-inference/build/aarch64/bin**

./imagenet images/orange_0.jpg images/test/output_0.jpg

(default network is googlenet)

...

[TRT] could not register plugin creator - ::FlattenConcat_TRT version 1

Per https://github.com/dusty-nv/jetson-inference/issues/835

It is normal the first time that you load a model, it will take some minutes for TensorRT to optimize the model. It should only happen the first time that you run each model, because the optimized model gets cached to disk. If you let it run for a while it should complete the optimizations and then start the camera normally.

...

[TRT] requested fasted precision for device GPU without providing valid calibrator, disabling INT8

...error: model file 'networks/bvlc_googlenet.caffemodel' was not found.

If loading a built-in model, maybe it wasn't downloaded before.

Run the Model Downloader tool again and select it for download:

$ cd <jetson-inference>/tools

$ ./download-models.sh

This FAILS w/ the same repeat directions. Since it's looking to googlenet, doubt it's a built-in model.

./imagenet.py images/orange_0.jpg images/test/output_0.jpg # (default network is googlenet)

also fails for same reason, no surprise.

Google'ing errors not helpful x- suggesting update/upgrade

^C, ^D to exit docker

$ **sudo apt update**

$ **sudo apt upgrade**

of course, should have done this FIRST!

This time ubuntu@NANO:~/jetson-inference$ **docker/run.sh**

has a pop-up requesting model download!

Accept **"Image Recognition - all models (2.2 GB)"**

bunch of files download.

root@NANO:/jetson-inference/build/aarch64/bin# **./imagenet images/orange_0.jpg images/test/output_0.jpg**

Does a LOOOOOONG list of [TRT] stuff, then finishes, saying

[image] saved ‘images/test/output_0.jpg’ (1024x683, 3 channels)

UNFORTUNATELY it’s saved in root@NANO and won’t display with eog which is installed in ubuntu@NANO nor can it find it to install!

The root@NANO:/jetson-interence/build/aarch64/bin/images/ directory is not accessible from ubuntu@NANO so output_0.jpg can’t be displayed. According to file manager the directory doesn’t even exist! I guess because it’s in a container that doesn’t exist outside itself.

I suspect this is because “sudo” had to be used to install the docker above making the whole thing inaccessible. Must try to re-install without “sudo”.

ubuntu@NANO~:$ **rm -rf jetson-inference**

make ubuntu super user: $ **sudo**

ubuntu@NANO:~$ **docker pull dustynv/jetson-inference:r32.6.1**

FAILS AGAIN permission denied.

Google again “**Got permission denied while trying to connect to the Docker daemon socket at unix**”

According to:

https://www.digitalocean.com/community/questions/how-to-fix-docker-got-permission-denied-while-trying-to-connect-to-the-docker-daemon-socket

According to the official Docker docs here:

https://docs.docker.com/install/linux/linux-postinstall/#manage-docker-as-a-non-root-user

You need to do the following:

To create the docker group and add your user:

Create the docker group.

sudo groupadd docker

Add your user to the docker group.

sudo usermod -aG docker ${USER}

You would need to log out and log back in so that your group membership is re-evaluated or type the following command:

su -s ${USER}

Verify that you can run docker commands without sudo.

docker run hello-world FAILS

ubuntu@NANO:~$ **groups**

ubuntu adm cdrom sudo audio dip video plugdev i2c lpadmin gdm lightdm sambashare weston-launch gpio

No docker here

ubuntu@NANO:~$ **sudo groupadd docker**

[sudo] password for ubuntu:

groupadd: group 'docker' already exists

ubuntu@NANO:~$ **sudo usermod -aG docker ${USER}**

Doesn’t help.

I think I need to start over & keep class & Jetbot separate SD cards & load docker w/o sudo if I can.

2022.02.14 ❤ HCC West Loop ITAI 2367

Based on my noted Diane discovered bad download link in syllabus.

There are too many variants of the Jetbot OS that are mutually incompatible. U. Md has one that fits all & will be linked to this week, prob Wed. I need to start over & download THAT image and keep any other projects for the Nano on separate SD cards. Ordered another 6 Sandisk 128GB micro SD cards from Amazon.

Also, the SD card reader on my laptop is DEFINITELY not working! D’ gave me an IOGEAR USB SD/microSD card reader which is working ok! Thanks D’!!

2022.02.18

The Jetson OS update promised by D’ isn’t ready:

Jetbot

Starting over from software install:

https://jetbot.org/master/software_setup/sd_card.html

Step 2 - Flash JetBot image onto SD card

Attention

To use one of the JetBot sdcard images based on JetPack 4.5, you first need to boot your Jetson Nano using a plain JetPack 4.5 SD card image and run through the operating system setup. This will perform a one-time configuration which enables you to use SD card images based on JetPack 4.5 on your device. You can find the original JetPack SD card images here: JetPack SD card image for Jetson Nano 2GB and JetPack SD card image for Jetson Nano (4GB). After doing this procedure once, you can then use the JetPack 4.5 based JetBot SD card images listed above on your device.

C:\Users\Me\Documents\My Downloads\Robotics\USAi Labs\HoustonCommunityCollege\ITAI 2374-01 (22680) Robot Operating Sys & Platform\JETBOT>**certutil -hashfile jetbot-043_nano-4gb-jp45.zip MD5**

760b1885646bfad8590633acca014289 [as expected]

SD Card Formatter: overwrite

BalenaEtcher: above .zip file

Step 3 - Boot Jetson Nano

Boot on Nano: **jetbot / jetbot 192.168.1.14**

jetbot@nano-4gb-jp45:~$ **sudo apt update**

jetbot@nano-4gb-jp45:~$ **sudo apt upgrade**

```
Processing triggers for initramfs-tools (0.130ubuntu3.13) ...

update-initramfs: Generating /boot/initrd.img-4.9.201-tegra

cryptsetup: WARNING: failed to detect canonical device of /dev/root

cryptsetup: WARNING: could not determine root device from /etc/fstab

Warning: couldn't identify filesystem type for fsck hook, ignoring.

/sbin/ldconfig.real: Warning: ignoring configuration file that cannot be opened:
/etc/ld.so.conf.d/aarch64-linux-gnu_EGL.conf: No such file or directory

/sbin/ldconfig.real: Warning: ignoring configuration file that cannot be opened:
/etc/ld.so.conf.d/aarch64-linux-gnu_GL.conf: No such file or directory
```

```
Processing triggers for nvidia-l4t-kernel (4.9.201-tegra-32.5.2-20210709090126) ...
Processing triggers for libc-bin (2.27-3ubuntu1.4) ...
Errors were encountered while processing:
 nvidia-l4t-bootloader
E: Sub-process /usr/bin/dpkg returned an error code (1)
```

Step 4 - Connect JetBot to WiFi

Connect to a WiFi network using the following command

sudo nmcli device wifi connect <SSID> password <PASSWORD>

Device 'wlan0' successfully activated with '46f8cfcc-3644-456e-a9ca-f204dc18a500'.

WiFi 192.168.1.15

Step 5 - Connect to JetBot from web browser

After your robot is connected to WiFi, you no longer need to have the robot connected by a monitor. You can connect to the robot from your laptop's web browser by performing the following steps *[this doesn’t seem necessary but might be good to do for practice]*

Shutdown JetBot using the command line

```
sudo shutdown now
```

Unplug your HDMI monitor, USB keyboard, mouse and power supply from Jetson Nano

Power the JetBot from the USB battery pack by plugging in the micro-USB cable

[this doesn’t apply here as power comes through powerpack ribbon cable]

Wait a bit for JetBot to boot

Check the IP address of your robot on the piOLED display screen. Enter this in place of <jetbot_ip_address> in the next command

Navigate to http://<jetbot_ip_address>:8888 from your desktop's web browser. You can do this from any machine on your local network.

Sign in using the password jetbot.

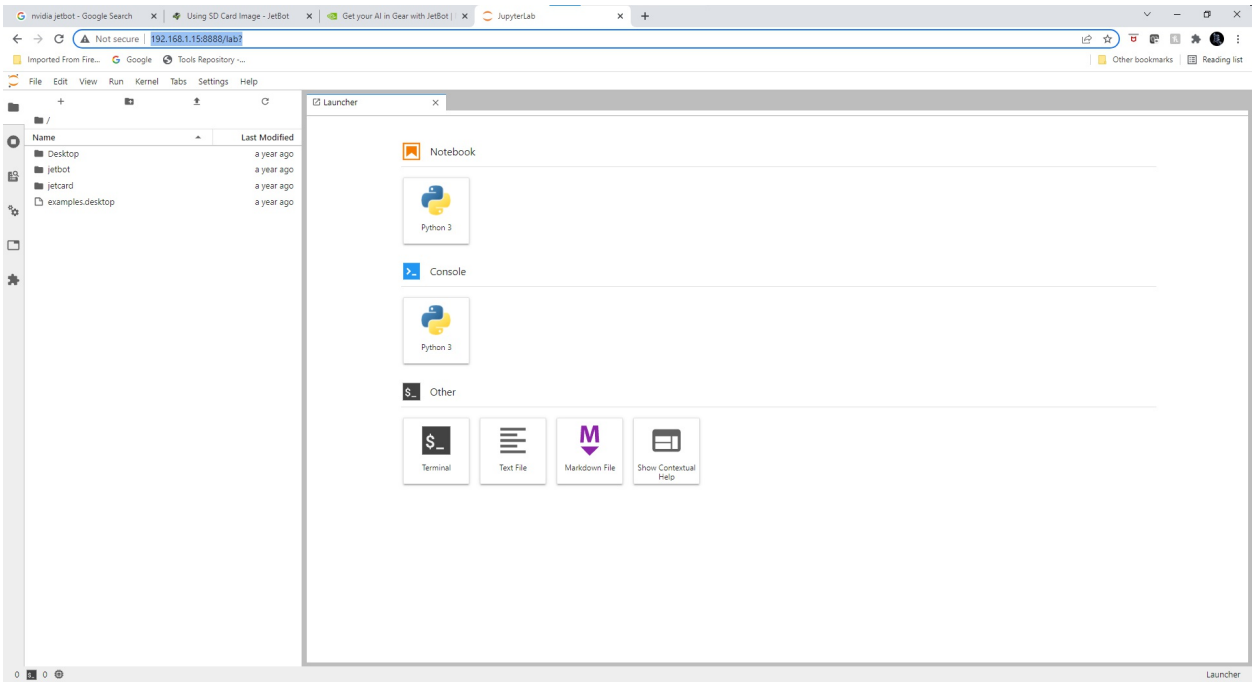
That's it, you've now accessed JetBot's remote programming environment!

I note the LED screen doesn’t update the WiFi IP address unless rebooted.

jetbot@nano-4gb-jp45:~$ **sudo reboot**

http://192.168.1.15:8888

You will be presented with a view similar to the following.



Here you can easily access the JetBot examples! From this point on, when you power on the JetBot, it should automatically connect to WiFi and display it's IP address. So all you need to do is reconnect using your web browser to start programming!

Now that you're finished setting up your JetBot, you're ready to run the **examples**.

https://jetbot.org/master/examples/basic_motion.html

Basic Motion

In this notebook we'll control JetBot by programming from a web browser.

3. Navigate to ~/jetbot/notebooks/basic_motion/

4. Open and follow the basic_motion.ipynb notebook

Caution

JetBot will physically move in this notebook, make sure it has enough space to move around.

2022.02.19

basic_motion Jupyter notebook works ok. Now on to teleoperation.

Teleoperation

jetbot/notebooks/teleoperation

teleoperation.ipynb

Create gamepad controller

Visit http://html5gamepad.com.

No gamepad detected.

Connect gamepad controller to robot motors

```
from jetbot import Robot

import traitlets

robot = Robot()

left_link = traitlets.dlink((controller.axes[1], 'value'), (robot.left_motor, 'value'),
transform=lambda x: -x)

right_link = traitlets.dlink((controller.axes[3], 'value'), (robot.right_motor, 'value'),
transform=lambda x: -x)
```

```
IndexErrorTraceback (most recent call last)

<ipython-input-6-12e166046241> in <module>

      4 robot = Robot()

      5

----> 6 left_link = traitlets.dlink((controller.axes[1], 'value'), (robot.left_motor, 'value'),
transform=lambda x: -x)

      7 right_link = traitlets.dlink((controller.axes[3], 'value'), (robot.right_motor, 'value'),
transform=lambda x: -x)


IndexError: tuple index out of range
```

Google “jetbot teleoperation tuple index out of range”:

https://github.com/NVIDIA-AI-IOT/jetbot/issues/60

joezen777 commented on Apr 28, 2020 •

A couple of things.

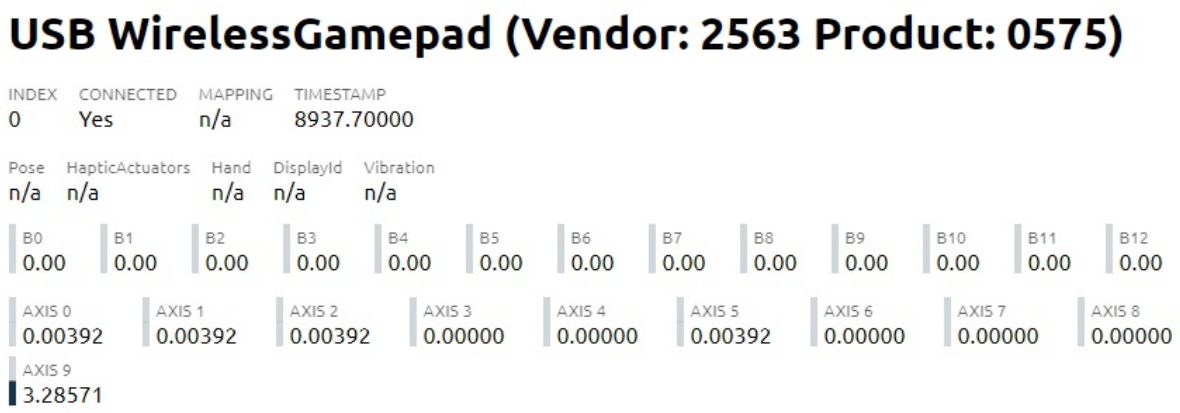
unlink after successfully linking (before linking again)

The remote controller USB dongle goes on the computer you're viewing the notebook on

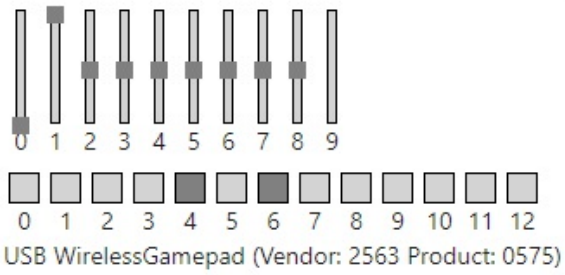
I had to press home twice to get two lights to show up on the game pad, and then I could link axis 5 to the right wheel, where as in any other mode only the left wheel gets hooked up to axis[1]. View the html5gamepad or starting widget display to validate you're using a valid axis.

Move gamepad dongle from Jetbot to laptop.

Interrupt kernel & close Jupyter page & start over.



Note: INDEX = 0



Connect gamepad controller to robot motors

For some reason, only the left motor/wheel is turning!?

Was it messed up by basic_motion sliders? Or by being bumped around during the basic_motion exercises?

3. Navigate to ~/jetbot/notebooks/basic_motion/

4. Open and follow the basic_motion.ipynb notebook

While playing w/ basic_motion, connection lost. Looking at Jetbot, no power. Plugged in power cable, no change. Flipped on power switch & booted back up. Maybe got switched off during maneuvers? Need to run Jetbot's stats.py to see voltage? (Later)

Playing with “heartbeat” function, at about 0.10 the motor will cut off after several seconds.

FORGET stats.py DEAD END!

Pay attention to this page:

https://www.waveshare.com/wiki/JetBot_AI_Kit

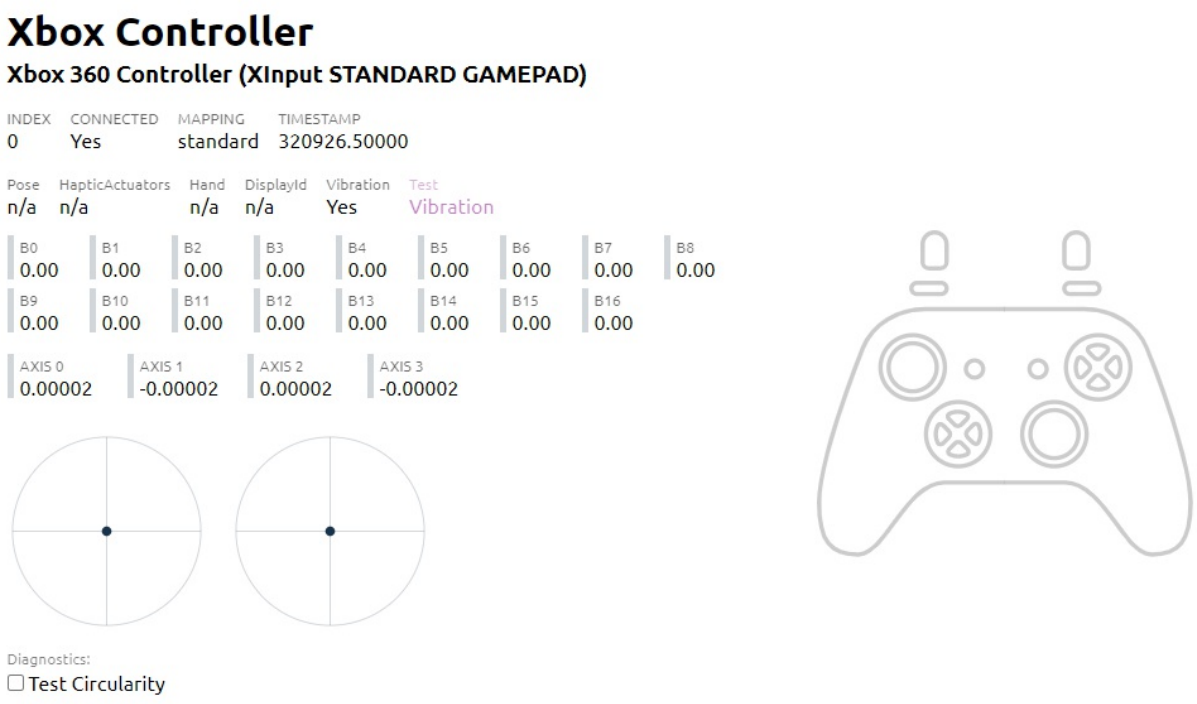
3. Teleoperations

Go to https://html5gamepad.com, check the INDEX of Gamepad

Before you run the example, let's learn how the gamepad work.

The gamepad included supports two working modes. One is PC/PS3/Andorid mode and another is Xbox 360 mode.

The gamepad is set to PC/PS3/Andorid mode by default, in this mode, the gamepad has two sub-modes. You can press the HOME button to switch it. In Mode 1, the front panel lights on only one LED, the right joystick is mapped to buttons 0,1,2 and 3, and you can get only 0 or -1/1 value from the joysticks. In Mode 2, the front panel lights on two LEDs, the right joystick is mapped to axes[2] and axes[5]. In this mode, you can get No intermediate values from joysticks.



To switch between PC/PS3/Andorid mode and the Xbox 360 mode, you can long-press the HOME button for about 7s. In Xbox mode, the left joystick is mapped to axes[0] and axes[1], right joystick is mapped to axes[2] and axes[3]. This mode is exactly what the NVIDIA examples use. We recommend you to set your gamepad to this mode when you use it. Otherwise, you need to modify the codes.*Be sure you’re in THIS MODE or the right motors won’t run!*

I was able to pilot the Jetbot around the house remotely.

I don't particularly like the tank-like steering with:

left-joystick = left motor

right-joystick = right motor

It makes steering awkward and jerky. I tried changing some of the code to put both motors on one joystick in normal fashion but didn't know how to do it. Reverted. I did modify according to:

https://forums.developer.nvidia.com/t/jetson-4gb-waveshare-jetbot-ps3-control-python-code/176823/3

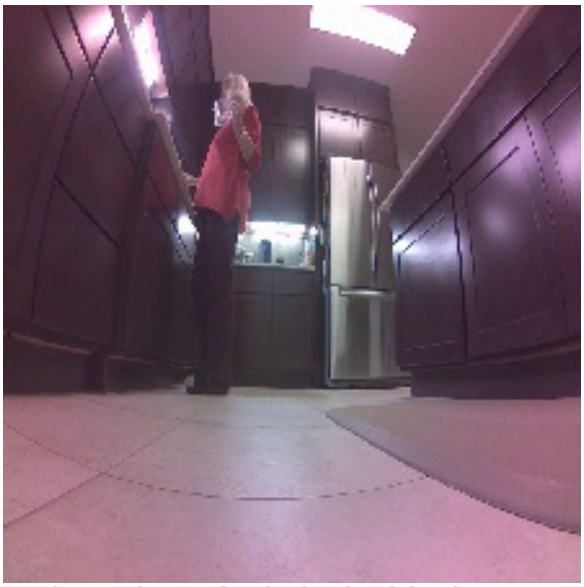
```
speed = 0.5

left_link = traitlets.dlink((controller.axes[0], 'value'),
(robot.left_motor, 'value'), transforms=lambda x: - speed * x)

right_link = traitlets.dlink((controller.axes[1], 'value'),
(robot.left_motor, 'value'), transforms=lambda x: - speed * x)
```

which slowed down the motors and made navigation smoother, especially navigating via the remote video stream!

For future reference, pink tint issue may be solved here:

https://github.com/NVIDIA-AI-IOT/jetbot/discussions/465

Jetbot view of Jim in study piloting



Jetbot view of Linda in kitchen

Collision Avoidance

https://jetbot.org/master/examples/collision_avoidance.html

In this example we'll collect an image classification dataset that will be used to help keep JetBot safe! We'll teach JetBot to detect two scenarios free and blocked. We'll use this AI classifier to prevent JetBot from entering dangerous territory.

Step 1 - Collect data on JetBot

3.Shutdown all other running notebooks by selecting Kernel -> Shutdown All Kernels...

4.Navigate to ~/Notebooks/collision_avoidance/

5.Open and follow the data_collection.ipynb notebook

Placing the Jetbot around the room and hallway in both "free" and "blocked" positions is tedious and tiring! The docs don't say how many you need! I've got 30+ each so far and figure I need at least 50 based on previous "thumbs up/down" lab.

Ultimately, the more data we have of scenarios the robot will encounter in the real world, the better our collision avoidance behavior will be. It's important to get varied data (as described by the above tips) and not just a lot of data, but you'll probably need at least **100 images of each class** (that's not a science, just a helpful tip here). But don't worry, it goes pretty fast once you get going :)

It would be great to combine the teleoperation and data_collection functions so you could drive the Jetbot around and click one button for "free" and another for "blocked"!

Would also be good to be able to display the free and blocked dataset folders with thumbnails to review and edit the images so you could delete/move any in the wrong category.approve

Step 2 - Train neural network

You should see a file named dataset.zip in the Jupyter Lab file browser. You should download the zip file using the Jupyter Lab file browser by right clicking and selecting Download.

Next, we'll need to upload this data to our GPU desktop or cloud machine (we refer to this as the host) to train the collision avoidance neural network. We'll assume that you've set up your training machine as described in the

JetBot WiKi.[??] If you have, you can navigate to http://<host_ip_address>:8888 to open up the Jupyter Lab environment running on the host. The notebook you'll need to open there is called collision_avoidance/**train_model.ipynb**. So head on over to your training machine and follow the instructions there! Once your model is trained, we'll return to the robot Jupyter Lab enivornment to use the model for a live demo!

But in https://jetbot.org/master/examples/collision_avoidance.html

it says “Open and follow the **train_model_resnet18.ipynb** notebook”

so which one do I follow?? [texted R&D]

Response from R&D:

“I have used resnet - there are a few utilities you need loaded. Dusty has a good forum about this. You might check his forum if you have any difficulties. Either one is good. alexnet is great for first time and is really fast. It makes you realize you can train at the edge which has always been the question mark. Dusty also has a good overview for alexnet. Alexnet is easiest. It is a good start.”

Dusty??

“There is a Jetson Nano forum ongoing which provides helpful information about notebooks, configurations, changes, links to new posts about helpful recommendations. This is part of the Jetson course downloads. It is a nice eco system that covers all Jetsons (nano, Xavier, Agx, etc.).”

In cc SD card to laptop there were MULTIPLE partitions E:-R:

Found online way to remove all but R:. Then cc image to laptop. Still need a way to change R: > E:. Later.

Rebooted SD card to Jetbot & will go with Alexnet to start.

train_model.ipynb

2022.02.21

This notebook has been running for 20 hrs with no output to suggest even 1 epoch has completed!

$ **top**

python3

python3

irq/74-tegradc

sogov:0

jupyter-labs & kworker/1.0 tie for 5th

Call to R&D w/ problem. Directed to NVIDIA forums:

https://forums.developer.nvidia.com/tags/c/agx-autonomous-machines/jetson-embedded-systems/jetson-nano/76/jetbot

jetbot@nano-4gb-jp45:~$ **sudo apt-get remove chromium-browser**

uninstall processes... then

E: Sub-process /usr/bin/dpkg returned an error code (1)

can't get chromium-browser to load on Jetbot. Jetpack gets in the way, apparently.

Killed train_model notebook /p 22 hrs of no output.

D' suggested creating smaller 10 vs 100 images each for test case.

Created FREE_LARGE and BLOCKED_LARGE directories next to free and blocked. Moved all files from free into FREE_LARGE and all files from blocked into BLOCKED_LARGE. Then moved first 10 files in each back to the regular free and blocked directories. Now I'll have a test case w/ just 10 samples each which will be split 5:5 into random training sets.

Skipped over unzip as already unzipped to free & blocked.0

Changed 50, 50 to 5, 5 in code:

```
train_dataset, test_dataset =
torch.utils.data.random_split(dataset, [len(dataset) 5, 5])
```

I see a problem here: dataset contains 4 files: free, blocked, FREE_LARGE, BLOCKED_LARGE. Need to move _LARGE out of dataset.

Try notebook again.

2022.02.22

Overnight - nothing!

20 hrs - nothing!

Bottom of Jupyter notebook says “No Kernel | Idle”

Adding

    print(‘Begin’)

as first line of notebook produces nothing.

Need to dig deeper into Jupyter Notebooks.

I THOUGHT I ran train_model overnight. But I learned something about jupyter notebook... if I mess up and want to start over and ‘Shut Down Kernel’ it doesn’t restart when I start the notebook over. It just sits there but doesn’t complain if you try to use it. I happened to notice in the corner ‘No kerenel | Idle’ when it was supposed to be running. I put a ‘print(‘Begin’)’ statement at the beginning and it didn’t print when “run”. Went to Kernel / Restart Kernel. Now Python3 appears upper right and Python3 | Idle in lower left. ‘print (‘Begin’)’ now produces ‘Begin’! I had reduced the sample set to 10 each and set [lef(dataset) - 5, 5] down from 50, 50.

Retrying train_model.ipynb now quickly produces the promised best_model.pth file quite quickly! I didn’t see any output from

    print ('%d: %f' % (epoch, test_accuracy))

however.

Now to try again w/ 100 each....

Only have 95 in free ?!? Lost 5 somewhere in translation.

So reduce both sets to 94 divided into 47 member sets:

```
train_dataset, test_dataset =
torch.utils.data.random_split(dataset, [len(dataset) - 47, 47])
```

Once last cell run took < 1min to finish best_model.pth!

2022.02.23

Step 3 - Optimize the model on Jetson Nano

Open and follow the **live_demo_resnet18_build_trt.ipynb** notebook to optimize the model with TensorRT

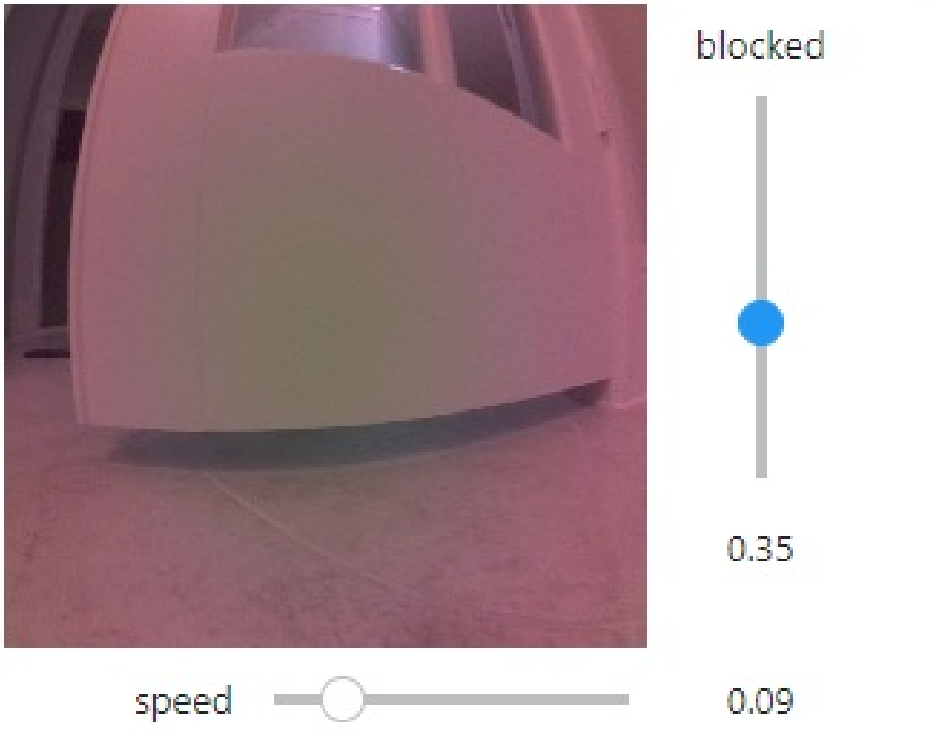
once done “Open **live_demo_resnet18_build_trt.ipynb** to move JetBot with the TensorRT optimized model.”

[No, you just ran this notebook. The Jetbot tutorial instructions say: “Open and follow the **live_demo_resnet18_trt.ipynb** notebook to run the optimized model” *which seems more logical*]

Step 4 - Run live demo on JetBot

Open and follow the **live_demo_resnet18_trt.ipynb** notebook to run the optimized model

Need to be sure the browser in connected to the Jetbot via WiFi and not ethernet as it needs to be unplugged to rove!



Clearly it is not recognizing when it’s getting blocked! It repeatedly gets itself stuck in corners or against walls!

2022.02.24

Team Robotic-Insight GroupMe requesting attention to Lab 03:

L03 Lab ITAI 2374

Prepare for this laboratory. Explore the GitHub site and get ready to run the lab. Go through the files: look at the image files, read the Jupyter notebook and Python files, just get familiar with what is contained there. Go over it, try it out if you want. We will be running it next week, so this is a look-ahead and some practice. Have your Team present your work at the class meeting of February 28. (Our class will not meet Monday, February 21 for the President's day holiday. On the class meeting of February 28, you will apply what you have learned to the setup of your Jetson Nano. Two hours to work on the Nano, and then about 15 minutes for a Team presentation.)

Several files are posted in Module 03 under Lab 03, because the link to the files (as shown in the image above) don't work because they become specific to the local computer. You will be presenting in our class meeting of February 28, and your grade for both L03 and L04 will be determined by this hands-on demonstration of what you learned. (You need not submit an online file for this set of labs.)

Laboratory

boat.mp4

Attachment

boat.mp4

hallway.mp4

Attachment

hallway.mp4

circles.png

Attachment

circles.png

Module-1-Lab-processing-images.html

Attachment

Module-1-Lab-processing-images.html

lena_color.tif

Attachment

lena_color.tif

Module-1-Lab-processing-images.py

Attachment

Module-1-Lab-processing-images.py

Module-1-Lab-processing-images.ipynb

Attachment

Module-1-Lab-processing-images.ipynb

Lab 03.zip

Attachment

Lab 03.zip

Reinstall Jetpack from scratch for class labs (SEPARATE from Jetbot SD Card!) **jetbot-043_nano-4gb-jp45.zip**

Boot on Nano and configure.

2022.02.25

$ **sudo apt-get update**

$ **sudo apt-get upgrade**

Setup / System Settings / Networks / WiFi setup for local WiFi

Carrying on from NVIDIA online course instructions:

https://courses.nvidia.com/courses/course-v1:DLI+S-RX-02+V2/courseware/b2e02e999d9247eb8e33e893ca052206/63a4dee75f2e4624afbc33bce7811a9b/?child=first

ubuntu@ubuntu:~$ **free -m**

```
              total        used        free      shared  buff/cache   available
Mem:           3956        1151         559          36        2244        2624
Swap:          1978           1        1976
```

ubuntu@ubuntu:~$ **sudo systemctl disable nvzramconfig**

ubuntu@ubuntu:~$ **sudo fallocate -l 4G /mnt/4GB.swap**

ubuntu@ubuntu:~$ **sudo chmod 600 /mnt/4GB.swap**

ubuntu@ubuntu:~$ **sudo mkswap /mnt/4GB.swap**

ubuntu@ubuntu:~$ **sudo su**

root@ubuntu:/home/ubuntu# **echo "/mnt/4GB.swap swap swap defaults 0 0" >> /etc/fstab**

root@ubuntu:/home/ubuntu# **exit**

ubuntu@ubuntu:~$ **sudo reboot**

Download Docker And Start JupyterLab

6.

ubuntu@ubuntu:~$ **mkdir -p ~/nvdli-data**

7. Run the Docker container

**sudo docker run --runtime nvidia -it --rm --network host **

**   --volume ~/nvdli-data:/nvdli-nano/data **

**   --device /dev/video0 **

**   nvcr.io/nvidia/dli/dli-nano-ai:v2.0.2-r32.6.1kr**

Long download & unpack...

To create and run a reusable script for this step try the following (example tag shown).

If using the alternate CSI camera instead of the USB webcam, add --volume /tmp/argus_socket:/tmp/argus_socket to your docker run command. For example:

echo "sudo docker run --runtime nvidia -it --rm --network host \

    --volume ~/nvdli-data:/nvdli-nano/data \

    --volume /tmp/argus_socket:/tmp/argus_socket \

    --device /dev/video0 \

    nvcr.io/nvidia/dli/dli-nano-ai:v2.0.1-r32.6.1" >

docker_dli_run.sh

make the script executable

ubuntu@ubuntu:~$ **chmod +x docker_dli_run.sh**

run the script   *[do this NEXT TIME you need to load]*

ubuntu@ubuntu:~$ **./docker_dli_run.sh**

docker: Error response from daemon: failed to create shim: OCI runtime create failed: container_linux.go:380: starting container process caused: error adding seccomp filter rule for syscall clone3: permission denied: unknown.

Google error message.

ubuntu@ubuntu:~$ **dockerd -v**

Docker version 20.10.7, build 20.10.7-0ubuntu5~18.04.3

Logging Into The JupyterLab Server

1. Open the following link address : 192.168.55.1:8888

won’t connect

try 192.168.1.22

won’t connect

try 192.168.1.15

won’t connect

WTF? This ran before w/ these directions!

Try this one online suggestion:

sudo apt-get install docker.io=20.10.7-0ubuntu1~20.04.2

E: Version '20.10.7-0ubuntu1~20.04.2' for 'docker.io' was not found

START OVER!!

Diane promises revised Docker files for the Nano for class, pending...

2022.03.05

Report from D' at USAi Labs online meeting:

Go to Canvas / Dashboard / ITAI 2374 / Modules / Module 04 /

Main Course Presentation /

ITAI 2374 Robot OS & Platforms 04 B Spr 22 Final.pdf - DOWNLOAD

Initial Jetson Nano setup already done in class last week.

I want to change computer name (hostname) from "ubuntu" to "**phelan**" in contrast to "hcc" on the class Nano's.

Needed to install nano (text editor, not Jetson hardware) in order to edit /etc/hostname & /etc/hosts. Done.

WiFi setup via system settings. Done. 192.168.1.15

Eth0 192.168.1.22

Headless setup - PuTTY already on laptop. ubuntu@phelan:~$

SAVE IMAGE of Nano pre-Docker download w/ current config. Done.

Beginning w/ p28 of above .pdf:

Run docker Command

First create a directory

ubuntu@phelan:~$ **mkdir -p ~/nvdli-data**

Download the container

ubuntu@phelan:~$ **sudo docker pull nvcr.io/nvidia/l4t-ml:r32.6.1-py3**

[multiple files download, verify, extract]

Status: Downloaded newer image for nvcr.io/nvidia/l4t-ml:r32.6.1-py3

Now run the container (be sure your camera is attached):

ubuntu@phelan:~$ **sudo docker run --runtime nvidia -it --rm --network host --volume ~/nvdli-data:/nvdli-nano/data --device /dev/video0 nvcr.io/nvidia/l4t-ml:r32.6.1-py3** *[all on 1 line]*

docker: Error response from daemon: failed to create shim: OCI runtime create failed: container_linux.go:380: starting container process caused: error adding seccomp filter rule for syscall clone3: permission denied: unknown.

Looking at link from class ppt:

https://catalog.ngc.nvidia.com/orgs/nvidia/teams/dli/containers/dli-nano-ai

Selecting CSI camera option:

```
sudo docker run --runtime nvidia -it --rm --network host \
 --volume ~/nvdli-data:/nvdli-nano/data \
 --volume /tmp/argus_socket:/tmp/argus_socket \
 --device /dev/video0 \
 nvcr.io/nvidia/dli/dli-nano-ai:v2.0.1-r32.6.1
```

```
ubuntu@phelan:~/nvdli-data$ sudo docker run --runtime nvidia -it --rm --network host \
>  --volume ~/nvdli-data:/nvdli-nano/data \
>  --volume /tmp/argus_socket:/tmp/argus_socket \
>  --device /dev/video0 \
docker: invalid reference format.
See 'docker run --help'.
```

It doesn't seem to process this part of the code, else that's what's causing the error:

```
     nvcr.io/nvidia/dli/dli-nano-ai:v2.0.1-r32.6.1
```

2022.03.07 HCC West Loop ITAI 2374 Class

Per D':

Needed to do Software Update (not just sudo apt update)

Also, download latest SD card image from

http://developer.nvidia.com/embedded/downloads

Choose Jetson Nano / first image, (NOT 2GB image):

jetson-nano-jp461-sd-card-image

SHA256: 735FEA3DF2509436CE43E480F2E70D633F0ADFE84007ED9CE7F43910E3814168

Can't download from class, no internet.

Roberto copied it for me!

2022.03.08 Home

Installed 128GB micro SD card flashed w/ above file & booted.

Personal and Local configurations.

Configure WiFi to local settings.

*Bluetooth **enable visible***

Reboot

Software Updater "235.8 MB will be installed.

Install now. Restart

Settings / Brightness & Lock

Turn screen off when inactive for **1 hour**

Lock > **OFF**

□ Require my password when waking from suspend. **UNCHECK**

Following Jetson AI Fundamentals - S1E1 - First Time Setup with JetPack:

https://www.youtube.com/watch?v=uvU8AXY1170

Set up 4GB swap memory.

ubuntu@PHELAN:~$ **free -m**

```
              total        used        free      shared  buff/cache
available
Mem:           3956        1347        1754          29         854
2447
Swap:          1978           0        1978
```

ubuntu@PHELAN:~$ **sudo systemctl disable nvzramconfig**

```
[sudo] password for ubuntu:
Removed /etc/systemd/system/multi-user.target.wants/nvzramconfig.service.
```

ubuntu@PHELAN:~$ **sudo fallocate -l 4G /mnt/4GB.swap**

ubuntu@PHELAN:~$ **sudo chmod 600 /mnt/4GB.swap**

ubuntu@PHELAN:~$ **sudo mkswap /mnt/4GB.swap**

```
Setting up swapspace version 1, size = 4 GiB (4294963200 bytes)
no label, UUID=6295f9f8-3502-455d-af40-3e94aff13713
```

ubuntu@PHELAN:~$ **sudo nano /etc/fstab**

```
sudo: nano: command not found
```

ubuntu@PHELAN:~$ **sudo apt install nano**

ubuntu@PHELAN:~$ **sudo nano /etc/fstab**

add the following:

/mnt/4GB.swap swap swap defaults 0 0

REBOOT

ubuntu@PHELAN:~$ **free -m**

```
              total        used        free      shared  buff/cache
available
Mem:           3956         984        2217          26         754
2799
Swap:          4095           0        4095
```

Module 04 / Lab 04:

Play Audio through speaker. First Image off Camera. Connect WiFi. Connect Bluetooth.

Was able to play various online radio stations from Rhythmbox through my new JBL Bluetooth speaker! Also Filezilla'd music from laptop > Nano & played it: "Drive it Like You Stole It", The Glitch Mob. Chosen for appropriateness to mobile robot.

First Nano photo!

2022.03.10

P04 Puzzle ITAI 2374 2022Spr

Which is best, headless operation or using a monitor?

Justify your opinion, perhaps in different situations. Use a Word file for your submission.

I think this is a solo Puzzle?

To free, or not to free: that is the question! Whether tis nobler for the Nano to suffer the links and cables of headed operation, or to take to WiFi, PuTTY and SSH, and by unplugging end them?

Well, that depends. If the Nano isn't going to be moving, as a robot, and just sit there training neural networks, then there's a distinct advantage of direct connection with the monitor being able to display the full Operating System desktop and direct access with keyboard and mouse without any transmission bottlenecks. There's also the security of being isolated from the Internet.

However, if the Nano needs to move as with a JetBot, then dragging a monitor, keyboard, and mouse behind like "Just Married" tin cans is more than awkward! With the right connections, Remote Desktop is almost as good. Alternatively, the Nano can act as a web server and deliver via http useful apps and images to a remote browser and receive commands via the remote keyboard, mouse, game controller, or other input device. There can be problems of bandwidth and latency using a remote connection so it's best to do as much preprocessing as possible on the sending side of the connection.

2022.03.10

Now to continue from class slideshow Module 04 p28/45:

Run docker Command

First create a directory

ubuntu@phelan:~$ **mkdir -p ~/nvdli-data**

Download the container

ubuntu@phelan:~$ **sudo docker pull nvcr.io/nvidia/l4t-ml:r32.6.1-py3**

[multiple files download, verify, extract]

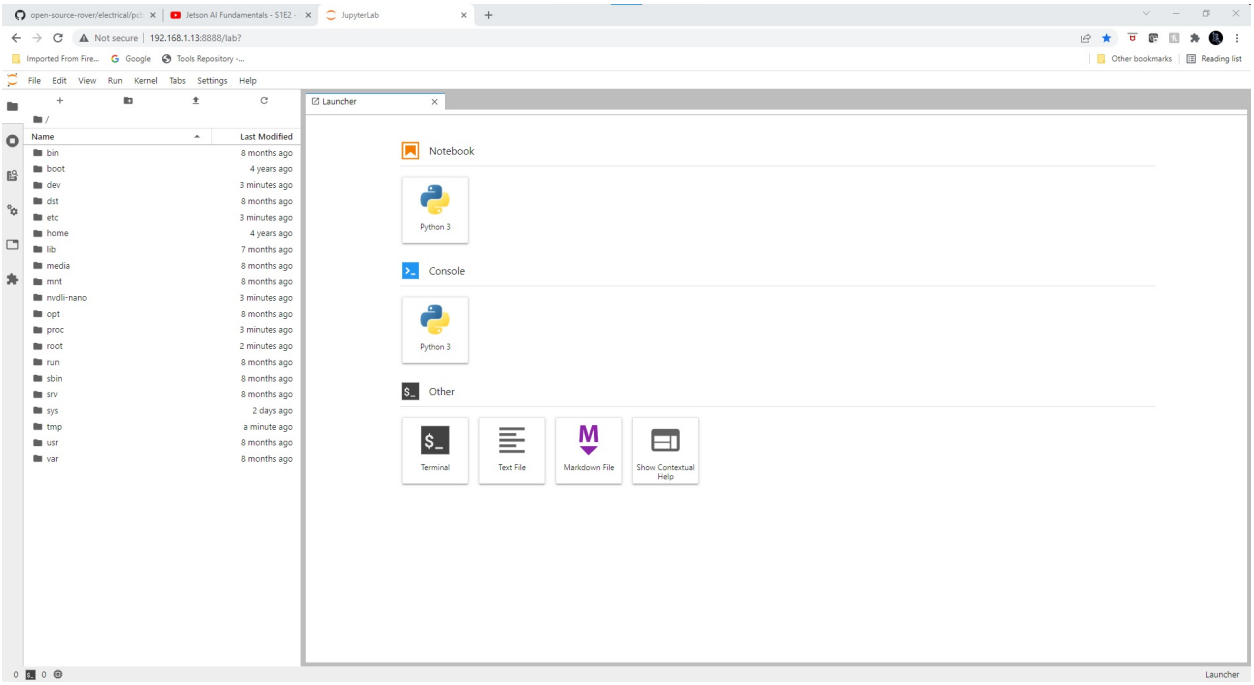
Status: Downloaded newer image for nvcr.io/nvidia/l4t-ml:r32.6.1-py3

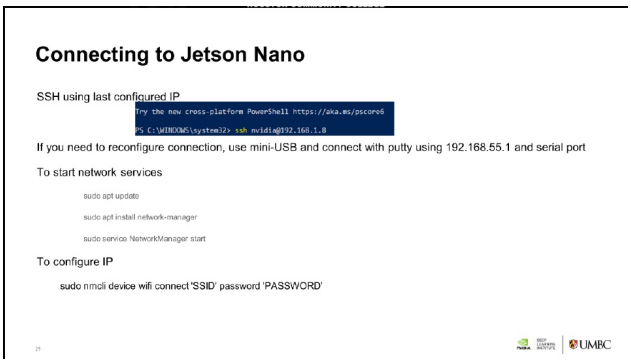
Now run the container (be sure your camera is attached):

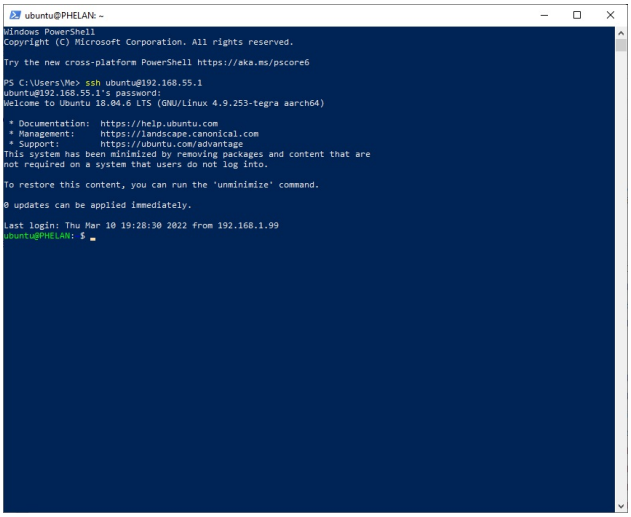
ubuntu@phelan:~$ **sudo docker run --runtime nvidia -it --rm --network host --volume ~/nvdli-data:/nvdli-nano/data --device /dev/video0 nvcr.io/nvidia/l4t-ml:r32.6.1-py3**

allow 10 sec for JupyterLab to start @ http://192.168.1.13:8888 (password nvidia)

JupterLab logging location: /var/log/jupyter.log (inside the container)







change **nvidia**@192.168.55.1

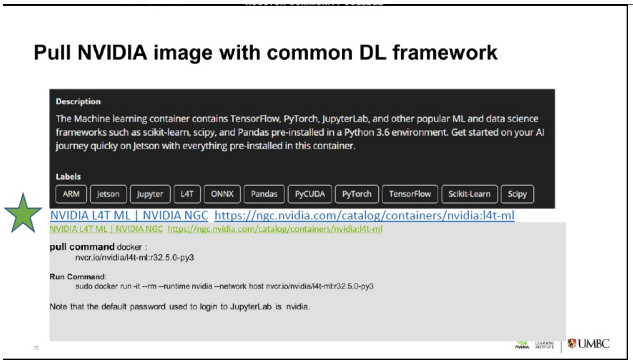
to **ubuntu**@192.168.55.1

Check for disk space

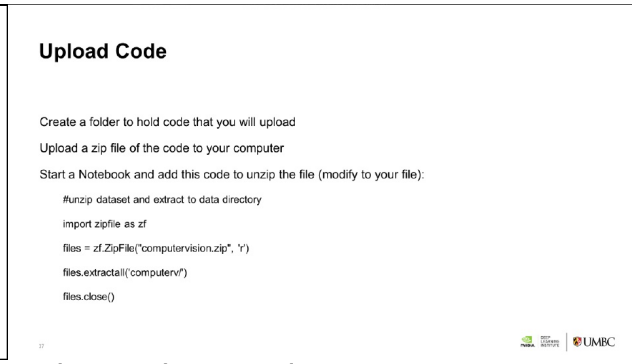
```
ubuntu@PHELAN:~$ df
Filesystem     1K-blocks     Used Available Use% Mounted on
/dev/mmcblk0p1 122762268 22305868  95312224  19% /
none             1780168        0   1780168   0% /dev
tmpfs            2025552       40   2025512   1% /dev/shm
tmpfs            2025552    29092   1996460   2% /run
tmpfs               5120        4      5116   1% /run/lock
tmpfs            2025552        0   2025552   0% /sys/fs/cgroup
tmpfs             405108      128    404980   1% /run/user/1000
```

Check for container images

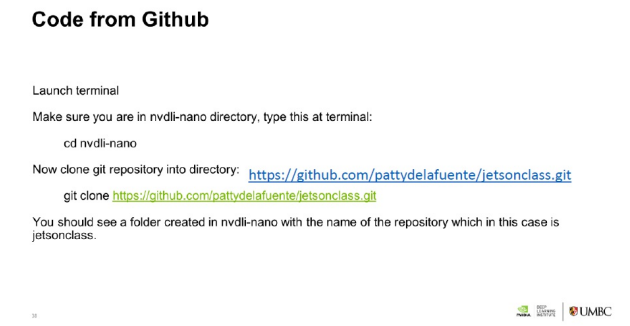
```
ubuntu@PHELAN:~$ docker image ls
Got permission denied while trying to connect to the Docker daemon socket at
unix:///var/run/docker.sock: Get
http://%2Fvar%2Frun%2Fdocker.sock/v1.24/images/json: dial unix
/var/run/docker.sock: connect: permission denied
ubuntu@PHELAN:~$ sudo docker image ls
[sudo] password for ubuntu:
REPOSITORY              TAG           IMAGE ID       CREATED        SIZE
nvcr.io/nvidia/l4t-ml   r32.6.1-py3   4818848f7fee   7 months ago   5GB
```



Already did this avove



Code? What code??



ubuntu@PHELAN:~$ **ls**

Desktop Documents Downloads examples.desktop Music **nvdli-data** Pictures Public Templates Videos

Do you mean nvdli-data?

Or create nvdli-nano *inside* nvdli-data?

Ah! Looking INSIDE Jupyter notebook I see a folder **nvdli-nano**.

Inside nvdli-nano is folder **data**.

/nvdli-nano/data folder is empty so far.

Using the docker launch terminal window:

root@PHELAN:/# ls

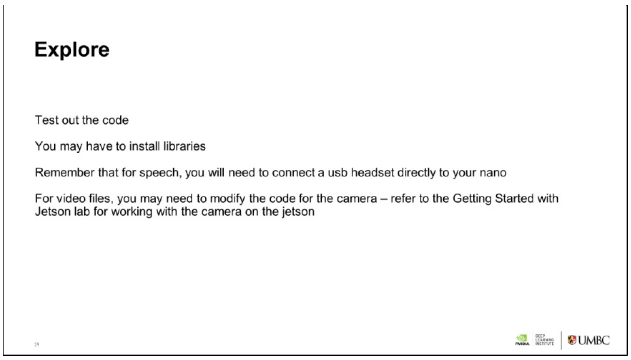
bin boot dev dst etc home lib media mnt **nvdli-nano** opt proc root run sbin srv sys tmp usr var

root@PHELAN:/# **cd nvdli-nano/**

root@PHELAN:/nvdli-nano# **git clone https://github.com/pattydelafuente/jetsonclass.git**

```
Cloning into 'jetsonclass'...
remote: Enumerating objects: 216, done.
remote: Counting objects: 100% (216/216), done.
remote: Compressing objects: 100% (209/209), done.
remote: Total 216 (delta 66), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (216/216), 48.66 MiB | 9.49 MiB/s, done.
Resolving deltas: 100% (66/66), done.
```

```
root@PHELAN:/nvdli-nano# ls
data  jetsonclass
```



USB headset?! What?!

First a Bluetooth speaker,

now a USB headset??

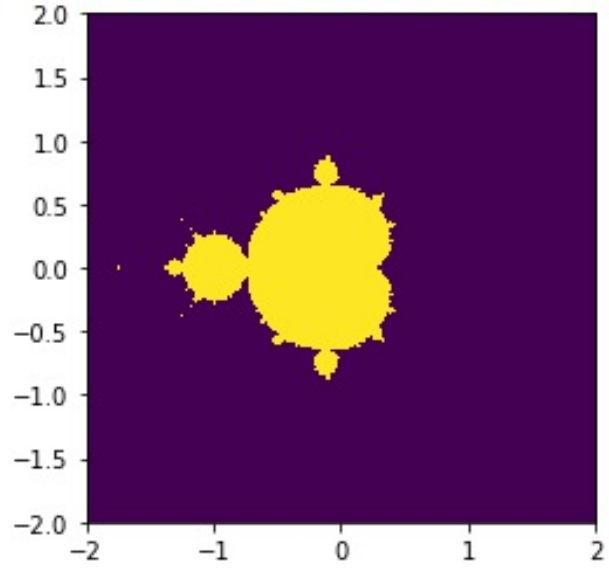
BestBuy / Amazon here I come..., again!

https://www.amazon.com/Logitech-Headset-H340-Stereo-Windows/dp/B008X3JGSI/ref=sr_1_1_sspa?crid=3RZ93CHDIN3CE&keywords=usb+headset+with+microphone+for+pc&qid=1646965469&refinements=p_85%3A2470955011&rnid=2470954011&rps=1&sprefix=usb+headset%2Caps%2C122&sr=8-1-spons&psc=1&spLa=ZW5jcnlwdGVkUXVhbGlmaWVyPUEyQ1JQMkpWNk5RTDc2JmVuY3J5cHRlZElkPUEwOTY3Mjg5MzE2S1JPUzQwSDROUiZlbmNyeXB0ZWRBZElkPUEwMjQ4NjY4MVdNUU5HT09aNzVMSCZ3aWRnZXROYW1lPXNwX2F0ZiZhY3Rpb249Y2xpY2tSZWRpcmVjdCZkb05vdExvZ0NsaWNrPXRydWU=#

Lab 1.4 AI Hardware and GPUs

Mandelbrot CPU

```
It took 118.02930188179016 seconds to calculate the Mandelbrot graph.
It took 0.37064123153686523 seconds to dump the image.
```



This notebook shows you how to generate Mandelbrot using the CPU and in the second code blocks uses PyCuda to generate a Mandelbrot using the GPU.

2022.03.12

Following from:

ITAI 2374 Robot OS & Platforms 04 B Spr 22 Final.pdf p.27/45

Windows PowerShell

[change screen background to black or you can't see the blue fonts]

PS C:\Users\Me> **ssh ubuntu@192.168.55.1**

```
ubuntu@192.168.55.1's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 4.9.253-tegra aarch64)
 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
This system has been minimized by removing packages and content that are
not required on a system that users do not log into.
To restore this content, you can run the 'unminimize' command.
0 updates can be applied immediately.
```

Setting Up wifi *Already done at initial boot*

Run docker command

This is set up for the USB camera, not the CSI camera.

See:
https://catalog.ngc.nvidia.com/orgs/nvidia/teams/dli/containers/dli-nano-ai

For CSI camera run:

```
sudo docker run --runtime nvidia -it --rm --network host \
 --volume ~/nvdli-data:/nvdli-nano/data \
 --volume /tmp/argus_socket:/tmp/argus_socket \
 --device /dev/video0 \
```

[leave off this '\' and put following on same command line]

```
 nvcr.io/nvidia/dli/dli-nano-ai:v2.0.1-r32.6.1
docker: invalid reference format.
See 'docker run --help'.
```

**sudo docker run --runtime nvidia -it --rm --network host **

**--volume ~/nvdli-data:/nvdli-nano/data **

**--volume /tmp/argus_socket:/tmp/argus_socket **

--device /dev/video0 *[following on same line]*

nvcr.io/nvidia/dli/dli-nano-ai:v2.0.1-r32.6.1

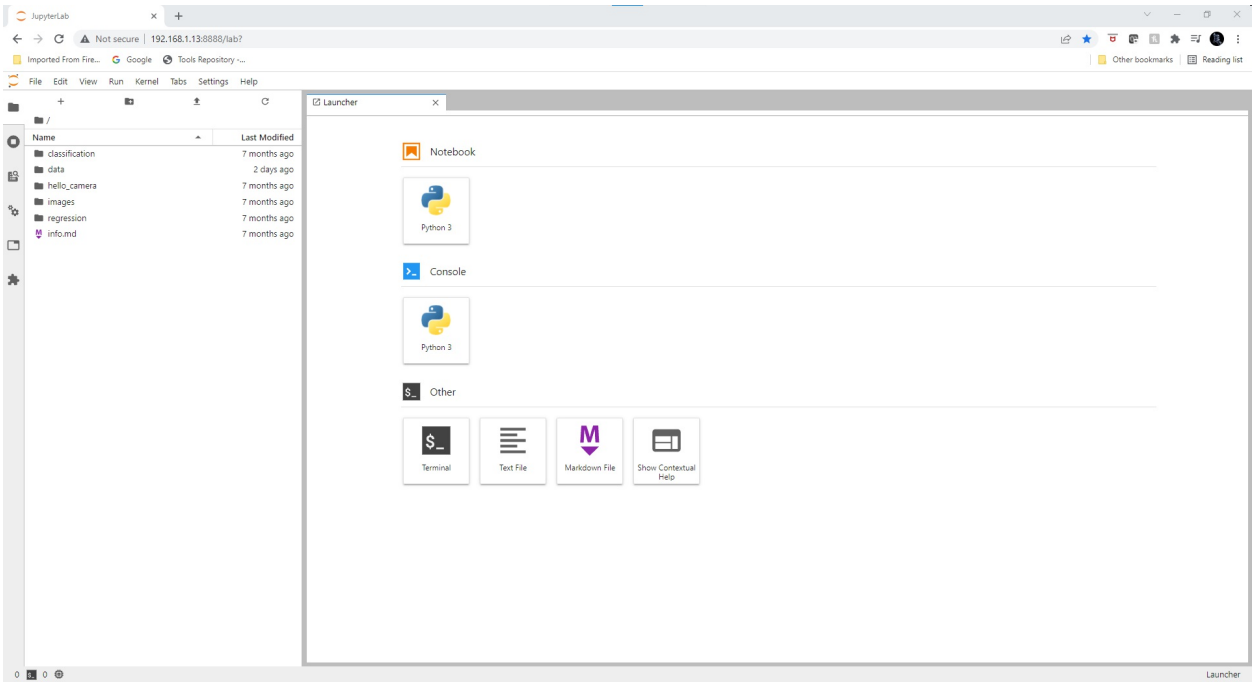
Long docker file install...

Digest: sha256:7d77446f42a89a1d9885c886ffc2a67e557c5c08665f3e4ee02e9f2d6d0d799b
220.6MB/355.6MB

Status: Downloaded newer image for nvcr.io/nvidia/dli/dli-nano-ai:v2.0.1-r32.6.1 219.5MB/355.6MB

allow 10 sec for JupyterLab to start @ **http://192.168.1.13:8888** (password **dlinano**)

JupterLab logging location: /var/log/jupyter.log (inside the container)



Check for Disk Space ***[Why are we doing this now?]***

NOTE This is in a regular PowerShell or PuTTY terminal, NOT in the Docker.

ubuntu@PHELAN:~$ **df**

```
Filesystem     1K-blocks     Used Available Use% Mounted on
/dev/mmcblk0p1 122762268 25369424  92248668  22% /
none             1780168        0   1780168   0% /dev
tmpfs            2025552       40   2025512   1% /dev/shm
tmpfs            2025552    38528   1987024   2% /run
tmpfs               5120        4      5116   1% /run/lock
tmpfs            2025552        0   2025552   0% /sys/fs/cgroup
tmpfs             405108      136    404972   1% /run/user/1000
/dev/loop0         16334       66     16268   1% /media/ubuntu/L4T-README
```

Shows a 128GB SD card in use

Check for container images [which window do we run this in?]

ubuntu@PHELAN:~$ **docker image ls**

```
Got permission denied while trying to connect to the Docker
daemon socket at unix:///var/run/docker.sock: Get
http://%2Fvar%2Frun%2Fdocker.sock/v1.24/images/json: dial unix
/var/run/docker.sock: connect: permission denied
```

ubuntu@PHELAN:~$ **sudo docker image ls**

```
[sudo] password for ubuntu:
REPOSITORY                          TAG              IMAGE ID       CREATED
SIZE
nvcr.io/nvidia/dli/dli-nano-ai      v2.0.1-r32.6.1   a4e89dcd35e9   7 months ago
3.38GB
nvcr.io/nvidia/l4t-ml               r32.6.1-py3      4818848f7fee   7 months ago
5GB
```

Pull NVIDIA image with common DL framework

pull command docker: *[again, which window??]*

nvcr.io/nvidia/l4t-ml:r32.5.0-py3

ubuntu@PHELAN:~$ **nvcr.io/nvidia/l4t-ml:r32.5.0-py3**

-bash: nvcr.io/nvidia/l4t-ml:r32.5.0-py3: No such file or directory

root@PHELAN:/nvdli-nano# **nvcr.io/nvidia/l4t-ml:r32.5.0-py3**

bash: nvcr.io/nvidia/l4t-ml:r32.5.0-py3: No such file or directory

Run Command

[didn't we already run this above, just a little different?]

sudo docker run -it --rm --runtime nvidia --network host nvcr.io/nvidia/l4t-ml:r32.5.0-py3

ubuntu@PHELAN:~$ **sudo docker run -it --rm --runtime nvidia --network host nvcr.io/nvidia/l4t-ml:r32.5.0-py3**

allow 10 sec for JupyterLab to start @ **http://localhost:8888** (password **nvidia**)

localhost only work on the Nano. For the PC use:

http://192.168.1.13:8888

Jupyter notebook appears with root directory file tree.

Code from Github

Launch terminal

Make sure you are in nvdli-nano directory

There is NO **nvdli-nano** directory:

ubuntu@PHELAN:~$ **ls**

Desktop Documents Downloads examples.desktop Music **nvdli-data** Pictures Public Templates Videos

ubuntu@PHELAN:~$ **cd nvdli-data/**

ubuntu@PHELAN:~/nvdli-data$ **ls**

[empty]

root@PHELAN:/# **ls**

bin boot dev dst etc home lib media mnt opt proc root run sbin srv sys tmp usr var

ubuntu@PHELAN:~$ **sudo docker image ls**

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
nvcr.io/nvidia/dli/dli-nano-ai	v2.0.1-r32.6.1	a4e89dcd35e9	7 months ago	3.38GB
nvcr.io/nvidia/l4t-ml	r32.**6.1**-py3	4818848f7fee	7 months ago	5GB
nvcr.io/nvidia/l4t-ml	r32.**5.0**-py3	b2944895855d	14 months ago	3.98GB

Try 6.1...

Nope, same issue

SIDEBAR: configure Nano for VNC per desktop readme. **Yes!!**

Give up on class instructions and continue with NVIDIA course:

https://courses.nvidia.com/courses/course-v1:DLI+S-RX-02+V2/courseware/b2e02e999d9247eb8e33e893ca052206/63a4dee75f2e4624afbc33bce7811a9b/?activate_block_id=block-v1%3ADLI%2BS-RX-02%2BV2%2Btype%40sequential%2Bblock%4063a4dee75f2e4624afbc33bce7811a9b

Download Docker And Start JupyterLab

PuTTY **ubuntu@192.168.1.13**

directory is already made: ubuntu@PHELAN:~/nvdli-data$

**sudo docker run --runtime nvidia -it --rm --network host **

**--volume ~/nvdli-data:/nvdli-nano/data **

**--volume /tmp/argus_socket:/tmp/argus_socket **

--device /dev/video0
nvcr.io/nvidia/dli/dli-nano-ai:v2.0.1-r32.6.1

Already loaded docker so no long download --

allow 10 sec for JupyterLab to start @ **http://192.168.1.13:8888** (password **dlinano**)

ubuntu@PHELAN:~$ **echo "sudo docker run --runtime nvidia -it --rm --network host **

**> --volume ~/nvdli-data:/nvdli-nano/data **

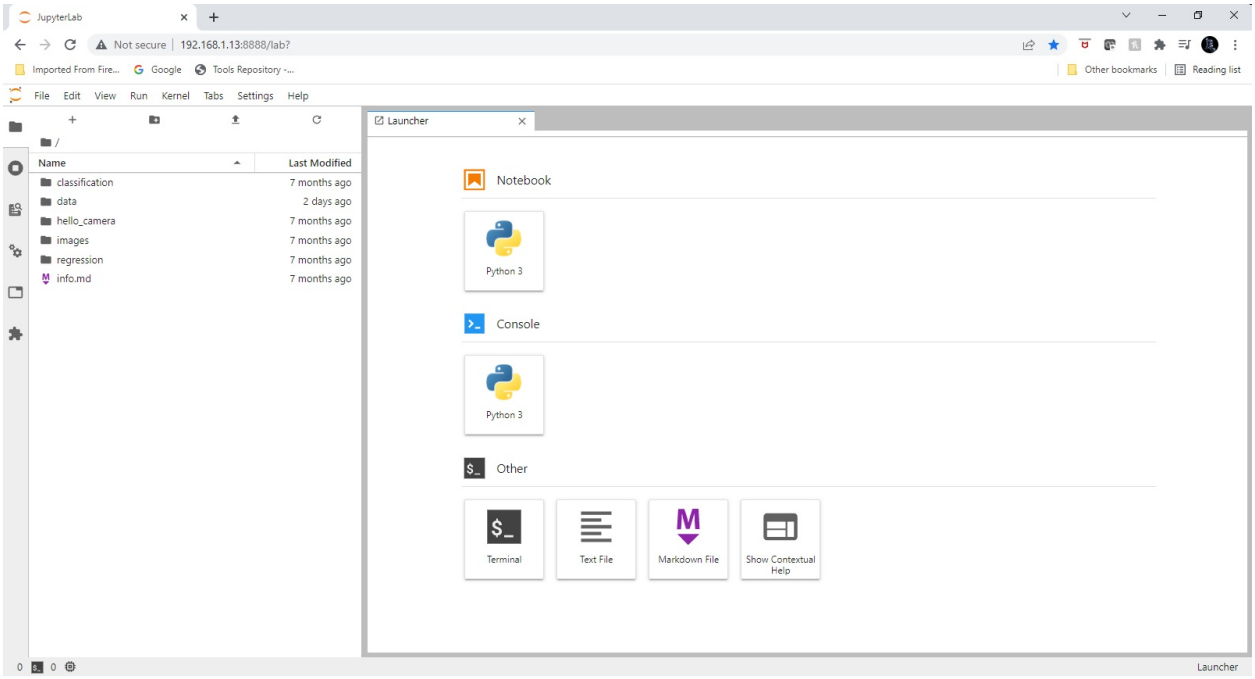
**> --volume /tmp/argus_socket:/tmp/argus_socket **

> --device /dev/video0
nvcr.io/nvidia/dli/dli-nano-ai:v2.0.1-r32.6.1" >
docker_dli_run.sh

ubuntu@PHELAN:~$ **chmod +x docker_dli_run.sh**

ubuntu@PHELAN:~$ **./docker_dli_run.sh**

allow 10 sec for JupyterLab to start @ **http://192.168.1.13:8888** (password **dlinano**)



2022.03.13

hello_camera

csi_camera.ipynb

successfully completed notebook

Correct pink tint of CSI camera images. Wandering Google search, dead ends, then finally:

https://www.waveshare.com/wiki/IMX219-160_Camera

If you find that the image captured is reddish. You can try to download .isp file and installed:

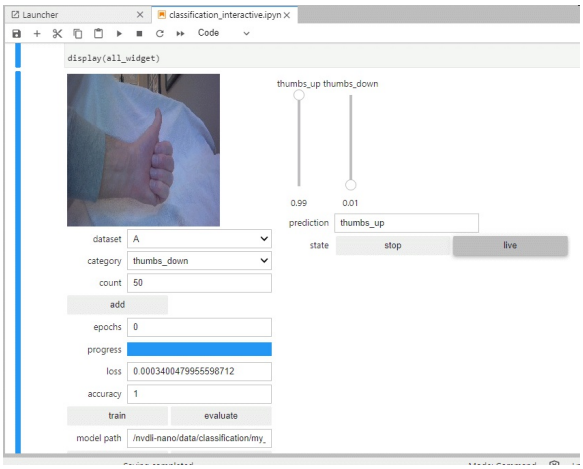
```
sudo wget https://www.waveshare.com/w/upload/e/eb/Camera_overrides.tar.gz
sudo tar zxvf Camera_overrides.tar.gz
sudo cp camera_overrides.isp /var/nvidia/nvcam/settings/
sudo chmod 664 /var/nvidia/nvcam/settings/camera_overrides.isp
sudo chown root:root /var/nvidia/nvcam/settings/camera_overrides.isp
```

Color seems a bit more natural now?

2022.03.13

Classification: Thumbs Up / Thumbs Down

Follow notebook per: https://courses.nvidia.com/courses/course-v1:DLI+S-RX-02+V2/courseware/b2e02e999d9247eb8e33e893ca052206/26aa9f8bdaa948d9b068a8275c89e546/?child=first



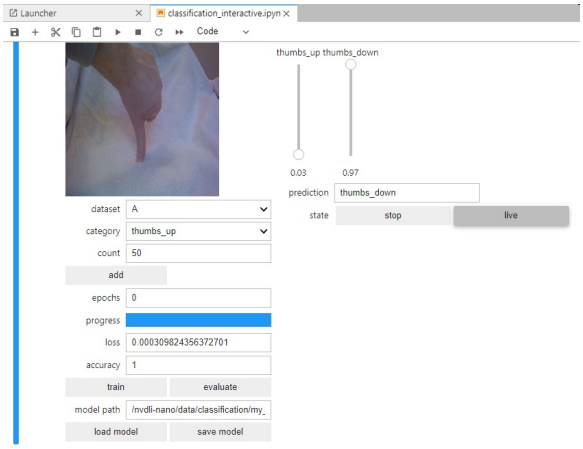
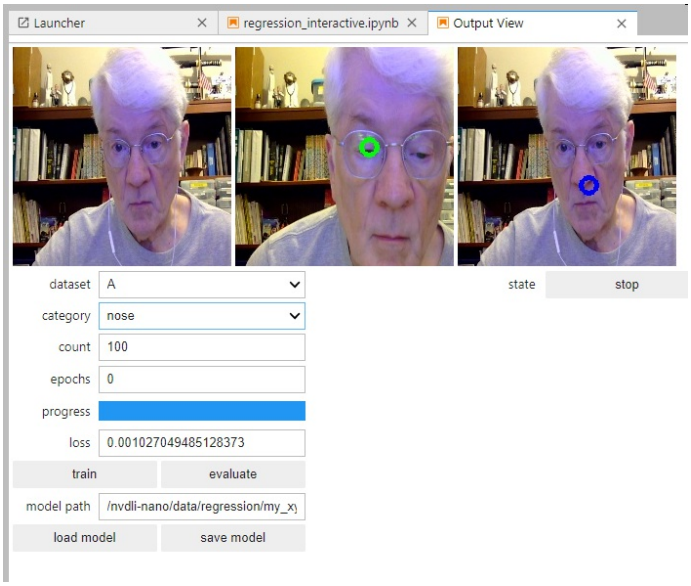


Image Regression Project /regression/regression_interactive.ipynb



This looks like it would work better with the USB camera which is easier to place in front of my face near the monitor than the CSI camera on the Jetbot. Have to shutdown & restart to do this....

Accuracy isn't particularyly good even w/ 100 images each and 3 trainings w/ 10 epochs each.

Finished “Getting Started with AI on Jetson Nano”.

Next stop is “Hello AI World” at

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

Which takes you by clicking on any of the 3 pictures to

https://github.com/dusty-nv/jetson-inference

then to Running the Docker Container at

https://github.com/dusty-nv/jetson-inference/blob/master/docs/aux-docker.md

Launching the Container

BEFORE

ubuntu@PHELAN:~$ **ls**

```
Desktop           Documents  examples.desktop  nvdli-data  Public     Videos
docker_dli_run.sh  Downloads  Music            Pictures    Templates
```

Due to various mounts and devices needed to run the container, it's recommended to use the docker/run.sh script to run the container:

ubuntu@PHELAN:~$ **git clone --recursive https://github.com/dusty-nv/jetson-inference**

```
Cloning into 'jetson-inference'...
```

AFTER

ubuntu@PHELAN:~$ **ls**

```
Desktop           Documents  examples.desktop  Music       Pictures
Templates
docker_dli_run.sh  Downloads  jetson-inference  nvdli-data  Public    Videos
```

ubuntu@PHELAN:~$ **cd jetson-inference/**

ubuntu@PHELAN:~/jetson-inference$ **ls**

```
c  calibration  CHANGELOG.md  CMakeLists.txt  CMakePreBuild.sh  data  docker
Dockerfile  docs  examples  LICENSE.md  plugins  python  README.md  tools
utils
```

ubuntu@PHELAN:~/jetson-inference$ **cd docker**

ubuntu@PHELAN:~/jetson-inference/docker$ **ls**

```
build.sh  pull.sh  push.sh  run.sh  tag.sh
```

ubuntu@PHELAN:~/jetson-inference$ **docker/run.sh**

reading L4T version from /etc/nv_tegra_release

L4T BSP Version: L4T R32.7.1

[sudo] password for ubuntu:

download...

Pop-up menu - accept default selections

more downloads...

docker downloads & extractions...

Classifying Images with ImageNet # (default network is googlenet)

root@PHELAN:/jetson-inference# **cd jetson-inference/build/aarch64/bin**

For C++

root@PHELAN:/jetson-inference/build/aarch64/bin# **./imagenet images/orange_0.jpg images/test/output_0.jpg**

*a **lot** of [TRT] statements fly by optimizing the network...*

For Python

change output_0 to output_1

root@PHELAN:/jetson-inference/build/aarch64/bin# **./imagenet.py images/orange_0.jpg images/test/output_1.jpg**

They’re both confident it’s an orange:

root@PHELAN:/jetson-inference/build/aarch64/bin# **./imagenet images/strawberry_0.jpg images/test/output_strawberry_cpp.jpg**

root@PHELAN:/jetson-inference/build/aarch64/bin# **./imagenet.py images/strawberry_0.jpg images/test/output_strawberry_py.jpg**

C++

Python

Downloading Other Classification Models

(default network is googlenet)

C++

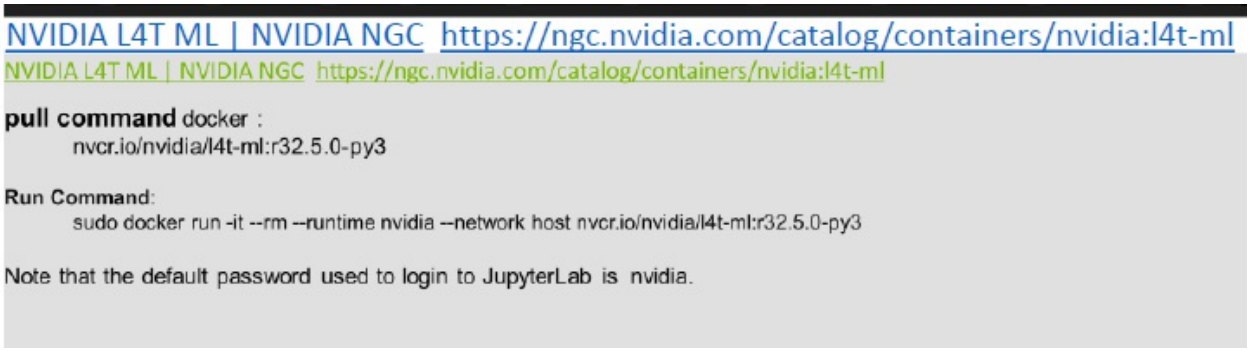
$ ./imagenet **--network=resnet-18** images/jellyfish.jpg images/test/output_jellyfish.jpg

Network	CLI argument	NetworkType enum
AlexNet	`alexnet`	`ALEXNET`
GoogleNet	`googlenet`	`GOOGLENET`
GoogleNet-12	`googlenet-12`	`GOOGLENET_12`
ResNet-18	`resnet-18`	`RESNET_18`
ResNet-50	`resnet-50`	`RESNET_50`
ResNet-101	`resnet-101`	`RESNET_101`
ResNet-152	`resnet-152`	`RESNET_152`
VGG-16	`vgg-16`	`VGG-16`
VGG-19	`vgg-19`	`VGG-19`
Inception-v4	`inception-v4`	`INCEPTION_V4`

2022.03.17

Revisiting the failed class lecture JN failure:

Lecture slide says:


NVIDIA L4T ML | NVIDIA NGC https://ngc.nvidia.com/catalog/containers/nvidia:l4t-ml

NVIDIA L4T ML | NVIDIA NGC https://ngc.nvidia.com/catalog/containers/nvidia:l4t-ml

pull command docker :
nvcr.io/nvidia/l4t-ml:r32.5.0-py3

Run Command:
sudo docker run -it --rm --runtime nvidia --network host nvcr.io/nvidia/l4t-ml:r32.5.0-py3

Note that the default password used to login to JupyterLab is nvidia.

"pull command docker: nvcr.io/nvidio/l4t-ml:**r32.5.0-py3**"

this should be:

sudo docker pull nvcr.io/nvidio/l4t-ml:**r32.5.0-py3**

Google: "**l4t-ml:r32.5.0-py3**" and you find:

https://catalog.ngc.nvidia.com/orgs/nvidia/containers/l4t-ml

which says:

JetPack **4.5** (**L4T R32.5.0**)

l4t-ml:r32.5.0-py3

Our current LT4 version is:

Package: nvidia-l4t-core

...

Maintainer: NVIDIA Corporation

Architecture: arm64

Version: **32.7.1**-20220219090432

From dusty-nv/jetson-inference:
https://github.com/dusty-nv/jetson-inference/blob/master/docs/aux-docker.md

Container Tag	L4T version	JetPack version
dustynv/jetson-inference:r32.7.1	L4T R32.7.1	JetPack 4.6.1
dustynv/jetson-inference:r32.6.1	L4T R32.6.1	JetPack 4.6
dustynv/jetson-inference:r32.5.0	L4T R32.5.0	JetPack 4.5
dustynv/jetson-inference:r32.4.4	L4T R32.4.4	JetPack 4.4.1
dustynv/jetson-inference:r32.4.3	L4T R32.4.3	JetPack 4.4

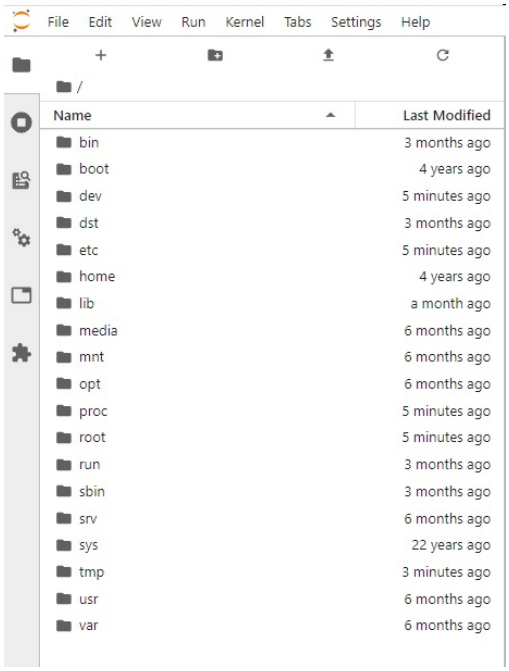
...that's -r32.7.1

So we *should* want:

sudo docker pull nvcr.io/nvidia/l4t-ml:r32.7.1-py3

...successful docker download, then

sudo docker run -it --rm --runtime nvidia --network host nvcr.io/nvidia/l4t-ml:r32.7.1-py3



Notebook STILL shows no nvidia-nano directory nor any .ipynb files.

root@PHELAN:/# **exit**

Though the instructions don't say to, maybe I need to create my own nvidia-nano directory?

some cleanup:

ubuntu@PHELAN:~$ **sudo docker image ls**

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
dustynv/jetson-inference	**r32.7.1**	**7a7d343029a2**	**3 weeks ago**	**2.83GB**
nvcr.io/nvidia/dli/dli-nano-ai	v2.0.1-r32.6.1	a4e89dcd35e9	7 months ago	3.38GB
nvcr.io/nvidia/l4t-ml	**r32.6.1-py3**	4818848f7fee	7 months ago	5GB
nvcr.io/nvidia/l4t-ml	r32.5.0-py3	b2944895855d	14 months ago	3.98GB

ubuntu@PHELAN:~$ **sudo docker image rm b2944895855d**

ubuntu@PHELAN:~$ sudo docker run -it --rm --runtime nvidia --network host nvcr.io/nvidia/l4t-ml:**r32.6.1**-py3

same problem

root@PHELAN:/# **exit**

Class lecture slide then says:

> Launch terminal
>
> Make sure you are in nvdli-nano directory, type this at terminal:
>
> cd nvdli-nano
>
> Now clone git repository into directory: https://github.com/pattydelafuente/jetsonclass.git
>
> git clone https://github.com/pattydelafuente/jetsonclass.git
>
> You should see a folder created in nvdli-nano with the name of the repository which in this case is jetsonclass.

Which says:

This repository contains code and instructions to support the Applied AI course using the Jetson Nano **2GB**.

I need 4GB *but maybe it will work anyway? Let's try...*

ubuntu@PHELAN:~$ **mkdir nvdli-nano**

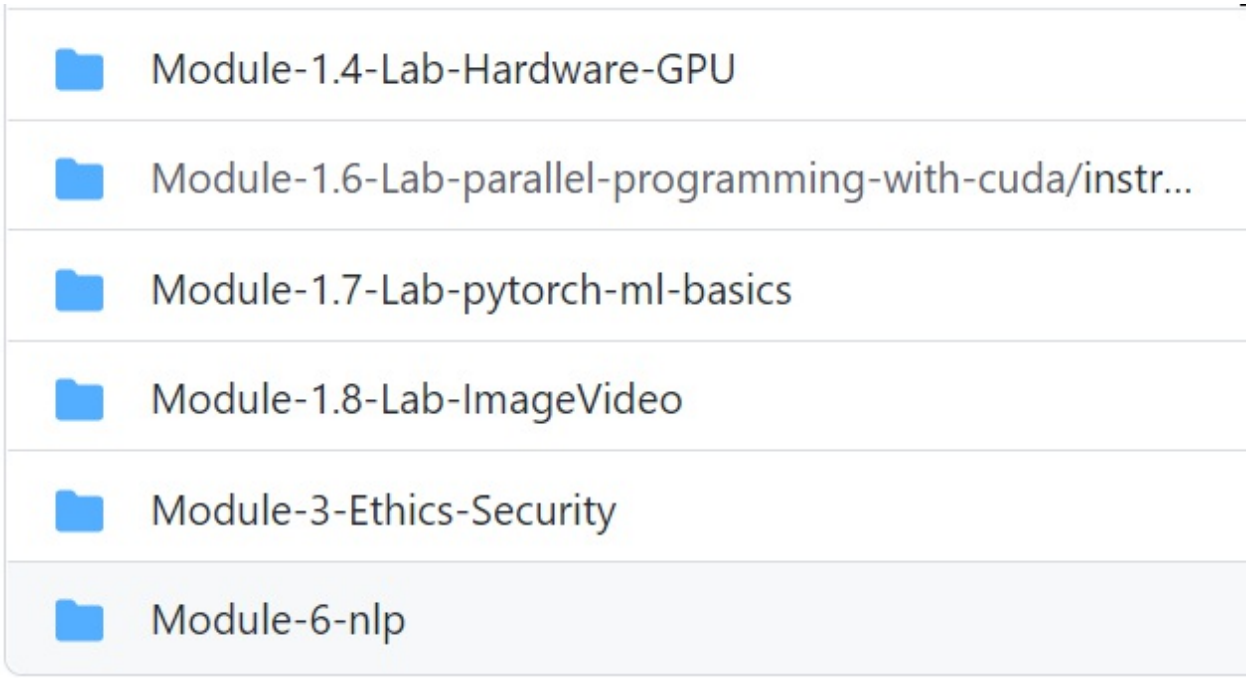
ubuntu@PHELAN:~$ **cd nvdli-nano**

ubuntu@PHELAN:~/nvdli-nano$ **git clone https://github.com/pattydelafuente/jetsonclass.git**

The README.md just says to pull an old lesson:

sudo docker pull nvcr.io/nvidia/l4t-ml:r32.6.1-py3

but the Labs folder has more stuff:



ubuntu@PHELAN:~/nvdli-nano$ **sudo docker pull nvcr.io/nvidia/l4t-ml:r32.6.1-py3**

Status: Image is up to date for nvcr.io/nvidia/l4t-ml:r32.6.1-py3

ubuntu@PHELAN:~/nvdli-nano$ **sudo docker run --runtime nvidia -it --rm --network host --volume ~/nvdli-data:/nvdli-nano/data --device /dev/video0 nvcr.io/nvidia/l4t-ml:r32.6.1-py3**

ALL ON ONE LINE

STILL RUNS NOTEBOOK WITH OLD STUFF!

The nvidia-nano I created is there, but only contains data/classification & regression from old labs!

2022.03.17 continued

Going back to

https://github.com/dusty-nv/jetson-inference/blob/master/docs/imagenet-console-2.md

and continuing after strawberries... Will leave out pix as online

Using Different Classification Models

run w/o downloading models (resnet-18 already downloaded):

C++

$ ./imagenet --network=**resnet-18** images/jellyfish.jpg images/test/output_jellyfish**01**.jpg

Python

$ ./imagenet.py --network=resnet-18 images/jellyfish.jpg images/test/output_jellyfish**02**.jpg

Processing a Video

ubuntu@PHELAN:~/jetson-inference$ **wget https://nvidia.box.com/shared/static/tlswont1jnyu3ix2tbf7utaekpzcx4rc.mkv -O jellyfish.mkv**

Dowmloaded at ubuntu@ so tricky to move and get command to point to the right directory for

jetson-inference/data/networks/jellyfish.mkv

but it worked with video being variably confident the swimming thing was a jellyfish.

Coding Your Own Image Recognition Program (Python)

./my-recognition.py my_image.jpg

“image is recognized as ‘West Highland white terrier’ (class #203) with 88.085938% confidence”

(confidence to 8 sig digits is ridiculous! Just call it 88%!)

Coding Your Own Image Recognition Program (C++)

skip for now

Running the Live Camera Recognition Demo

https://github.com/dusty-nv/jetson-inference/raw/dev/docs/images/detectnet.jpg

Still recognized a Westie, but was really bad about thinking earphones were a stethescope! Didn't recognize person at all!

Locating Objects with DetectNet

./detectnet --network=ssd-mobilenet-v2 images/peds_0.jpg images/test/output.jpg

worked reasonably well but failed to recognize some.

Failed to label the banana in the output file.

SYSTEM UPGRADE ISSUES

Software Updated notified that upgrade was available.

Accepted as prev advised by D'.

After that the **docker/build.sh refused to run for v32.7.1**

Looked at build.sh:

```
if [ -z $BASE_IMAGE ]; then
        if [ $L4T_VERSION = "32.6.1" ]; then
BASE_IMAGE="nvcr.io/nvidia/l4t-pytorch:r32.6.1-pth1.9-py3"
        elif [ $L4T_VERSION = "32.5.1" ]; then
BASE_IMAGE="nvcr.io/nvidia/l4t-pytorch:r32.5.0-pth1.6-py3"
        elif [ $L4T_VERSION = "32.5.0" ]; then
BASE_IMAGE="nvcr.io/nvidia/l4t-pytorch:r32.5.0-pth1.6-py3"
```

No provision for 32.6.1

Googled "lt4-pytorch:r32.7.1-pth" and found

https://catalog.ngc.nvidia.com/orgs/nvidia/containers/l4t-pytorch/collections

where it indicates

```
Latest Tag  r32.7.1-pth1.10-py3
Modified    March 14, 2022
```

So modified build.sh to build-PHELAN.sh:

```
if [ -z $BASE_IMAGE ]; then
        if [ $L4T_VERSION = "32.7.1" ]; then
BASE_IMAGE="nvcr.io/nvidia/l4t-pytorch:r32.7.1-pth1.10-py3"
        elif [ $L4T_VERSION = "32.6.1" ]; then
BASE_IMAGE="nvcr.io/nvidia/l4t-pytorch:r32.6.1-pth1.9-py3"
        elif [ $L4T_VERSION = "32.5.1" ]; then
BASE_IMAGE="nvcr.io/nvidia/l4t-pytorch:r32.5.0-pth1.6-py3"
```

Then ran

$ **docker/build-PHELAN**

it downloaded and ran a **lot** of stuff!

Still running at bedtime. Hope I didn't trash the system so I have to start over! BEWARE UPGRADES!

2022.03.19

Terminal window is back to ubuntu@PHELAN:~/jetson-inference$

so...

ubuntu@PHELAN:~/jetson-inference$ **docker/run.sh**

```
<pre>reading L4T version from /etc/nv_tegra_release

L4T BSP Version:  L4T R32.7.1

[sudo] password for ubuntu:

size of data/networks:  745322914 bytes

Downloading pytorch-ssd base model...

python/training/detection/ssd/model
100%[=====================================================================&gt;]  36.23M   525KB/s
in 68s

CONTAINER:     dustynv/jetson-inference:r32.7.1

DATA_VOLUME:   --volume /home/ubuntu/jetson-inference/data:/jetson-inference/data --volume
/home/ubuntu/jetson-inference/python/training/classification/data:/jetson-inference/python/traini
ng/classification/data --volume
/home/ubuntu/jetson-inference/python/training/classification/models:/jetson-inference/python/trai
ning/classification/models --volume
/home/ubuntu/jetson-inference/python/training/detection/ssd/data:/jetson-inference/python/trainin
g/detection/ssd/data --volume
/home/ubuntu/jetson-inference/python/training/detection/ssd/models:/jetson-inference/python/train
ing/detection/ssd/models

USER_VOLUME:

USER_COMMAND:

V4L2_DEVICES:    --device /dev/video0

localuser:root being added to access control list</pre>
```

Processing a Directory or Sequence of Images

modified to

./detectnet "images/fruit_*.jpg" images/test/fruit_output_%i.jpg

again failed to label the output files. May have to specify in the command --overlay=box,labels,conf

...did a miserable job. Most of the images were not labeled at all. Many of those that had labels were wrong: bunch of grapes as potted plant, pineapple as bird, plate of cake surrounded by strawberries only recognized the cake, pan of oranges as bowl, melon as bird. Still, like the dancing bear....[“it’s not that it dances well, but that it dances at all”]

Let’s specify the network: --network=ssd-mobilenet-v2 *same*

Specify --network=resnet-18 *fails to run*

Semantic Segmentation with SegNet

[spoiler - see success after next section]

./segnet --network=fcn-resnet18-cityscapes images/city_0.jpg images/test/output.jpg fails:

[TRT] model format 'custom' not supported by jetson-inference

[TRT] segNet -- failed to load.

segnet: failed to initialize segNet

note: to download additional networks, run the Model Downloader tool:

cd jetson-inference/tools

./download-models.sh

Pop-up menu:

[*] 1 Image Recognition - all models (2.2 GB)

same error

Google “model format 'custom' not supported by jetson-inference”

https://forums.developer.nvidia.com/t/custom-trained-model-detectnet-jetson-inference/172940

Solved by dusty_nv in post #4

Hi @nitinkumar96, try it like this instead: net = jetson.inference.detectNet(argv=['--model=models/fruit/ssd-mobilenet.onnx', '--labels=models/fruit/labels.txt', '--input-blob=input_0', '--output-cvg=scores', '--output-bbox=boxes'])

This applies to the original python code. Maybe I can fix?

The only similar line in segnet.py is

```
net = jetson.inference.detectNet(opt.network, sys.argv, opt.threshold)
```

the statements above and below use

```
argv=sys.argv
```

Let's try that

Nope, didn't fix it.

Give up on this one

DeepScene

works ok but not in batch mode [didn't do it right. Omitted "'s]

Try segnet again

./segnet --network=fcn-resnet18-cityscapes images/city_0.jpg images/test/output.jpg

LONG output of [TRT]...

This time a colored fuzzy street scene.

Multi-Human Parsing (MHP)

root@PHELAN:/jetson-inference/build/aarch64/bin# **./segnet --network=fcn-resnet18-mhp images/humans_0.jpg images/test/output_humans_0.jpg**

Fuzzy colored overlay of head, trunk, hips, lower legs.

Pascal VOC

root@PHELAN:/jetson-inference/build/aarch64/bin# **./segnet --network=fcn-resnet18-voc images/object_0.jpg images/test/output_obj_0.jpg**

first run is long [TRT] process...

Then more fuzzy overlays

SUN RGB-D

Running the Live Camera Segmentation Demo

Could be useful for robot navicatio: "drive to the green"

root@PHELAN:/jetson-inference/build/aarch64/bin# **./segnet --network=fcn-resnet18-cityscapes /dev/video0**

tried this out the window, but not a very good view. Too much clutter. Would be worth trying outside w/ 160o CSI camera on teleoperated JetBot!

Pose Estimation with PoseNet

Good stick figures w/ pix. Fair w/ USB camera but narrow field of view. Want to try w/ CSI camera 160o FoV and Tai Chi & portable HDMI screen.

./posenet csi://0

./posenet --network=densenet121-body csi://0

LOT of [TRT] preparation...

./posenet --network=densenet121-body csi://0 tai-chi.mp4 or

./posenet --network=densenet121-body csi://0 > tai-chi.mp4

don't work as camera input is open-ended so .mp4 can't be opened

Is there a way to save just the stick figures as a .mp4?

2022.03.20

Monocular Depth with DepthNet

Very interesting how well you can judge depth from a mono perspective, yet sometimes misjudges.

Transfer Learning with PyTorch

Verifying PyTorch

ubuntu@PHELAN:~/jetson-inference$ **./docker/run.sh**

root@PHELAN:/jetson-inference# **cd build**

root@PHELAN:/jetson-inference/build# **./install-pytorch.sh**

```
bash: ./install-pytorch.sh: No such file or directory
```

wrong directory, it's really

root@PHELAN:/jetson-inference# **./tools/install-pytorch.sh**

Nothing installed

root@PHELAN:/jetson-inference# python3

```
Python 3.6.9 (default, Dec  8 2021, 21:08:43)
>>> import torch
>>> print(torch.__version__)
1.10.0
>>> print('CUDA available: ' + str(torch.cuda.is_available()))
CUDA available: True
>>> a = torch.cuda.FloatTensor(2).zero_()
>>> print('Tensor a = ' + str(a))
Tensor a = tensor([0., 0.], device='cuda:0')
>>> b = torch.randn(2).cuda()
Tensor b = tensor([ 2.4270, -0.6641], device='cuda:0')
>>> c = a + b
>>> print('Tensor c = ' + str(c))
Tensor c = tensor([ 2.4270, -0.6641], device='cuda:0')
>>> import torchvision
>>> print(torchvision.__version__)
0.11.0a0+fa347eb
```

Note that the torch version should be reported as 1.6.0 [1.10.1] and the torchvision version should be 0.7.0. [0.11.0]

Type	Dataset	Size	Classes	Training Images	Time per Epoch*	Training Time**
Classification	Cat/Dog	800MB	2	5,000	~7-8 minutes	~4 hours
Classification	PlantCLEF	1.5GB	20	10,475	~15 minutes	~8 hours
Detection	Fruit	2GB	8	6,375	~15 minutes	~8 hours

No. We're not going there today!

Going back to last class lecture slide:

https://github.com/pattydelafuente/jetsonclass

Pull the latest deep learning docker container:

sudo docker pull nvcr.io/nvidia/l4t-ml:r32.6.1-py3

*this needs to be **7.1** but will that change alone work??*

ubuntu@PHELAN:~$ **sudo docker pull nvcr.io/nvidia/l4t-ml:r32.7.1-py3**

```
[sudo] password for ubuntu:
r32.7.1-py3: Pulling from nvidia/l4t-ml
Digest: sha256:238be76a09a10904b6b9d3cfca5cd2b34c369a3c3c99340050f825998f552a5b
Status: Image is up to date for nvcr.io/nvidia/l4t-ml:r32.7.1-py3
nvcr.io/nvidia/l4t-ml:r32.7.1-py3
```

ubuntu@PHELAN:~$ **sudo docker image ls**

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
<none>	<none>	56a2754ec1e2	37 hours ago	2.66GB
dustynv/jetson-inference	r32.7.1	7a7d343029a2	4 weeks ago	2.83GB
nvcr.io/nvidia/l4t-ml	**r32.7.1-py3**	**bc7e6d602c66**	**4 weeks ago**	**5.67GB**
nvcr.io/nvidia/l4t-pytorch	r32.7.1-pth1.10-py3	c8887f7ca152	5 weeks ago	1.96GB
nvcr.io/nvidia/dli/dli-nano-ai	v2.0.1-r32.6.1	a4e89dcd35e9	7 months ago	3.38GB
nvcr.io/nvidia/l4t-ml	r32.6.1-py3	4818848f7fee	7 months ago	5GB

Run the container by Executing the following:

```
sudo docker run --runtime nvidia -it --rm --network host
--volume ~/nvdli-data:/nvdli-nano/data
--device /dev/video0
nvcr.io/nvidia/l4t-ml:r32.6.1-py3
```

Need to add for CSI camera:

```
--volume /tmp/argus_socket:/tmp/argus_socket
```

and change repository:tab to

```
nvcr.io/nvidia/l4t-ml:r32.7.1-py3
```

and ALL ON ONE LINE!

```
sudo docker run --runtime nvidia -it --rm --network host --volume
~/nvdli-data:/nvdli-nano/data --volume /tmp/argus_socket:/tmp/argus_socket
--device /dev/video0 nvcr.io/nvidia/l4t-ml:r32.7.1-py3
```

...JupyterLab to start @ http://192.168.1.15:8888 (password **nvidia**)

NO NOTEBOOKS (same story)

Doing the same thing over and over again and expecting a different result is NOT the definition of insanity. Loss of reality testing is. But it IS a pretty good definition of STUPIDITY! I give up pending upload of new files from R&D.

2022.03.24

Following from Module05 class slides to

https://github.com/dusty-nv/jetson-inference#hello-ai-world

and

https://www.youtube.com/watch?v=QXIwdsyK7Rw

2022.03.26

Tried MANY avenues to get CSI camera to flip but nothing worked.

video-viewer csi://0 --input-flip=rotate-180 *or just*

video-viewer --input-flip=rotate-180

causes video to freeze.

Have to click image closed & ^C command line.

Changing gstCamera.cpp ineffective.

Changing csi_camera.py ineffective.

--input-flip=FLIP flip method to apply to input:

* none (default)
* counterclockwise
* **rotate-180**

...EXCEPT when I do this, the video frame freezes.

root@PHELAN:/jetson-inference/build/aarch64/bin# **video-viewer**

--input-flip=rotate-180 csi://0

Back to Dusty's video....

Tried viewing a rtp/rtsp stream. Found some sources online which would display in browser but couldn't get to show in video-viewer with error:

[gstreamer] gstDecoder - - failed to retrieve next image buffer

video-viewer: failed to capture video frame

Tried several variations in command line but all failed.

Use RTP to display headlessly:

My PC IP address is 192.168.1.99

PuTTY - open NANOeth0 192.168.1.26

ubuntu@NANO:~/jetson-inference$ **docker/run.sh**

root@NANO:/jetson-inference/build/aarch64/bin **detectnet**

--headless csi://0 rtp://192.168.1.99:1234

On PC install gstreamer from:

https://gstreamer.freedesktop.org/download/

command prompt > gst-launch-1.0 -v udpsrc port=1234 caps = "application/x-rtp, media=(string)video, clock-rate=(int)90000, encoding-name=(string)H264, payload=(int)96" ! rtph264depay ! decodebin ! videoconvert ! autovideosink

'gst-launch-1.0' is not recognized as an internal or external command, operable program or batch file.

Found gstreamer was installed on S: backup drive!?!

Deleted it. Unplugged backup drive. Reinstalled to C:/

>cd **C:\gstreamer\1.0\msvc_x86_64\bin**

and ran from there: [all one line, no \'s

> **gst-launch-1.0 -v udpsrc port=1234 caps = "application/x-rtp, media=(string)video, clock-rate=(int)90000, encoding-name=(string)H264, payload=(int)96" ! rtph264depay ! decodebin ! videoconvert ! autovideosink**

It didn't blow up. It slowly output some statements.

Recheck the sending side and saw I left off the :1234 at the end.

Fixing that, restarting transmit, restarting receive & VOILA!

A window popped up & displayed the camera image on the PC!!!!!

Can't get it to work with VLC regardless of the .sdp file configuration!?

Set McAfee firewall to allow VLC. Didn't fix it.

Can't find an online rtp/rstp source that I can get to run.

2022.03.27

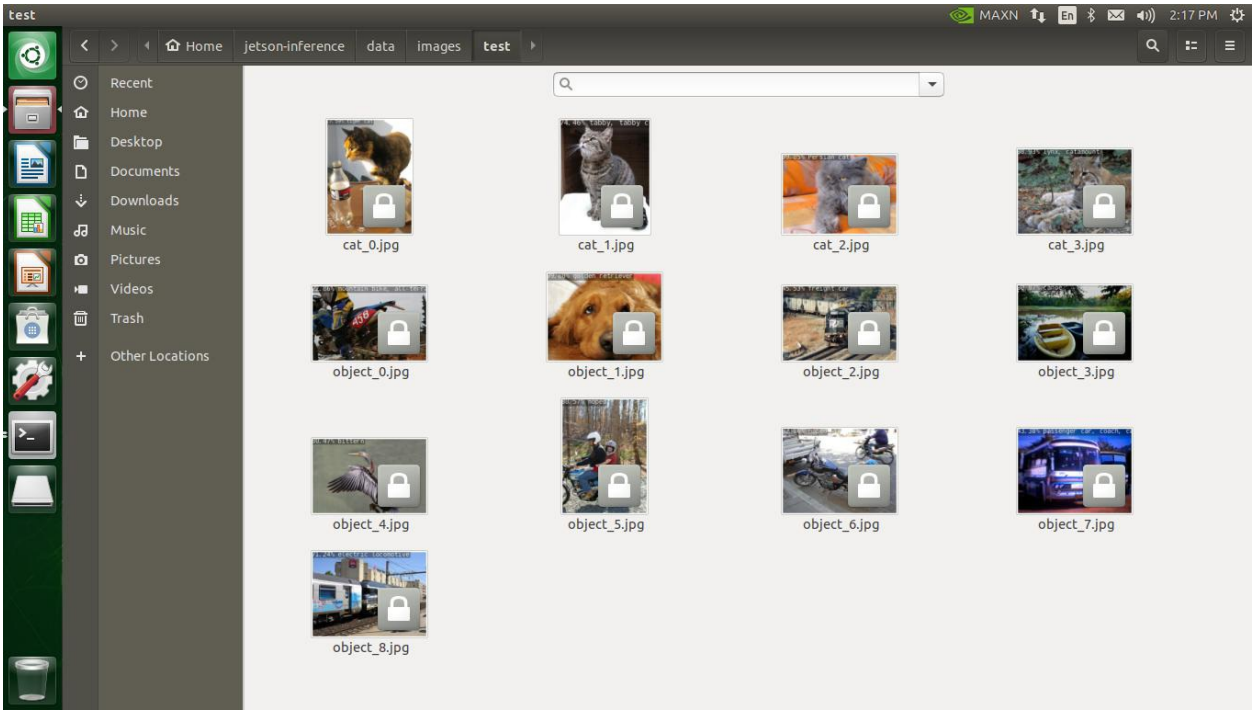
Jetson AI Fundamentals S3E2 Image Classification Inference

https://www.youtube.com/watch?v=QatH8iF0Efk

ubuntu@NANO:~/jetson-inference$ **docker/run.sh**

root@NANO:/jetson-inference/build/aarch64/bin# **./imagenet "images/object_*.jpg" "images/test/object_%i.jpg"**

root@NANO:/jetson-inference/build/aarch64/bin# **./imagenet "images/cat_*.jpg" "images/test/cat_%i.jpg"**



jellyfish.mkv

Coding Your Own Image Recognition Program (Python)

ubuntu@NANO:~$ **mkdir my-recognition**

ubuntu@NANO:~$ **cd my-recognition/**

ubuntu@NANO:~/my-recognition$ **touch my-recognition.py**

ubuntu@NANO:~/my-recognition$ **wget https://github.com/dusty-nv/jetson-inference/raw/master/data/images/black_bear.jpg**

ubuntu@NANO:~/my-recognition$ **wget https://github.com/dusty-nv/jetson-inference/raw/master/data/images/brown_bear.jpg**

ubuntu@NANO:~/my-recognition$ **wget https://github.com/dusty-nv/jetson-inference/raw/master/data/images/polar_bear.jpg**

ubuntu@NANO:~/jetson-inference$ **docker/run.sh --volume ~/my-recognition:/my-recognition** *[source:destination in docker]*

Back to host to edit program:

ubuntu@NANO:~/my-recognition$ **nano my-recognition.py**

cc from github:

```
#!/usr/bin/python3

import jetson.inference

import jetson.utils

import argparse

# parse the command line

parser = argparse.ArgumentParser()

parser.add_argument("filename", type=str, help="filename of the image to process")

parser.add_argument("--network", type=str, default="googlenet", help="model to use, can be:  googlenet, resnet-18, ect.")

args = parser.parse_args()

# load an image (into shared CPU/GPU memory)

img = jetson.utils.loadImage(args.filename)

# load the recognition network

net = jetson.inference.imageNet(args.network)

# classify the image

class_idx, confidence = net.Classify(img)

# find the object description

class_desc = net.GetClassDesc(class_idx)

# print out the result

print("image is recognized as '{:s}' (class #{:d}) with {:f}% confidence".format(class_desc, class_idx, confidence * 100))
```

ubuntu@NANO:~/my-recognition$ **python3 my-recognition.py black_bear.jpg**

```
Traceback (most recent call last):
  File "my-recognition.py", line 3, in <module>
    import jetson.inference
ModuleNotFoundError: No module named 'jetson'
```

Oops! I'm running from OUTSIDE the docker! Get back in....

root@NANO:/my-recognition# **python3 my-recognition.py black_bear.jpg**

```
...
image is recognized as 'American black bear, black bear, Ursus americanus, Euarctos americanus' (class #295) with 98.925781% confidence
```

root@NANO:/my-recognition# **python3 my-recognition.py brown_bear.jpg**

```
image is recognized as 'brown bear, bruin, Ursus arctos' (class #294) with 99.951172% confidence
```

root@NANO:/my-recognition# **python3 my-recognition.py polar_bear.jpg**

```
image is recognized as 'ice bear, polar bear, Ursus Maritimus, Thalarctos maritimus' (class #296) with 100.000000% confidence
```

Running the Live Camera Recognition Demo

ubuntu@NANO:~/jetson-inference$ **docker/run.sh --volume ~/my-recognition:/my-recognition** *[source:destination in docker]*

root@NANO:/my-recognition **imagenet csi://0**

wildly inaccurate!!

Jetson AI Fundamentals - S3E3 - Training Image Classification Models

Transfer Learning with PyTorch

https://github.com/dusty-nv/jetson-inference/blob/master/docs/pytorch-transfer-learning.md

Re-training on the Cat/Dog Dataset

https://github.com/dusty-nv/jetson-inference/blob/master/docs/pytorch-cat-dog.md

ubuntu@NANO:~$ **cd jetson-inference/python/training/classification/**

ubuntu@NANO:~/jetson-inference/python/training/classification$ **ls**

data **models** onnx_export.py onnx_validate.py README.md requirements.txt reshape.py train.py

ubuntu@NANO:~/jetson-inference/python/training/classification$ **cd data**

ubuntu@NANO:~/jetson-inference/python/training/classification/data$ **wget https://nvidia.box.com/shared/static/o577zd8yp3lmxf5zhm38svrbrv45am3y.gz -O cat_dog.tar.gz**

ubuntu@NANO:~/jetson-inference/python/training/classification/data$ **tar xvzf cat_dog.tar.gz**

ubuntu@NANO:~/jetson-inference/python/training/classification/data$ **cd ~/jetson-inference/**

ubuntu@NANO:~/jetson-inference$ **docker/run.sh**

```
reading L4T version from /etc/nv_tegra_release
L4T BSP Version:  L4T R32.7.1
[sudo] password for ubuntu:
```

root@NANO:/jetson-inference# **cd python/training/classification/**

root@NANO:/jetson-inference/python/training/classification# **ls data/cat_dog**

```
labels.txt  test  train  val
```

root@NANO:/jetson-inference/python/training/classification# **python3 train.py --model-dir=models/cat_dog data/cat_dog --batch-size=4 --workers=1 --epochs=10 data/cat_dog**

example epochs=1, default=35, ideal=100 (overnight)

```
train.py: error: unrecognized arguments: data/cat_dog
```

root@NANO:/jetson-inference/python/training/classification# **python3 train.py --model-dir=models/cat_dog --batch-size=4 --workers=1 --epochs=10 data/cat_dog**

```
* Acc@1 50.000 Acc@5 100.000
```

[accuracy 50% is pretty lame for 2 catagory test! Dusty did 50% with just 1 epoch! Should be ~65%]

```
saved checkpoint to:  models/cat_dog/checkpoint.pth.tar
```

Converting the Model to ONNX *[NOT onyx!]*

root@NANO:/jetson-inference/python/training/classification# **python3 onnx_export.py --model-dir=models/cat_dog**

```
model exported to:  models/cat_dog/resnet18.onnx
```

root@NANO:/jetson-inference/python/training/classification# **ls models/cat_dog/**

```
checkpoint.pth.tar  model_best.pth.tar  resnet18.onnx
```

Processing Images with TensorRT

Processing all the Test Images

mkdir $DATASET/test_output_cat $DATASET/test_output_dog

becomes

root@NANO:/jetson-inference/python/training/classification# **mkdir data/cat_dog/test_cat_output data/cat_dog/test_dog_output**

root@NANO:/jetson-inference/python/training/classification# **ls data/cat_dog/**

```
labels.txt  test  test_cat_output  test_dog_output  train  val
```

imagenet --model=$NET/resnet18.onnx --input_blob=input_0 --output_blob=output_0 --labels=$DATASET/../labels.txt $DATASET/test/cat $DATASET/test_output_cat *[becomes]*

root@NANO:/jetson-inference/python/training/classification# **imagenet --model=models/cat_dog/resnet18.onnx --labels=data/cat_dog/labels.txt --input_blob=input_0 --output_blob=output_0 data/cat_dog/test/cat data/cat_dog/test_cat_output**

and

root@NANO:/jetson-inference/python/training/classification# **imagenet --model=models/cat_dog/resnet18.onnx --labels=data/cat_dog/labels.txt --input_blob=input_0 --output_blob=output_0 data/cat_dog/test/dog data/cat_dog/test_dog_output**

root@NANO:/jetson-inference/python/training/classification# **imagenet --model=models/cat_dog/resnet18.onnx --labels=data/cat_dog/labels.txt --input_blob=input_0 --output_blob=output_0 /dev/video0**

Need to run this one from the Nano with display. Another day!

After class

root@NANO:/jetson-inference/python/training/classification# **python3 train.py --model-dir=models/cat_dog --batch-size=4 --workers=1 --epochs=100 data/cat_dog**

let run overnight...

root@NANO:/jetson-inference/python/training/classification# **python3 onnx_export.py --model-dir=models/cat_dog**

```
model exported to:  models/cat_dog/resnet18.onnx
```

root@NANO:/jetson-inference/python/training/classification# **ls models/cat_dog/**

```
checkpoint.pth.tar  model_best.pth.tar  resnet18.onnx
```

root@NANO:/jetson-inference/python/training/classification# **imagenet --model=models/cat_dog/resnet18.onnx --labels=data/cat_dog/labels.txt --input_blob=input_0 --output_blob=output_0 data/cat_dog/test/cat data/cat_dog/test_cat_output**

and

root@NANO:/jetson-inference/python/training/classification# **imagenet --model=models/cat_dog/resnet18.onnx --labels=data/cat_dog/labels.txt --input_blob=input_0 --output_blob=output_0 data/cat_dog/test/dog data/cat_dog/test_dog_output**

Examining test_dog_output and test_cat_output after 100 epoch retraining:

Images were only classified ~50% correctly with only 50-55% confidence. Pretty miserable performance, not much better than a coin toss!

Now to try the Tools classification experiment.

Skipped over the camera-capture tool here

Yesterday's class consisted of 8 plastic tools:


DRILL


HAMMER


PIPE-WRENCH


PLIERS


SAW


SCREWDRIVER


SLIP-WRENCH


TONGS

root@NANO:/jetson-inference/python/training/classification# **python3 train.py --model-dir=models/Tools --batch-size=4 --workers=1 --epochs=35 data/Tools**

root@NANO:/jetson-inference/python/training/classification# **python3 onnx_export.py --model-dir=models/Tools**

root@NANO:/jetson-inference/python/training/classification# **imagenet --model=models/Tools/resnet18.onnx --labels=data/Tools/labels.txt --input_blob=input_0 --output_blob=output_0 data/Tools/test/drill data/Tools/test_drill_output**

Recognized with ~99% accuracy but gave this error for each:

[image] failed to save 1280x720 image to 'data/Tools/test_drill_output/0.jpg

[image] imagewriter --failed to save 'data/Tools/test_drill_output/0.jpg'

There's 76.7 GB of free space on my card.

Github Issues: couple similar probs but no fix.

2022.03.31

Here's the answer, I failed to create the output directories. I thought they'd make themselves:

mkdir $DATASET/test_output_cat $DATASET/test_output_dog

so

root@NANO:/jetson-inference/python/training/classification# **mkdir data/cat_dog/test_cat_output data/cat_dog/test_dog_output**

becomes

root@NANO:/jetson-inference/python/training/classification# **mkdir data/Tools/test_drill_output data/Tools/test_hammer_output data/Tools/test_pipe-wrench_output data/Tools/test_pliers_output data/Tools/test_saw_output data/Tools/test_screwdriver_output data/Tools/test_slip-wrench_output data/Tools/test_tongs_output**

root@NANO:/jetson-inference/python/training/classification# **imagenet --model=models/Tools/resnet18.onnx --labels=data/Tools/labels.txt --input_blob=input_0 --output_blob=output_0 data/Tools/test/drill data/Tools/test_drill_output**

Bingo!


10/10 drill


6/10 hammer


5/10 pipe-wren


10/10 pliers


9/10 saw


6/10 screwdriv


8/10 slip-wren


10/10 tongs

Looking back I see we should have created a class "background" to distinguish it from the actual samples.

Re-training SSD-Mobilenet

2022.03.31 continued

Software Updater

uncheck references to python 2.7

Hope this doesn't screw up nvidia jetpacks!

Software is "up to date" (we shall see...!)

ubuntu@NANO:~/jetson-inference$ **./docker/run.sh**

root@NANO:/jetson-inference/python/training/detection/ssd# **python3 open_images_downloader.py --stats-only --class-names "Apple,Orange,Banana,Strawberry,Grape,Pear,Pineapple,Watermelon" --data=data/fruit**

root@NANO:/jetson-inference/python/training/detection/ssd# **python3 train_ssd.py --data=data/fruit --model-dir=models/fruit --batch-size=2 --workers=1 --epochs=30**

...

numpy.core._exception._ArrayMemoryError: Unable to allocate 4.21 GiB for an array with shape (15854, 23782, 3) and data type float32

Looks like will have to disable GUI & try again. Maybe make swap file 8GB?

ubuntu@NANO:~$ sudo fallocate -l 8G /mnt/8GB.swap

ubuntu@NANO:~$ sudo chmod 600 /mnt/8GB.swap

ubuntu@NANO:~$ sudo mkswap /mnt/8GB.swap

Setting up swapspace version 1, size = 8 GiB (8589930496 bytes)

no label, UUID=0a79072f-804c-4e54-9713-3bc7937dea37

ubuntu@NANO:~$ sudo su

root@NANO:/home/ubuntu# echo "/mnt/8GB.swap swap swap swap defaults 0 0" >> /etc/fstab

Disabling the Desktop GUI

```
$ sudo init 3     # stop the desktop

# log your user back into the console

# run the PyTorch training scripts

$ sudo init 5     # restart the desktop

$ sudo systemctl set-default multi-user.target     # disable desktop on boot

$ sudo systemctl set-default graphical.target      # enable desktop on boot
```

Now, back above & try to retrain w/o desktop + 8GB swap

& try limiting data...

ubuntu@NANO:~/jetson-inference$ ***./docker/run.sh***

root@NANO:/jetson-inference/python/training/detection/ssd# ***python3 open_images_downloader.py --stats-only --class-names "Apple,Orange,Banana,Strawberry,Grape,Pear,Pineapple,Watermelon" --data=data/fruit***

```
2022-04-04 01:44:05 - Requested 8 classes, found 8 classes

2022-04-04 01:44:05 - Read annotation file
data/fruit/train-annotations-bbox.csv

2022-04-04 01:46:14 - Available train images:  5145

2022-04-04 01:46:14 - Available train boxes:   23539

2022-04-04 01:46:14 - Read annotation file
data/fruit/validation-annotations-bbox.csv

2022-04-04 01:46:16 - Available validation images:  285

2022-04-04 01:46:16 - Available validation boxes:   825

2022-04-04 01:46:16 - Read annotation file
data/fruit/test-annotations-bbox.csv

2022-04-04 01:46:19 - Available test images:  930

2022-04-04 01:46:19 - Available test boxes:   2824

2022-04-04 01:46:19 - Total available images: 6360

2022-04-04 01:46:19 - Total available boxes:  27188
```

```
-------------------------------------
 'train' set statistics
-------------------------------------
  Image count:  5145
  Bounding box count:  23539
  Bounding box distribution:
    Strawberry:  7553/23539 = 0.32
    Orange:  6186/23539 = 0.26
    Apple:  3622/23539 = 0.15
    Grape:  2560/23539 = 0.11
    Banana:  1574/23539 = 0.07
    Pear:  757/23539 = 0.03
    Watermelon:  753/23539 = 0.03
    Pineapple:  534/23539 = 0.02
-------------------------------------
 'validation' set statistics
-------------------------------------
  Image count:  285
  Bounding box count:  825
  Bounding box distribution:
    Strawberry:  326/825 = 0.40
    Grape:  153/825 = 0.19
    Orange:  148/825 = 0.18
    Apple:  102/825 = 0.12
    Watermelon:  31/825 = 0.04
    Pineapple:  25/825 = 0.03
    Banana:  22/825 = 0.03
    Pear:  18/825 = 0.02
```

root@NANO:/jetson-inference/python/training/detection/ssd# **python3 open_images_downloader.py --max-images=2500 --class-names "Apple,Orange,Banana,Strawberry,Grape,Pear,Pineapple,Watermelon" --data=data/fruit**

```
2022-04-04 01:51:11 - Requested 8 classes, found 8 classes
2022-04-04 01:51:11 - Read annotation file data/fruit/train-annotations-bbox.csv
2022-04-04 01:53:08 - Available train images:  5145
2022-04-04 01:53:08 - Available train boxes:   23539
2022-04-04 01:53:08 - Read annotation file data/fruit/validation-annotations-bbox.csv
2022-04-04 01:53:10 - Available validation images:  285
2022-04-04 01:53:10 - Available validation boxes:   825
2022-04-04 01:53:10 - Read annotation file data/fruit/test-annotations-bbox.csv
2022-04-04 01:53:13 - Available test images:  930
2022-04-04 01:53:13 - Available test boxes:   2824
2022-04-04 01:53:13 - Total available images: 6360
2022-04-04 01:53:13 - Total available boxes:  27188
2022-04-04 01:53:13 - Limiting train dataset to:  2022 images (9240 boxes)
2022-04-04 01:53:13 - Limiting validation dataset to:  112 images (318 boxes)
2022-04-04 01:53:13 - Limiting test dataset to:  365 images (1093 boxes)
-------------------------------------
 'train' set statistics
-------------------------------------
  Image count:  2022
  Bounding box count:  9240
  Bounding box distribution:
    Strawberry:  2822/9240 = 0.31
    Orange:  2452/9240 = 0.27
    Apple:  1413/9240 = 0.15
    Grape:  1083/9240 = 0.12
    Banana:  580/9240 = 0.06
    Pear:  356/9240 = 0.04
    Watermelon:  335/9240 = 0.04
    Pineapple:  199/9240 = 0.02


-------------------------------------
 'validation' set statistics
-------------------------------------
  Image count:  112
  Bounding box count:  318
  Bounding box distribution:
    Strawberry:  114/318 = 0.36
    Grape:  64/318 = 0.20
    Orange:  64/318 = 0.20
```

```
    Apple:  43/318 = 0.14
    Banana:  12/318 = 0.04
    Watermelon:  10/318 = 0.03
    Pineapple:  8/318 = 0.03
    Pear:  3/318 = 0.01
-------------------------------------
 'test' set statistics
-------------------------------------
  Image count:  365
  Bounding box count:  1093
  Bounding box distribution:
    Orange:  324/1093 = 0.30
    Strawberry:  287/1093 = 0.26
    Apple:  151/1093 = 0.14
    Grape:  147/1093 = 0.13
    Pineapple:  52/1093 = 0.05
    Watermelon:  52/1093 = 0.05
    Pear:  45/1093 = 0.04
    Banana:  35/1093 = 0.03
-------------------------------------
 Overall statistics
-------------------------------------
  Image count:  2499
  Bounding box count:  10651
2022-04-04 01:53:13 - Saving 'train' data to data/fruit/sub-train-annotations-bbox.csv.
2022-04-04 01:53:14 - Saving 'validation' data to data/fruit/sub-validation-annotations-bbox.csv.
2022-04-04 01:53:14 - Saving 'test' data to data/fruit/sub-test-annotations-bbox.csv.
2022-04-04 01:53:14 - Starting to download 2499 images.
2022-04-04 01:53:15 - Downloaded 100 images.
2022-04-04 01:53:15 - Downloaded 200 images.
2022-04-04 01:53:15 - Downloaded 300 images.
2022-04-04 01:53:15 - Downloaded 400 images.
2022-04-04 01:53:16 - Downloaded 500 images.
2022-04-04 01:53:16 - Downloaded 600 images.
2022-04-04 01:53:16 - Downloaded 700 images.
2022-04-04 01:53:16 - Downloaded 800 images.
2022-04-04 01:53:16 - Downloaded 900 images.
2022-04-04 01:53:17 - Downloaded 1000 images.
2022-04-04 01:53:17 - Downloaded 1100 images.
2022-04-04 01:53:17 - Downloaded 1200 images.
2022-04-04 01:53:17 - Downloaded 1300 images.
2022-04-04 01:53:17 - Downloaded 1400 images.
2022-04-04 01:53:18 - Downloaded 1500 images.
```

2022-04-04 01:53:18 - Downloaded 1600 images.

2022-04-04 01:53:18 - Downloaded 1700 images.

2022-04-04 01:53:18 - Downloaded 1800 images.

2022-04-04 01:53:19 - Downloaded 1900 images.

2022-04-04 01:53:19 - Downloaded 2000 images.

2022-04-04 01:53:19 - Downloaded 2100 images.

2022-04-04 01:53:19 - Downloaded 2200 images.

2022-04-04 01:53:19 - Downloaded 2300 images.

2022-04-04 01:53:20 - Downloaded 2400 images.

2022-04-04 01:53:20 - Task Done.

root@NANO:/jetson-inference/python/training/detection/ssd# **python3 train_ssd.py --data=data/fruit --model-dir=models/fruit --batch-size=2 --workers=1 --epochs=3** [30 WAY too much!]

running...

- Saved model models/fruit/mb1-ssd-Epoch-2-Loss-5.6—.pth

- Task done, exiting program.

Converting the Model to ONNX

root@NANO:/jetson-inference/python/training/detection/ssd# **python3 onnx_export.py --model-dir=models/fruit**

model exported to: models/fruit/ssd-mobilenet.onnx

task done, exiting program

Processing Images with TensorRT

To run these commands, the working directory of your terminal should still be located in: jetson-inference/python/training/detection/ssd/

IMAGES=<path-to-your-jetson-inference>/data/images # substitute your jetson-inference path here

becomes IMAGES=/jetson-inference/data/images *which becomes*

root@NANO:/jetson-inference/python/training/detection/ssd# **detectnet --model=models/fruit/ssd-mobilenet.onnx --labels=models/fruit/labels.txt --input-blob=input_0 --output-cvg=scores --output-bbox=boxes**

"/jetson-inference/data/images/fruit_*.jpg"

/jetson-inference/data/images/test/fruit_%i.jpg

looooonnnngggg run

```
[TRT] CUDA engine context initialized on device GPU:
[TRT]    – layers     106
[TRT]    – maxBatchSize 1
[TRT]    – deviceMemory 8881664
[TRT]    – bindings   3
[TRT]    binding 0
...
[TRT]    binding 1
...
[TRT]    binding 2
...
[TRT] 3: Cannot find binding of given name: data
[TRT] failed to find requested input layer data in network
[TRT] device GPU, failed to create resources for CUDA engine
[TRT] detectnet – failed to initialize.
detectnet: failed to load detectNet model
```

I think it's because, based on prev lesson, I made a class "BACKGROUND" but didn't provide any background images to train.

Deleted BACKGROUND from labels.txt.

I think I need to delete the files created during training and start over.

ubuntu@NANO:~/jetson-inference/python/training/detection/ssd/models/fruit$ **sudo rm *.pth**

Permission denied

Try running again w/ corrected labels.txt file.

Same errors.

Can rm files from w/in docker:

rm *.pth

rm *.onnx

Try this again:

root@NANO:/jetson-inference/python/training/detection/ssd# **python3 train_ssd.py --data=data/fruit --model-dir=models/fruit --batch-size=2 --workers=1 --epochs=3**

2022-04-06 0 : : = *[but it's 2022.04.05!]*

...

root@NANO:/jetson-inference/python/training/detection/ssd# **python3 onnx_export.py --model-dir=models/fruit**

root@NANO:/jetson-inference/python/training/detection/ssd# **detectnet --model=models/fruit/ssd-mobilenet.onnx --labels=models/fruit/labels.txt --input-blob=input_0 --output-cvg=scores --output-bbox=boxes**

"/jetson-inference/data/images/fruit_*.jpg"

/jetson-inference/data/images/test/fruit_%i.jpg

same error!

Tried w/o wildcard fruit_0.jpg test/fruit_0.jpg

same error.

Try test/fruit_out_%i.jpg

same error

Let's see if Dusty's YouTube video reveals the secret incantation:

Jetson AI Fundamentals - S3E4 - Object Detection Inference

https://www.youtube.com/watch?v=obt60r8ZeB0

been there, done that already.

Jetson AI Fundamentals - S3E5 - Training Object Detection Models

https://www.youtube.com/watch?v=2XMkPW_sIGg

this is what I'm looking for...

Re-training SSD-Mobilenet

Hmmm, maybe it's hanging up because it can't display the output because the desktop GUI is disabled...?

$ **sudo init 5**   # restart the desktop

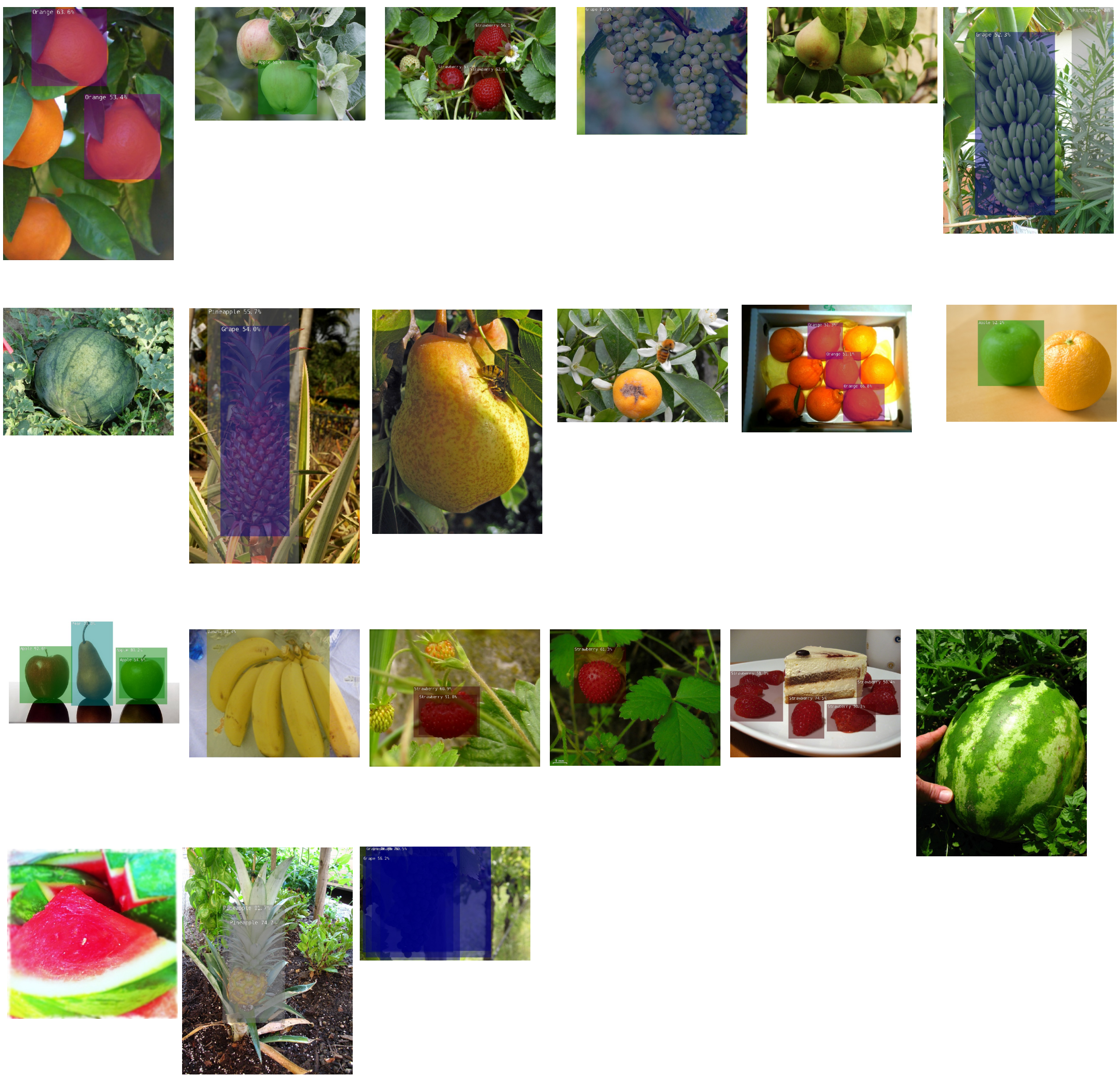
$ **sudo systemctl set-default graphical.target**   # enable desktop on boot

root@NANO:/jetson-inference/python/training/detection/ssd# **detectnet --model=models/fruit/ssd-mobilenet.onnx --labels=models/fruit/labels.txt --input-blob=input_0 --output-cvg=scores --output-bbox=boxes**

"/jetson-inference/data/images/fruit_*.jpg"

/jetson-inference/data/images/test/fruit_%i.jpg

YES!!!!!



2022.04.09

Nano autoupdate telling me updates available.

D’ says $ sudo apt-get update works better & does more.

Since between sessions:

ubuntu@NANO:~$ **sudo apt-get update**

done

ubuntu@NANO:~$ **sudo apt-get upgrade**

done, didn’t do much

ubuntu@NANO:~$ **cd jetson-inference/**

ubuntu@NANO:~/jetson-inference$ **docker/run.sh**

root@NANO:/jetson-inference# **apt-get update**

updates a lot

root@NANO:/jetson-inference# **apt-get upgrade**

upgrades a lot. Done.

Now, to continue...

https://www.youtube.com/watch?v=2XMkPW_sIGg

https://github.com/dusty-nv/jetson-inference/blob/master/docs/pytorch-collect-detection.md

Collecting your own Detection Datasets

ubuntu@NANO:~/jetson-inference/python/training/detection/ssd/data$ **mkdir tools**

ubuntu@NANO:~/jetson-inference/python/training/detection/ssd/data/tools$ **nano labels.txt** *[no BACKGROUND class]*

```
clamp
cutters
pliers
screwdriver
stripper
tape
```

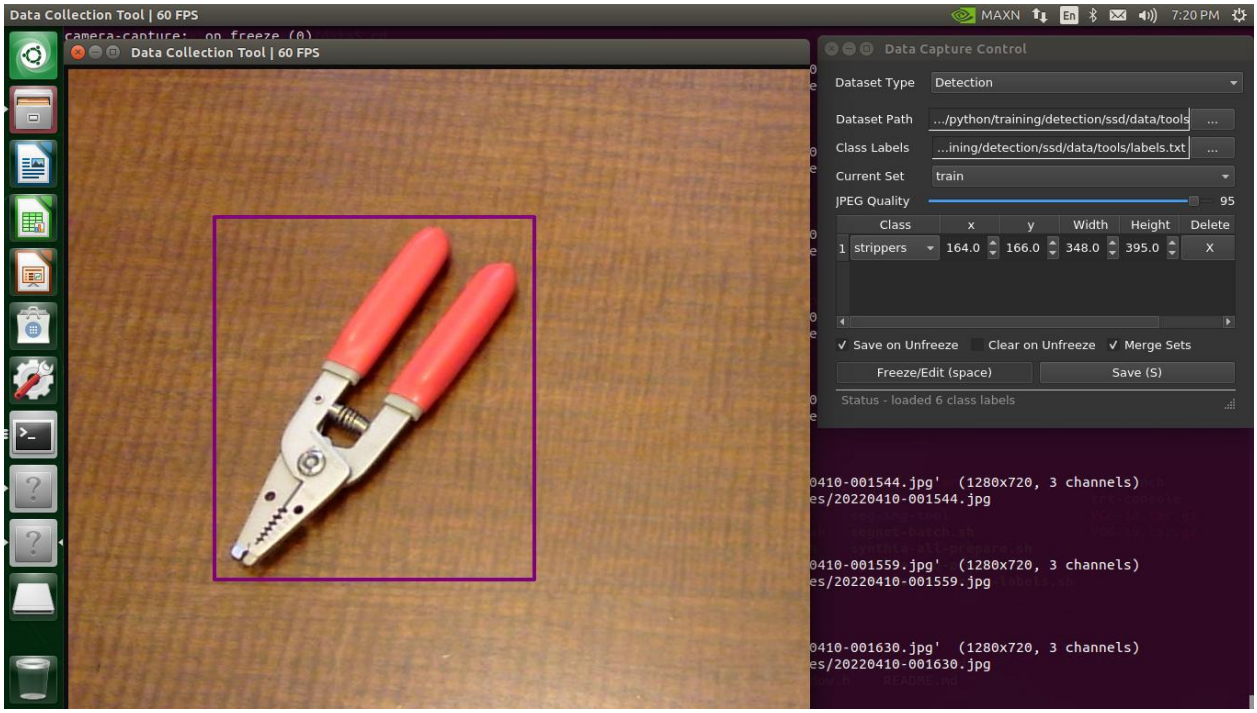
RUN DOCKER:

ubuntu@NANO:~/jetson-inference$ ***./docker/run.sh***

root@NANO:~/jetson-inference/python/training/detection/ssd/data/tools# **cd ../..**

root@NANO:~/jetson-inference/python/training/detection/ssd# **camera-capture /dev/video0**

I dont'see any counter per class to keep track of how many I've done!! I can use file folder/properties to see how many in the folder but it's awkward.



Training your Model

$ python3 train_ssd.py --dataset-type=voc --data=data/<YOUR-DATASET> --model-dir=models/<YOUR-MODEL>

becomes

root@NANO:~/jetson-inference/python/training/detection/ssd# **python3 train_ssd.py --dataset-type=voc --data=data/tools --model-dir=models/tools --batch-size=2 --workers=1 --epochs=3**

19:42 - 21:10

Stuck at Epoch: 1, Step: 70/75

^C

turn off Desktop GUI:

In home terminal:

$ **sudo init 3** # stop the desktop

GUI disappears

ubuntu@NANO:~/jetson-inference$ ***./docker/run.sh***

Unable to open display *[expected]*

root@NANO:~/jetson-inference/python/training/detection/ssd#
python3 train_ssd.py --dataset-type=voc --data=data/tools --model-dir=models/tools --batch-size=2 --workers=1 --epochs=3

Ran overnight, still stuck at same place...?!

Swap space = 8 GB

Try batch-size=1, epochs=1

Gets stuck at Step: 140/150

based on advice from:

https://github.com/dusty-nv/jetson-inference/issues/1353

and tried with ***--debug-steps=1*** *(instead of default 10) to try to catch where it crashed. This time it finished!?!*

root@NANO:/jetson-inference/python/training/detection/ssd#
python3 onnx_export.py --model-dir=models/tools

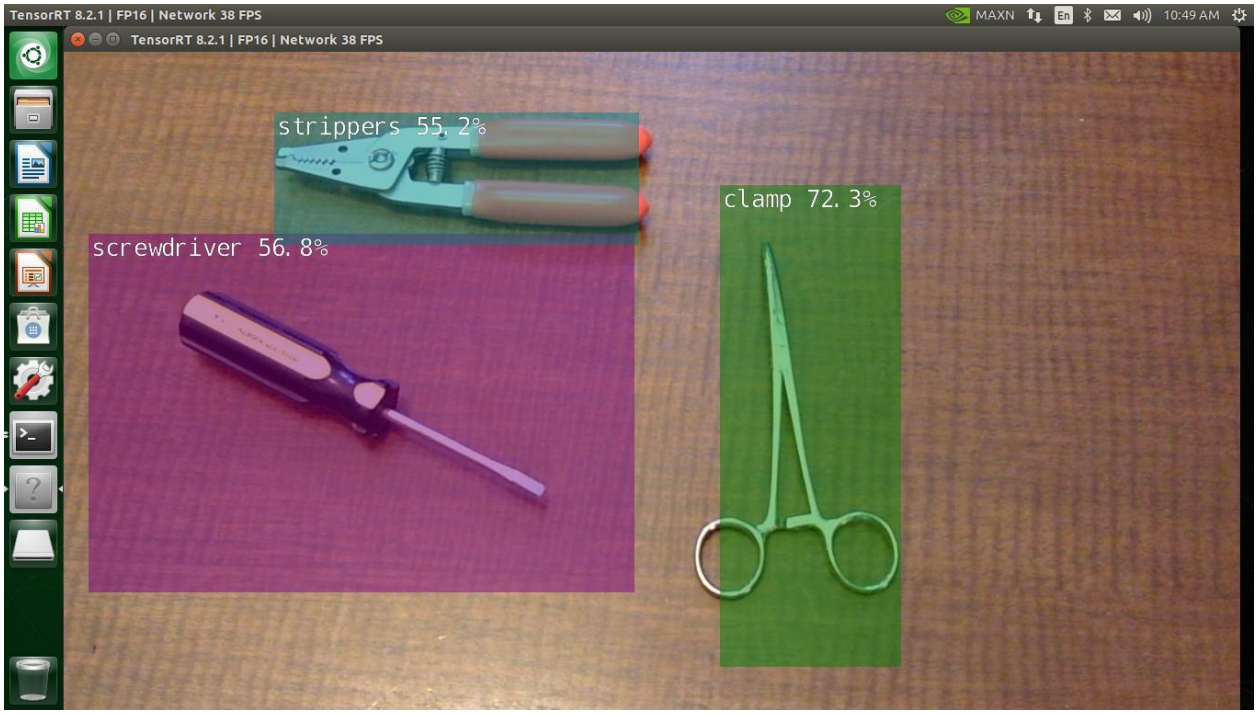
task done, exiting program

root@NANO:/jetson-inference/python/training/detection/ssd#
detectnet --model=models/tools/ssd-mobilenet.onnx --labels=models/tools/labels.txt --input-blob=input_0 --output-cvg=scores --output-bbox=boxes /dev/video0

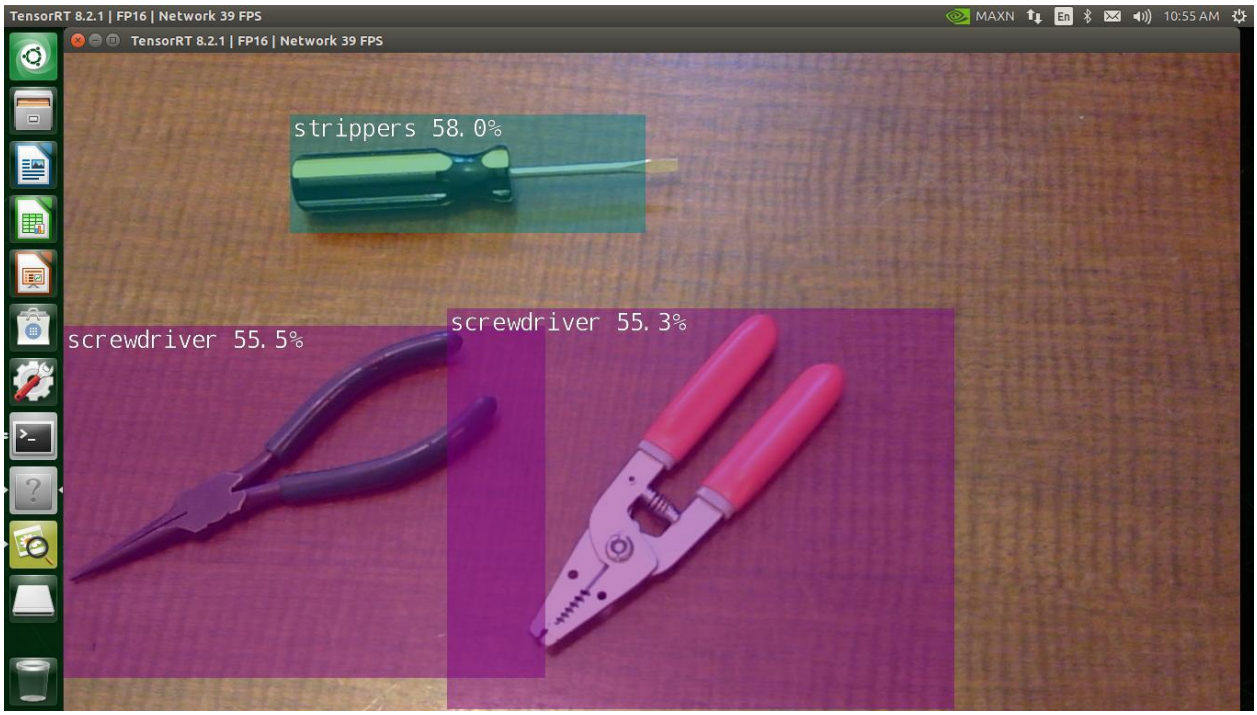
Forgot to restart Desktop GUI. [TNT]... keeps running and ^C won't stop it! Had to $ sudo reboot from PuTTY terminal. Rebooted w/ Desktop GUI in place as I didn't permanently disable it. Now to try again!

Fails to bring up display...

Finally, after several minutes while I search Issues for the problem, ***the video display suddenly pops open!*** *Does a terrible job of detecting items, but then I only trained w/ ~25 images each!*



Sometimes it gets it right...



...and sometimes not so much!

Jetson AI Fundamentals - S3E6 - Semantic Segmentation

https://www.youtube.com/watch?v=AQhkMLaB_fY

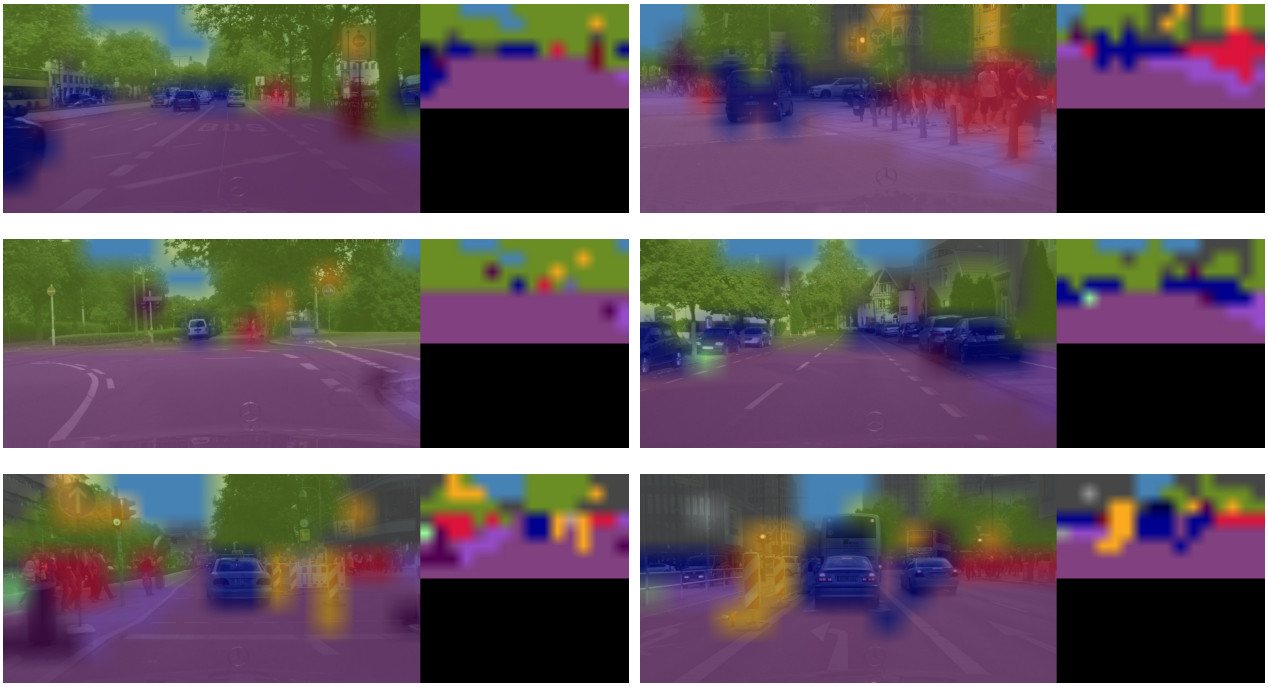
Semantic Segmentation with SegNet

https://github.com/dusty-nv/jetson-inference/blob/master/docs/segnet-console-2.md

Cityscapes

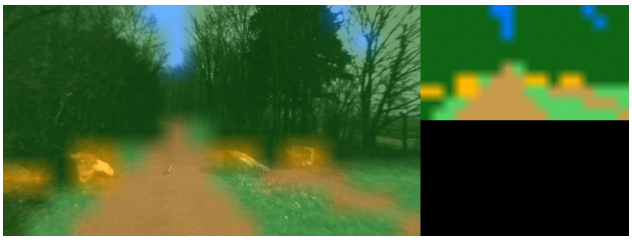
```
$ ./segnet --network=fcn-resnet18-cityscapes images/city_0.jpg
images/test/output.jpg        [becomes...]
```

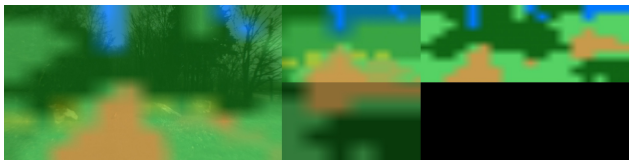
```
root@PHELAN:/jetson-inference/build/aarch64/bin# ./segnet
--network=fcn-resnet18-cityscapes “images/city_*.jpg”
images/test/city_%i.jpg
```



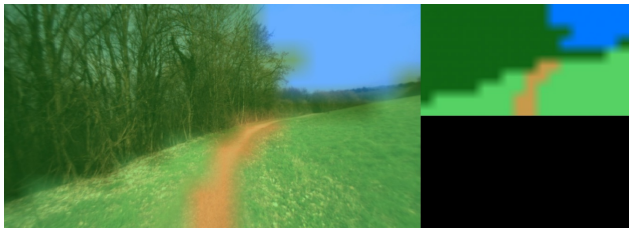
DeepScene

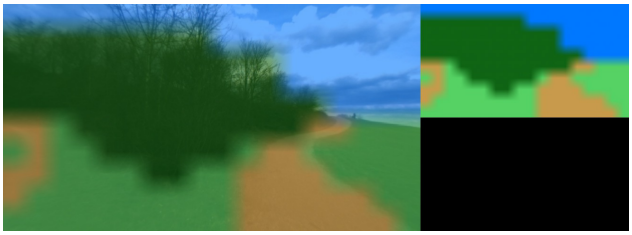
root@PHELAN:/jetson-inference/build/aarch64/bin# **segnet --network=fcn-resnet18-deepscene “images/trail_*.jpg” images/test/trail_%i.jpg**

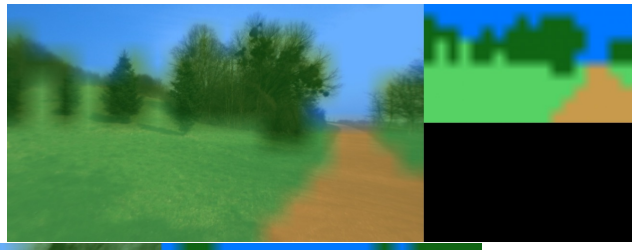


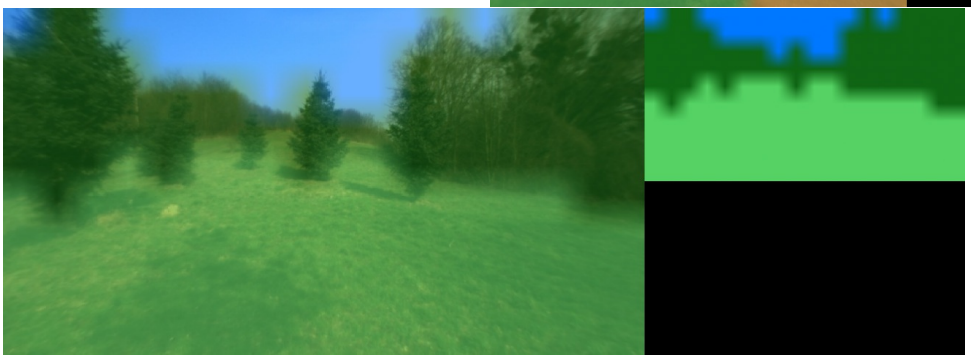


Usually it finds a trail...





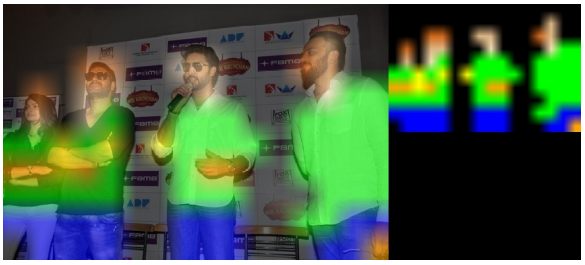


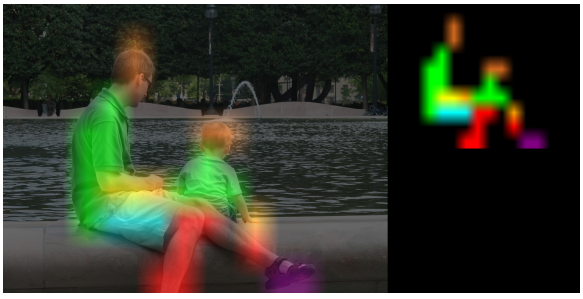


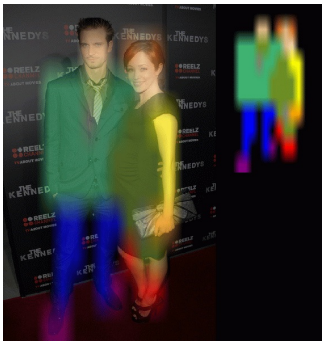
But sometimes there isn’t one

Multi-Human Parsing (MHP)

root@PHELAN:/jetson-inference/build/aarch64/bin# **segnet --network=fcn-resnet18-mhp “images/humans_*.jpg” images/test/humans_%i.jpg**

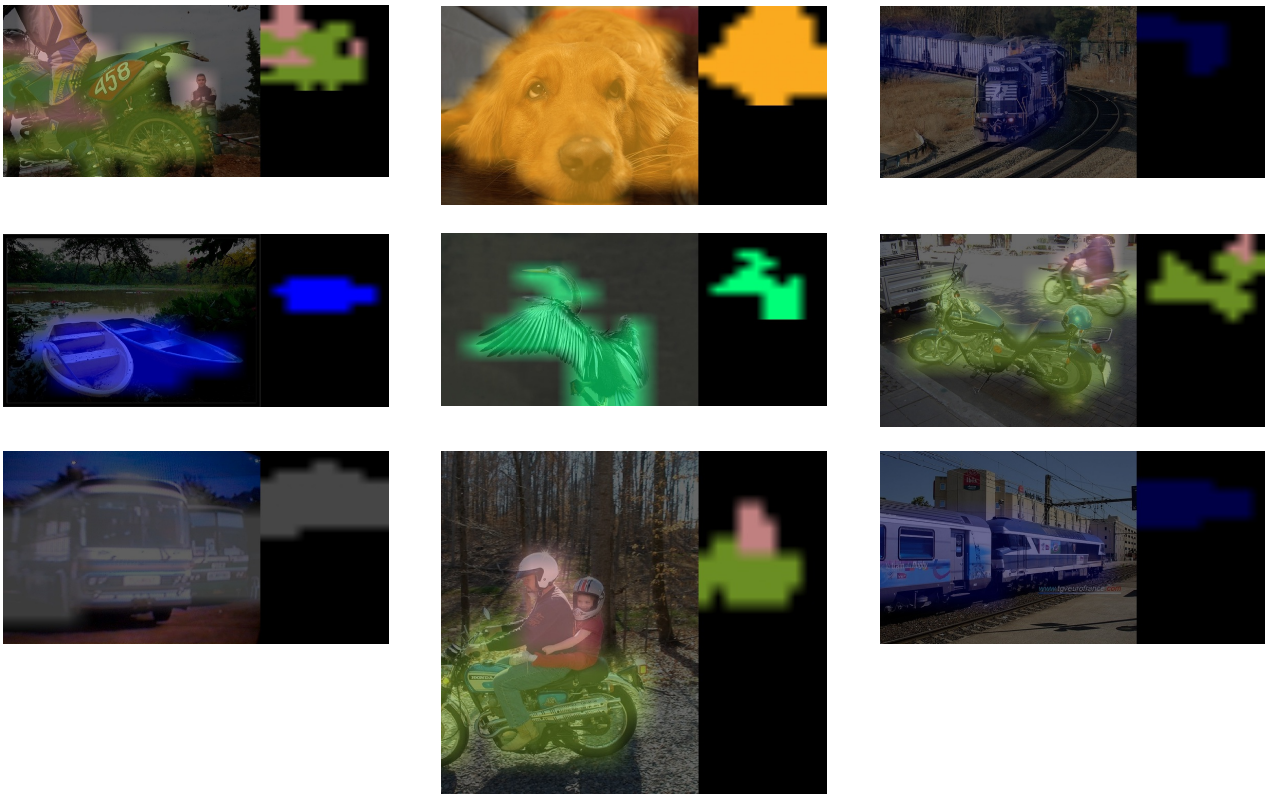






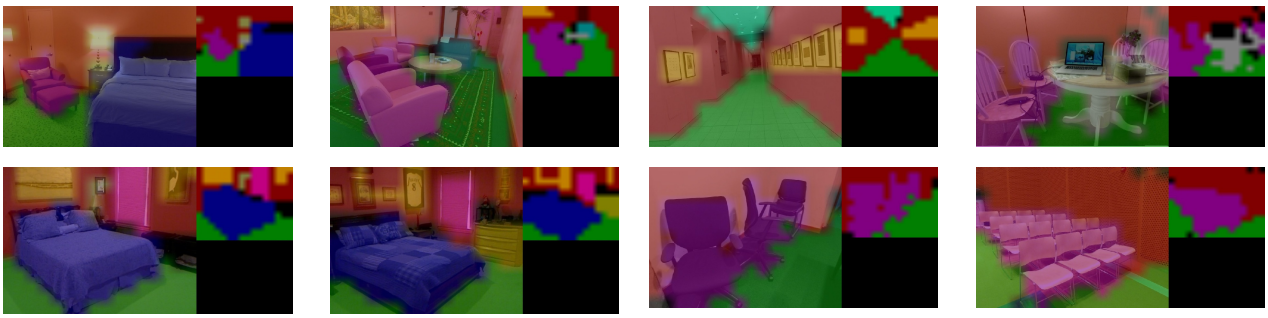
Pascal VOC

root@PHELAN:/jetson-inference/build/aarch64/bin# **segnet --network=fcn-resnet18-voc “images/object_*.jpg” images/test/object_%i.jpg**



SUN RGB-D

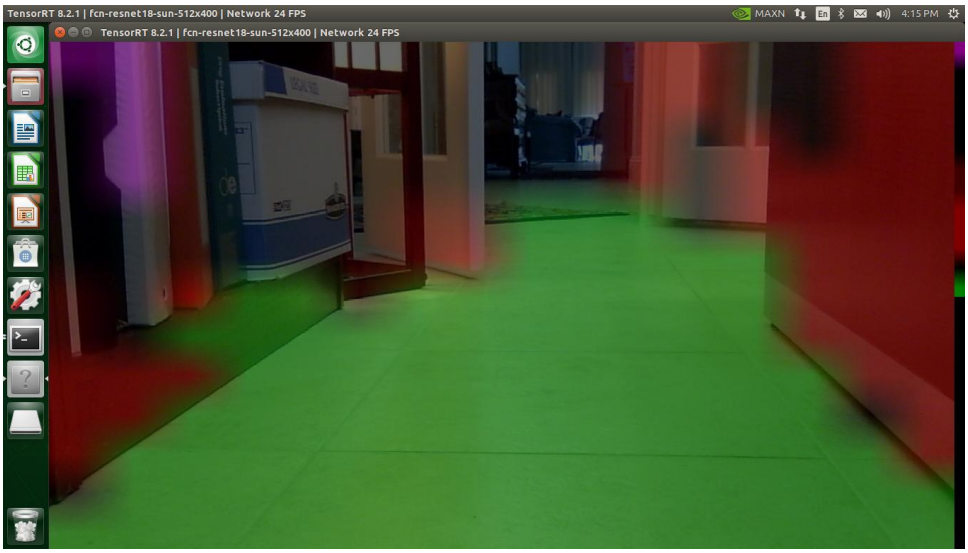
root@PHELAN:/jetson-inference/build/aarch64/bin# **segnet --network=fcn-resnet18-sun “images/room_*.jpg” images/test/room_%i.jpg**



Running the Live Camera Segmentation Demo

https://github.com/dusty-nv/jetson-inference/blob/master/docs/segnet-camera-2.md

root@PHELAN:/jetson-inference/build/aarch64/bin# **segnet --network=fcn-resnet18-sun /dev/video0**



Follow the Green Brick Road...!

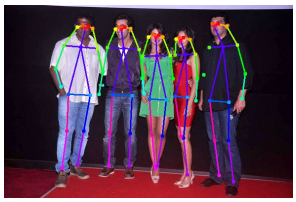
root@PHELAN:/jetson-inference/build/aarch64/bin# **segnet --network=fcn-resnet18-mhp /dev/video0**

Dusty's YouTube video stops here, but HELLO AI WORLD continues...

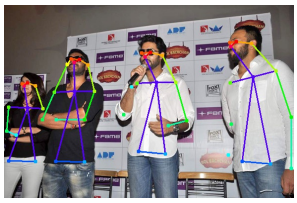
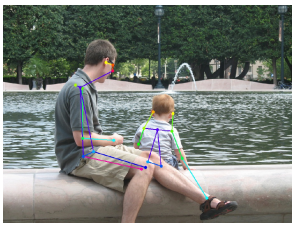
Pose Estimation with PoseNet

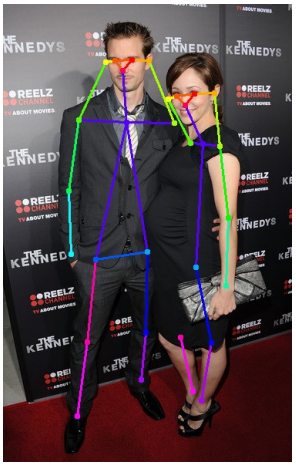
https://github.com/dusty-nv/jetson-inference/blob/master/docs/posenet.md

```
$ ./posenet "images/humans_*.jpg" images/test/pose_humans_%i.jpg
```



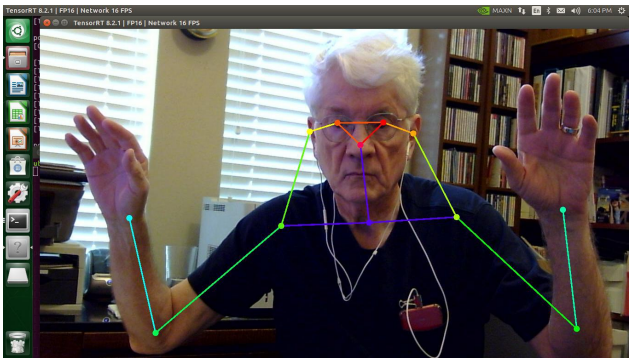






Pose Estimation from Video

```
root@PHELAN:/jetson-inference/build/aarch64/bin# ./posenet /dev/video0
```

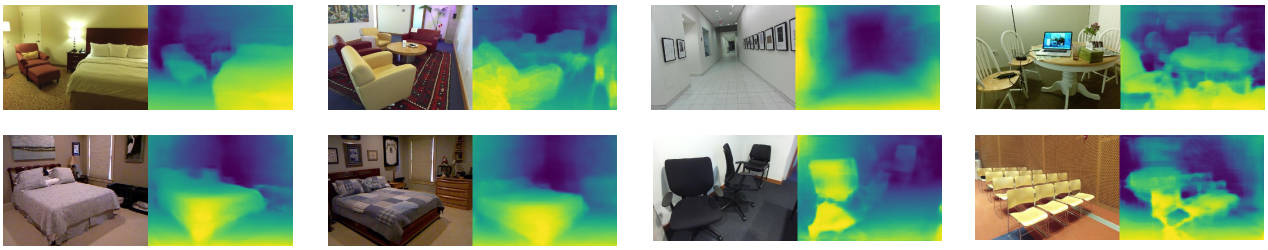
```
root@PHELAN:/jetson-inference/build/aarch64/bin# ./posenet --network=resnet18-hand /dev/video0
```



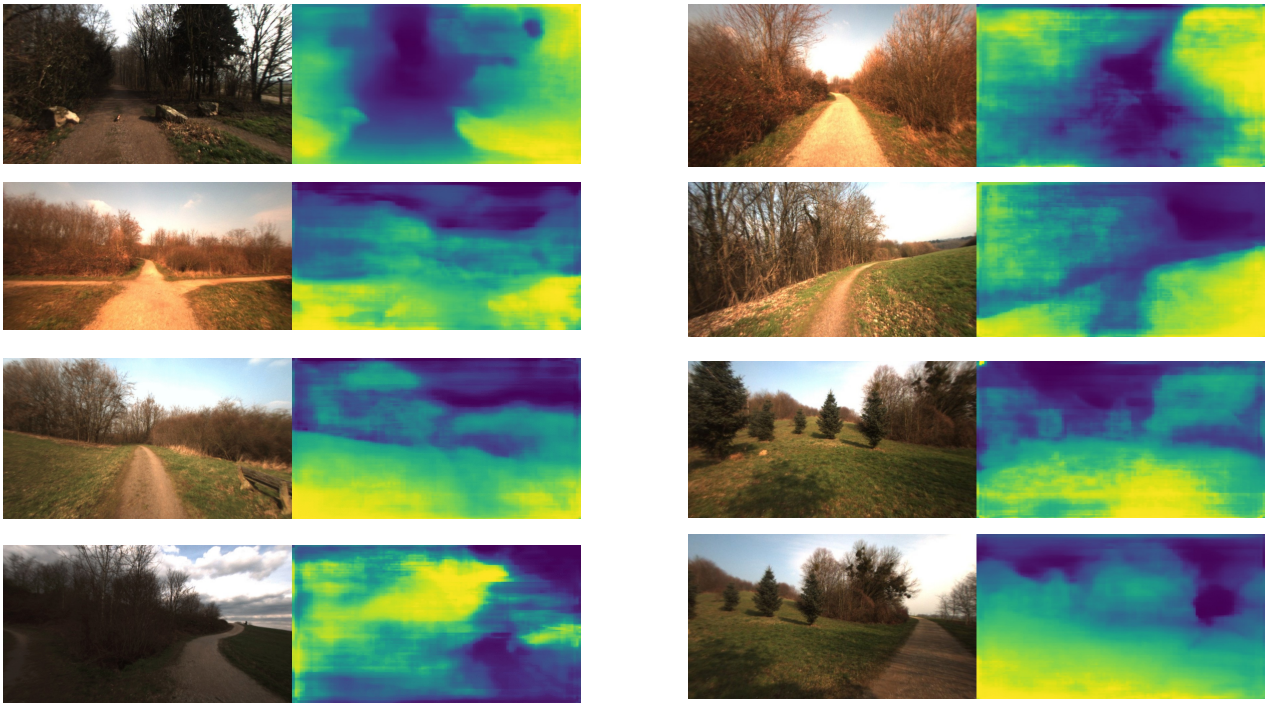

Monocular Depth with DepthNet

https://github.com/dusty-nv/jetson-inference/blob/master/docs/depthnet.md

root@PHELAN:/jetson-inference/build/aarch64/bin# **./depthnet "images/room_*.jpg" images/test/depth_room_%i.jpg**

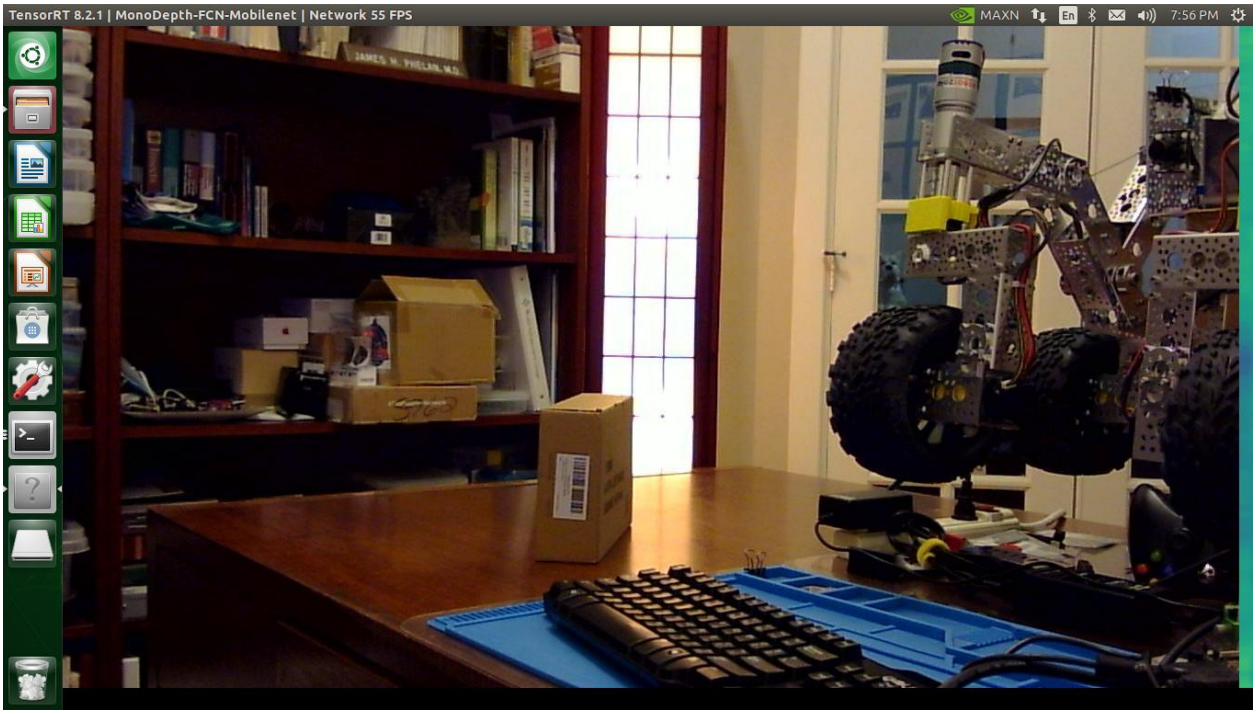


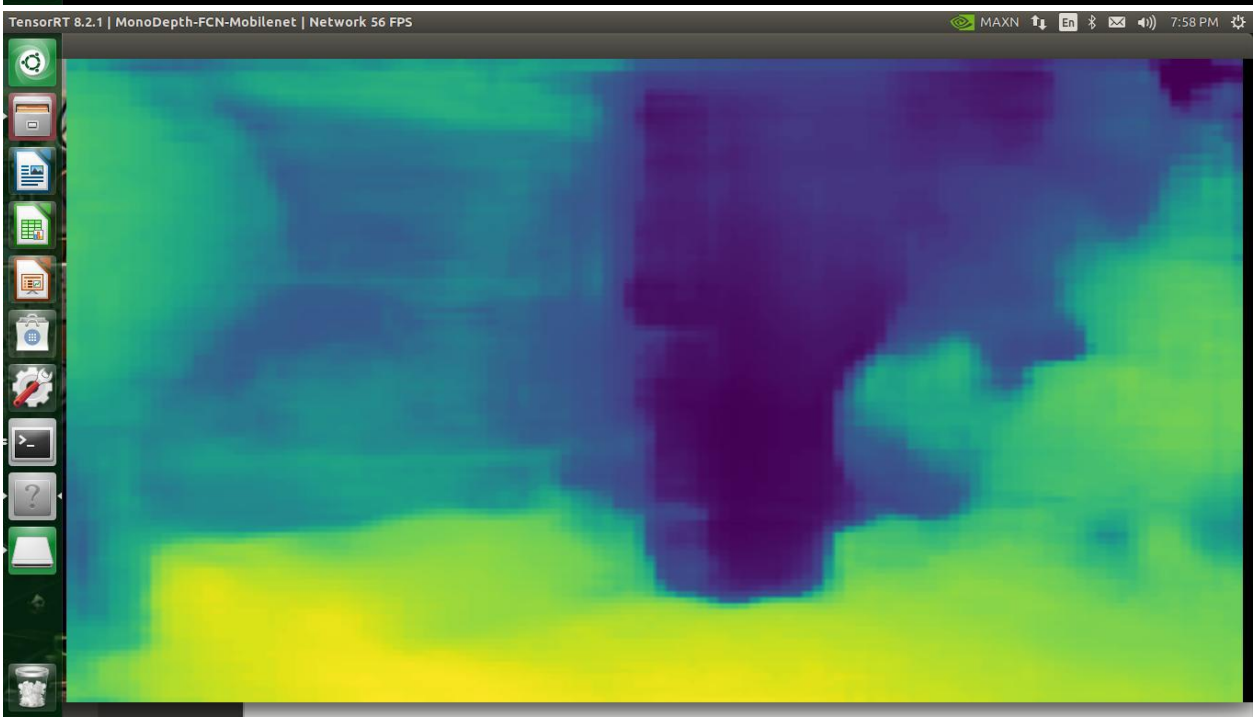
root@PHELAN:/jetson-inference/build/aarch64/bin# **./depthnet "images/trail_*.jpg" images/test/depth_trail_%i.jpg**



Mono Depth from Video

root@PHELAN:/jetson-inference/build/aarch64/bin# **./depthnet /dev/video0**





End of HELLO AI WORLD. Next: Jetbot

2022.04.11 ITAI 2374 HCC West Loop

JetBot

Refer back to 2022.02.08 JetBot unboxing experience.

Good starting place is:

https://jetbot.org/master/getting_started.html

*My original SD card image **jetbot-043_nano-4gb-jp45.zip** was still valid.*

Loaded & booted without difficulty in the JetBot.

Problem arose when I couldn't remember the login and password.

D' reminded me it should be "jetbot/jetbot" which worked!

Jetbot booted up to automatically run the Jupyter notebook.

Accessing from the PC was partially successful despite TERRIBLE WiFi at HCC!

Was able to use Basic Motion to move the JetBot appropriately on the floor using WiFe and the Jupyter notebook commands.

Trying to run Teleoperation failed as the camera would not be recognized:

notebooks/teleoperation.ipybk

Create camera instance:

```
from jetbot import Camera

camera = Camera.instance()
```

```
RuntimeErrorTraceback (most recent call last)

/usr/local/lib/python3.6/dist-packages/jetbot-0.4.3-py3.6.egg/jetbot/camera/opencv_gst_camera.py
in __init__(self, *args, **kwargs)

     29             if not re:

---> 30                 raise RuntimeError('Could not read image from camera.')

     31

RuntimeError: Could not read image from camera.

During handling of the above exception, another exception occurred:

RuntimeErrorTraceback (most recent call last)

<ipython-input-4-87257c35b9ce> in <module>

      1 from jetbot import Camera

      2

----> 3 camera = Camera.instance()

/usr/local/lib/python3.6/dist-packages/jetbot-0.4.3-py3.6.egg/jetbot/camera/opencv_gst_camera.py
in instance(*args, **kwargs)

     70     @staticmethod

     71     def instance(*args, **kwargs):

---> 72         return OpenCvGstCamera(*args, **kwargs)

/usr/local/lib/python3.6/dist-packages/jetbot-0.4.3-py3.6.egg/jetbot/camera/opencv_gst_camera.py
in __init__(self, *args, **kwargs)

     35             self.stop()

     36             raise RuntimeError(

---> 37                 'Could not initialize camera.  Please see error trace.')

     38

     39         atexit.register(self.stop)

RuntimeError: Could not initialize camera.  Please see error trace.
```

2022.04.13 Home

Jupyter notebook

/notebooks/basic_motion.jpynb

step [28]

But what if we don't want a bi-directional link, let's say we only want to use the sliders to display the motor values, but not control them. For that we can use the dlink function. The left input is the source and the right input is the target

left_link = traitlets.dlink((robot.left_motor, 'value'), (left_slider, 'value'))

right_link = traitlets.dlink((robot.right_motor, 'value'), (right_slider, 'value'))

Now try moving the sliders. You should see that the robot doesn't respond. But when set the motors using a different method, the sliders will update and display the value!

Wrong! The motors still respond to the sliders.

Also, the motors run too fast.

2022.04.14

Google “NVIDIA Jetbot teleoperation failed to instantiate camera”

https://forums.developer.nvidia.com/t/jetbot-cant-work-with-raspberry-pi-version-2-camera/77014

https://developer.nvidia.com/blog/training-your-jetbot-in-isaac-sim/

https://github.com/NVIDIA-AI-IOT/jetbot/tree/master/docker

JetBot Docker

Quick Start

Step 1 - Configure System

$ cd jetbot

$ ./scripts/configure_jetson.sh

Removed /etc/systemd/system/multi-user..target.wants/nvzramconfig.service

$ cd docker

$ source configure.sh

JETBOT_BASE_IMAGE not found...

$ ./set_nvidia_runtime.sh

If needed, you can also set memory limits on the Jupyter container.

Skip

sudo systemctl enable docker   # enable docker daemon at boot

./enable.sh $HOME   # we'll use home directory as working directory, set this as you please.

JETBOT_BASE_IMAGE not found...

jetbot/docker/camera/

Camera Container

A launch script to publish camera images using ZMQ for access in other containers

cd docker/camera

./build.sh

Manifest for jetbot/jetbot:base-0.4.3-32.5.2 not found...

chandra.kusuma Dec 31

I had the same problem until I found a great article here:

https://developer.nvidia.com/blog/training-your-jetbot-in-isaac-sim/ 8

I followed the instructions and run the camera with the following file after running the enable.sh file inside the docker folder:

./docker/camera/enable.sh

After modifying the import code as follows:

from jetbot.camera.zmq_camera import ZmqCamera as Camera

my Jetbot can now work with the CSI camera flawlessly :)

In notebook [4] changed to

from jetbot.camera.zmq_camera import ZmqCamera as Camera

camera = Camera.instance()

This time program didn't give errors, but no image appeared in the widget window.

$ **sudo init 5** # restart the desktop

$ **sudo systemctl set-default graphical.target**

$ **sudo reboot**

$ jetbot@nano-4gb-jp45:~jetbot$ **./docker/camera/enable.sh**

docker: invalid reference format.

"These must be run on the JetBot directly or through SSH, not from the Jupyter terminal window. If you see docker: invalid reference format, set your environment variables again by calling source configure.sh."
https://developer.nvidia.com/blog/training-your-jetbot-in-isaac-sim/

$ jetbot@nano-4gb-jp45:~jetbot$ **cd docker**

$ jetbot@nano-4gb-jp45:~jetbot/docker$ **source configure.sh**

JETBOT_BASE_IMAGE not found for 32.5.2. Please manually set the

JETBOT_BASE_IMAGE environment variable. (i.e: export JETBOT_BASE_IMAGE=...)

Synchronizing state of docker.service with SysV service script with /lib/systemd/systemd-sysv-install.

Executing: /lib/systemd/systemd-sysv-install enable docker

/Desktop/L4T-Readme/nvidia-l4t-core.dpkg-s.txt

"Version: 32.5.2-20110709090126"

I think I need to rebuild the JetBot OS from scratch to ensure the latest versions of everything....

https://jetbot.org/master/getting_started.html

jetbot-043_nano-4gb-jp45.zip is still valid SD card image

Boot-up: login:jetbot / password:jetbot

Connect to a WiFi network using the following command

$ **sudo nmcli device wifi connect** <SSID> **password** <PASSWORD>

Step 5 - Connect to JetBot from web browser

1 Shutdown JetBot using the command line

```
sudo shutdown now
```

defer already plugged in to battery pack & wall charger

2 Unplug your HDMI monitor, USB keyboard, mouse and power supply from Jetson Nano

defer, may want to use for debugging later

Power the JetBot from the USB battery pack by plugging in the micro-USB cable

defer already plugged in to battery pack & wall charger.

Micro-USB cable has been replaced by GPIO 6-conductor cable.

4 Wait a bit for JetBot to boot

not necessary as didn't reboot

Need to update/upgrade to latest tho directions don’t say so:

jetbot@nano-4gb-jp45:~$ **sudo apt-get update**

...

```
Processing triggers for initramfs-tools (0.130ubuntu3.13) ...

update-initramfs: Generating /boot/initrd.img-4.9.201-tegra

cryptsetup: WARNING: failed to detect canonical device of /dev/root

cryptsetup: WARNING: could not determine root device from /etc/fstab

Warning: couldn't identify filesystem type for fsck hook, ignoring.

/sbin/ldconfig.real: Warning: ignoring configuration file that cannot be
opened: /etc/ld.so.conf.d/aarch64-linux-gnu_EGL.conf: No such file or
directory

/sbin/ldconfig.real: Warning: ignoring configuration file that cannot be
opened: /etc/ld.so.conf.d/aarch64-linux-gnu_GL.conf: No such file or directory

Processing triggers for nvidia-l4t-kernel
(4.9.201-tegra-32.5.2-20210709090126) ...

Errors were encountered while processing:

 nvidia-l4t-bootloader

E: Sub-process /usr/bin/dpkg returned an error code (1)
```

Not sure of the significance of the above errors

5 Check the IP address of your robot on the piOLED display screen. Enter this in place of <jetbot_ip_address> in the next command

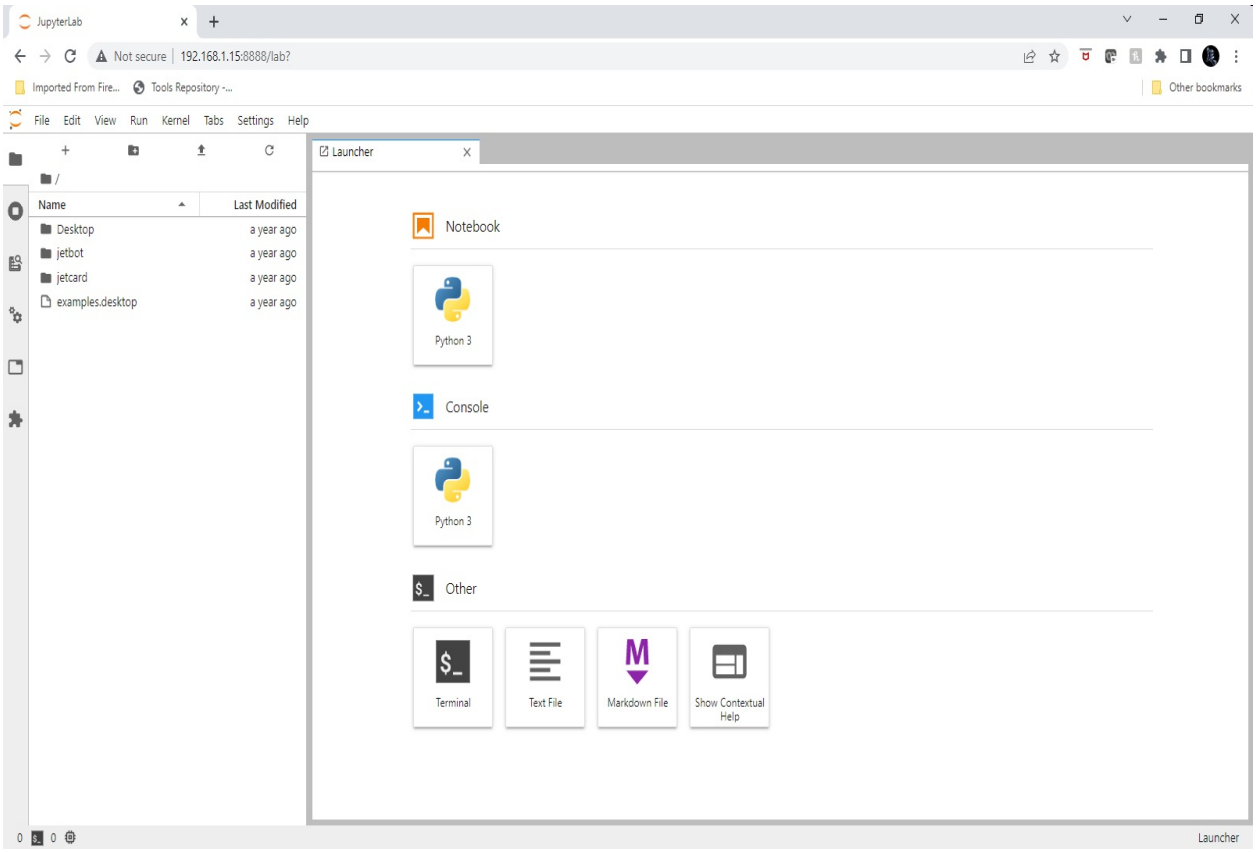
192.168.1.15

6 Navigate to **http://192.168.1.15:8888** from your desktop's web browser. You can do this from any machine on your local network.

Be sure not to use https:

7 Sign in using the password jetbot.

Done



Software Setup (Docker)

Please note, the JetBot containers described in this page currently target a Jetson Nano SD card image flashed with **JetPack 4.4.** These containers will not work with other version of JetPack.

Mine is:

JetPack **4.5** (**L4T R32.5.0**) l4t-ml:r32.5.0-py3

perhaps git clone will correct this??

$ **sudo init 5** # restart the desktop *for debug*

not working

$ **sudo systemctl set-default graphical.target**

$ **sudo reboot**

system locks up. Start over. :-/

System program problem detected

This is so annoying! Fix is here:

https://askubuntu.com/questions/1160113/system-program-problem-detected

How to get rid of it ?

Depends on what you call "get rid". The ideal fix would be to check what is inside the reports, and try and find a fix for it. If the package it is about is unneeded or benign you could also purge it. Most times it is a core functionality though.

If you can not understand those crash reports most times you can google the error notice (there will always be one in there). Or drop a message in chat. Generally crashes are off topic on AU as those are bugs and would need to be reported (through this service ;)).

You can pick any of these to remove the crash report up to actually removing the package (would be rather ironic if the error comes from apport itself):

sudo rm /var/crash/* will delete old crashes and stop informing you about them until some package crashes again.

You can stop the service with **sudo systemctl disable apport** (and enable it again with sudo systemctl enable apport)

If you do not want to see crash reports you can disable it by doing sudo nano **/etc/default/apport** and changing enabled=1 to **enabled=0**. Editing it in reverse will enable it again.

You can delete the service with ***sudo apt purge apport*** (and install it again with sudo apt install apport) *[will skip this one]*

Had to install nano:

$ **sudo apt-get install nano**

installed ok, but during this I get the error:

Errors were encountered while processing: nvidia-l4t-bootloader

Hmmm?

Perhaps because JetBot seem to be configured to auto-launch Jupyter notebook?

Software Updater is suggesting installing several NVIDIA L4T updates. Maybe this will fix the issue? Install!

Package operation failed.

The installation or removal of a software package failed.

So much for that hope!

The computer needs to restart to finish installing updates. **REBOOT**

Software Updater

Failed to download repository information.

Check your internet connection *[it’s fine]*

$ **sudo apt-get update**

multiple instances of:

...is configured multiple times in /etc/apt/sources.list:8 and :51

$ **sudo apt-get upgrade**

again errors w/ nvidia-l4t-bootloader

Try updating from w/in the Docker but, hmmm, not sure how to do that...?

2022.04.16

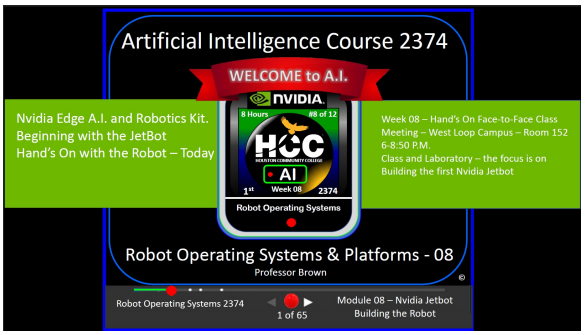
Let's take it from the class lecture:

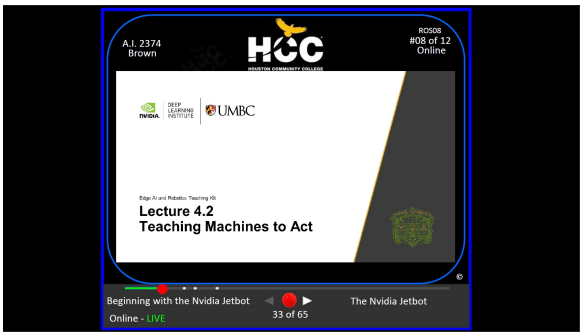
ITAI 2374 Robot OS & Platforms 08 B Spr 22 Final.pdf

Which links to:

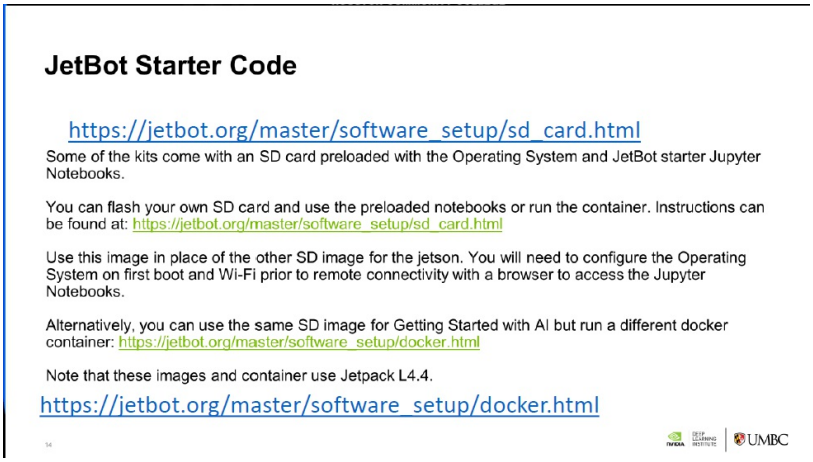
https://github.com/dusty-nv/jetson-inference#hello-ai-world

Then on to slide 33/65 Lecture 4.2 Teaching Machines to Act:





and skip down to slide 46/65 re the code:



The kicker:

"Note that these images and container use Jetpack L4.4"

The current Jetpack is L4.5. Will have to regress the Jetpack

which takes you to:

https://jetbot.org/master/software_setup/sd_card.html

Scroll down to:

Old releases

Platform	JetPack Version	JetBot Version	Download
Jetson Nano 2GB	4.4.1	0.4.2	jetbot-042_nano-2gb-jp441.zip
Jetson Nano (4GB)	4.4.1	0.4.2	jetbot-042_nano-4gb-jp441.zip
Jetson Nano 2GB	4.4.1	0.4.1	jetbot-041_nano-2gb-jp441.zip
Jetson Nano (4GB)	4.4.1	0.4.1	jetbot-041_nano-4gb-jp441.zip
Jetson Nano (4GB)	4.3	0.4.0	jetbot_image_v0p4p0.zip
Jetson Nano (4GB)	4.2	0.3.2	jetbot_image_v0p3p2.zip

and choose the most recent release for your JetBot 2GB or 4GB. In my case

Jetson Nano (4GB) 4.4.1 0.4.2
jetbot-042_nano-4gb-jp441.zip *[I hope!]*

SD Card Formatter

Balena Etcher flash to SD card

Boot on JetBot login: **jetbot** / password: **jetbot**

Step 1 - Setup Jetson Nano

$ **sudo init 5** *[to enable GUI to aid in setup]*

enable WiFi

disable lock screen

Software Updater *[all x- Thunderbird Mail]*

extensive update

Crash report: Package: nvidia-l4t-bootloader 32.4.4-.....

This is 3d party package so may be ok.

Hopefully to be fixed from th git clone?

Recheck for updates - needs to restart to finish. **Reboot.**

Next step:

https://jetbot.org/master/software_setup/docker.html

Using Docker Container

Step 1 - Setup Jetson Nano *done*

Step 2 - Connect to Wi-Fi *done*

Step 3 - Clone JetBot repo

jetbot@nano-4gb-jp441:~$ **git clone http://github.com/NVIDIA-AI-IOT/jetbot.git**

fatal: destination path 'jetbot' already exists and is not an empty directory.

It's already there

Step 4 - Configure System

jetbot@nano-4gb-jp441:~$ **cd jetbot**

jetbot@nano-4gb-jp441:~/jetbot$ **./scripts/configure_jetson.sh**

Removed /etc/systemd/system/multi-user.target.wants/nvzramconfig.service.

Re-enable GUI on boot

jetbot@nano-4gb-jp441:~/jetbot$ **sudo systemctl set-default graphical.target**

```
Removed /etc/systemd/system/default.target.

Created symlink /etc/systemd/system/default.target → /lib/systemd/system/graphical.target.
```

Enable swap

jetbot@nano-4gb-jp441:~/jetbot$ **./scripts/enable_swap.sh**

```
### You have swap memory space, but not big enough.

Setting up swapspace version 1, size = 4 GiB (4294963200 bytes)

no label, UUID=655dc05a-2291-4ea5-83c4-2b1765bb7f6a

# /etc/fstab: static file system information.

#

# These are the filesystems that are always mounted on boot, you can

# override any of these by copying the appropriate line from this file into

# /etc/fstab and tweaking it as you see fit.  See fstab(5).

#

# <file system> <mount point>             <type>          <options>
                  <dump> <pass>

/dev/root            /                       ext4           defaults
                        0 1

/swfile none swap sw 0 0

jetbot@nano-4gb-jp441:~/jetbot$ free -m

              total        used        free      shared  buff/cache
available

Mem:           3956        1539        1076          37        1340
2219

Swap:          6074           0        6074
```

Step 5 - Enable all containers

jetbot@nano-4gb-jp441:~/jetbot$ **cd docker**

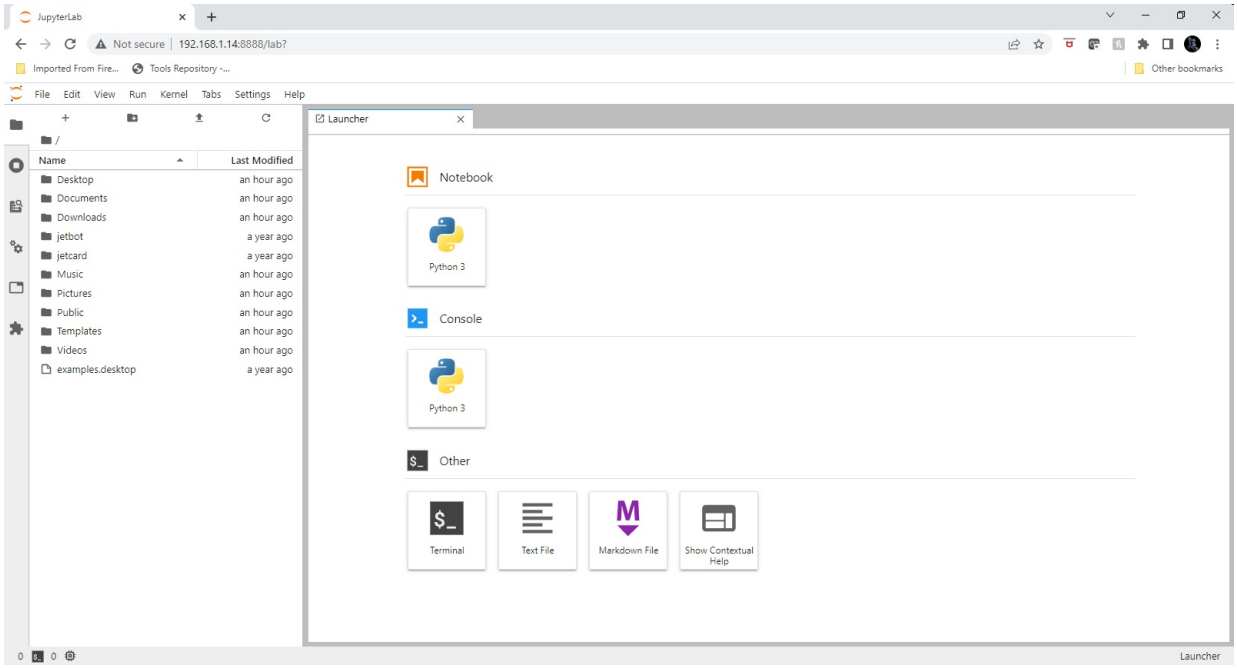
jetbot@nano-4gb-jp441:~/jetbot/docker$ **./enable.sh $HOME**

```
Synchronizing state of docker.service with SysV service script with
/lib/systemd/systemd-sysv-install.

Executing: /lib/systemd/systemd-sysv-install enable docker

docker: Error response from daemon: Conflict. The container name
"/jetbot_display" is already in use by container
"daae838838e3aa0ec8207a4a140ae47b6c9b0a8c00ddc9b3e5c6b0afe310153c". You have
to remove (or rename) that container to be able to reuse that name.

See 'docker run --help'.

docker: Error response from daemon: Conflict. The container name
"/jetbot_jupyter" is already in use by container
"56ab2923e573a7d2f9915e5b1dea18ca98d200500c30a2e6b1dac026e9e252dc". You have
to remove (or rename) that container to be able to reuse that name.

See 'docker run --help'.
```

Now you can go to https://<jetbot_ip>:8888 from a web browser and start programming JetBot!

You can do this from any machine on your local network. The password to log in is jetbot.

http://192.168.1.14:8888 *[WiFi]*



On to next web page...

Basic Motion

https://jetbot.org/master/examples/basic_motion.html

4. Open and follow the basic_motion.ipynb notebook

all work well as indicated.

On heartbeat slider, motor stops at **0.06** *[WiFi]*

Restart kernel & clear all outputs

Shut down all kernels

Close Basic Motion tab

Teleoperation

https://jetbot.org/master/examples/teleoperation.html

open /.../notebooks/teleoperation/teleoperation.ipnb

Gamepad tester

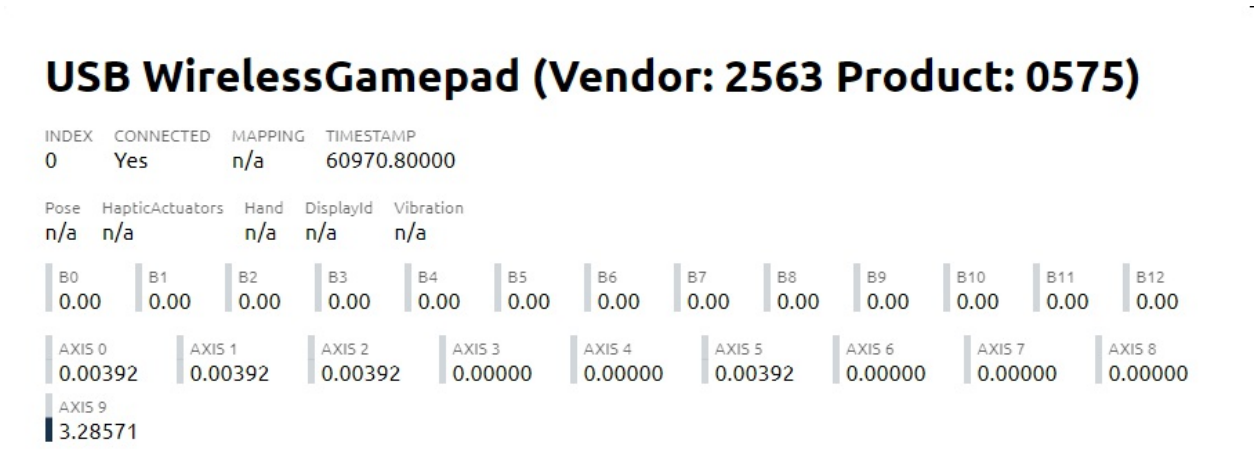
https://gamepad-tester.com/

Put receiver dongle from gamepad controller into USB port ON THE CONTROLLING PC, not on the JetBot!

Place batteries into gamepad controller.

Turn gamepad controller ON.

Push the HOME button so there are TWO red lights on the display.



working

change statement below to the indicated axes:

```
left_link = traitlets.dlink((controller.axes[1], 'value'), (robot.left_motor,
'value'), transform=lambda x: -x)

right_link = traitlets.dlink((controller.axes[5], 'value'),
(robot.right_motor, 'value'), transform=lambda x: -x)
```

this makes the left joystick run the left motor and the right joystick run the right motor. This makes effectively a "differential steering" as opposed to interpreting the direction of a single joystick into appropriate motor commands.

It works

Create camera instance

```
from jetbot import Camera

camera = Camera.instance()
```

```
RuntimeErrorTraceback (most recent call last)

/usr/local/lib/python3.6/dist-packages/jetbot-0.4.1-py3.6.egg/jetbot/camera/op
encv_gst_camera.py in __init__(self, *args, **kwargs)

     29             if not re:

---> 30                 raise RuntimeError('Could not read image from
camera.')

     31

RuntimeError: Could not read image from camera.

During handling of the above exception, another exception occurred:

RuntimeErrorTraceback (most recent call last)

<ipython-input-4-87257c35b9ce> in <module>

      1 from jetbot import Camera

      2

----> 3 camera = Camera.instance()

/usr/local/lib/python3.6/dist-packages/jetbot-0.4.1-py3.6.egg/jetbot/camera/op
encv_gst_camera.py in instance(*args, **kwargs)

     70     @staticmethod

     71     def instance(*args, **kwargs):

---> 72         return OpenCvGstCamera(*args, **kwargs)

/usr/local/lib/python3.6/dist-packages/jetbot-0.4.1-py3.6.egg/jetbot/camera/op
encv_gst_camera.py in __init__(self, *args, **kwargs)
```

```
     35             self.stop()
     36             raise RuntimeError(
---> 37                 'Could not initialize camera.  Please see error
trace.')
     38
     39         atexit.register(self.stop)
RuntimeError: Could not initialize camera.  Please see error trace.
```

There is no such directory:

/usr/local/lib/python3.6/dist-packages/jetbot-0.4.1-py3.6.egg/jetbot/camera/opencv_gst_camera.py

On advice from:

https://github.com/NVIDIA-AI-IOT/jetbot/issues/306

jetbot@nano-4gb-jp441:~/jetbot/docker$ **./disable.sh**

```
[sudo] password for jetbot:
Error response from daemon: No such container: jetbot_camera
Error: No such container: jetbot_camera
jetbot_jupyter
jetbot_jupyter
jetbot_display
jetbot_display
```

jetbot@nano-4gb-jp441:~/jetbot/docker$ **JETBOT_CAMERA=zmq_camera ./enable.sh $HOME**

```
Synchronizing state of docker.service with SysV service script with
/lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable docker
df0cd70ca9d9a02f90cb0e46d931a67d68bfbb6f5cfc9ec917533bb55abbddbe
WARNING: Published ports are discarded when using host network mode
92420f30f190a98cbf581f59c4ab43e692cb1d5a8a459891a6afce9665d58d96
```

Replace the camera code in notebooks with the following. It should behave similarily, but will internally subscribe to camera messages from the publisher.

```
from jetbot.camera.zmq_camera import ZmqCamera
camera = ZmqCamera()
```

This time no errors, but image is just black square.

The JetBot instructions REALLY SHOULD START WITH A CAMERA CHECK and troubleshooting guide!

Hello, regarding syslog, you could try:

However, to prevent syslog build-up at the nano (can reach Max SD card space e.g. 35 GiB) deactivate rsyslog service with

```
$sudo service rsyslog stop

$sudo systemctl disable rsyslog.service

$cd /var/log

$sudo rm syslog

$sudo truncate –s 0 *

truncate: cannot open 'apt' for writing: Is a directory

truncate: cannot open 'dist-upgrade' for writing: Is a directory

truncate: cannot open 'gdm3' for writing: Is a directory

truncate: cannot open 'installer' for writing: Is a directory

$sudo systemctl status rsyslog
```

Didn’t work. Still black image.

jetbot@nano-4gb-jp441:~/jetbot$ **sudo systemctl restart nvargus-daemon**

Didn’t work. Still black image.

2022.04.18 HCC ITAI

Got SD card image from D’ w/ JetPack 4.5

Suggested visiting https://ngc.nvidia.com/signin

Booted

enabled GUI, *not helpful as system config unavailable! [disable later]*

enabled WiFi from command line

$ **sudo apt-get update**

$ **sudo apt-get upgrade**

Per D’ the CURRENT image should be the one I’ve done before:

jetbot-043_nano-4gb-jp45.zip

fumbled around and still couldn’t get the camera to work.

Go home and try to sort it out again from scratch.

2022.04.19 Home

JetBot

STARTING OVER!

Let's go back to what seemed to at least partly work before,

*p 33 above, which is here since I have the **Waveshare 4GB JetBot**:*

JetBot AI Kit

https://www.waveshare.com/wiki/JetBot_AI_Kit

Which directs you here for software based on **JetPack 4.3**.

It doesn't distinguish 2GB or 4GB. Will this be a problem?

jetbot_image_v0p4p0.zip (drive.google.com/...)

SD Card Formatter: **Overwrite** for superclean SD card

Step 1. Write JetBot image to SD card

Balena Etcher flash above .zip file

Step 2. Startup Jetson Nano Developer Kit

Boot on JetBot: logon: jetbot / password: jetbot

Step 3. Connect Jetbot to WIFI using GUI [or you can...]

jetbot@nano-4gb-jp411:$ **sudo nmcli device wifi connect** <SSID> **password** <PASSWORD>

Step 4 - Connect to JetBot from web browser

open browser window on PC.

http://192.168.1.14:8888

Jupyter notebook opens. Password: **jetbot**

Step 5 - Install latest software (optional)

1. If you haven't already, connect to your robot by going to http://<jetbot_ip_address>:8888 [already done]

2. Click the + icon to open the Jupyter Lab launcher

3. Launch a new terminal (from w/in notebook)

4. Get and install the latest JetBot repository from GitHub by entering the following commands:

jetbot@jetson-4-3:~$ **git clone https://github.com/NVIDIA-AI-IOT/jetbot**

jetbot@jetson-4-3:~$ **cd jetbot**

jetbot@jetson-4-3:~/jetbot$ **sudo python3 setup.py install**

significant install

5. Replace the old notebooks with the new notebooks by entering

jetbot@jetson-4-3:~/jetbot$ **sudo apt-get install rsync**

jetbot@jetson-4-3:~/jetbot$ **rsync jetbot/notebooks ~/Notebooks**

rsync: link_stat "/home/jetbot/jetbot/jetbot/notebooks" failed: No such file or directory (2)

rsync error: some files/attrs were not transferred (see previous errors) (code 23) at main.c(1196) [sender=3.1.2]

jetbot@jetson-4-3:~/jetbot$ **ls**

assets CHANGELOG.md CMakeFiles CMakeLists.txt dist docs jetbot.egg-info Makefile notebooks scripts

build CMakeCache.txt cmake_install.cmake CONTRIBUTING.md docker jetbot LICENSE.md mkdocs.yml README.md setup.py

jetbot@jetson-4-3:~/jetbot$ **cd notebooks/**

jetbot@jetson-4-3:~/jetbot/notebooks$ **ls**

basic_motion collision_avoidance object_following road_following teleoperation

jetbot@jetson-4-3:~/jetbot/notebooks$ **rsync ~/jetbot/notebooks ~/Notebooks**

skipping directory notebooks

[does this mean it worked??]

Step 6 - Configure power mode

```
jetbot@jetson-4-3:~/jetbot$ sudo nvpmodel -m1
jetbot@jetson-4-3:~/jetbot$ sudo nvpmodel -q
NVPM WARN: fan mode is not set!     [ignore this - only set for Xavier]
NV Power Mode: 5W
1
```

Next Follow the examples:

https://github.com/NVIDIA-AI-IOT/jetbot/wiki/examples

Basic motion

Notebook works appropriately

Teleoperation

Be sure gamepad receiver dongle is in the computer running the Jupyter notebook. *[This is usually the PC but may be run on the nano using http://localhost:8888.]*

Click the link to the gamepad test site:

https://gamepad-tester.com/

May have to hold down HOME button 7 seconds to get into Xbox mode that displays 2 LEDs.

Check to see whether the “index” [first parameter in grid] is 0 or 1. Edit the notebook statement accordingly:

```
controller = widgets.Controller(index=0)  # replace with index of your controller
```

Run the statement and observe the sliders as you manipulate the gamepad to see which axis numbers correspond to each joystick. In my case it’s 1 and 5 instead of default 1 and 3.

```
from jetbot import Robot

import traitlets

robot = Robot()

left_link = traitlets.dlink((controller.axes[1], 'value'), (robot.left_motor, 'value'), transform=lambda x: -x)

right_link = traitlets.dlink((controller.axes[5], 'value'), (robot.right_motor, 'value'), transform=lambda x: -x)
```

produces error:

```
Error loading module `ublox_gps`: No module named 'serial'

image = widgets.Image(format='jpeg', width=300, height=300)

display(image)
```

Fails to create a box for an image.

```
from jetbot import Camera

camera = Camera.instance()
```

does not error

```
from jetbot import bgr8_to_jpeg

camera_link = traitlets.dlink((camera, 'value'), (image, 'value'),
transform=bgr8_to_jpeg)
```

fails to display a video.

This notebook needs a configure camera notebook first!

Try this from advice elsewhere:

```
    from jetbot.camera.zmq_camera import ZmqCamera

    camera = ZmqCamera()
```

doesn't help

Tomorrow let's try the intro camera app from computer vision

Maybe update/upgrade first:

jetbot@jetson-4-3:~$ **sudo apt-get update**

jetbot@jetson-4-3:~$ **sudo apt-get upgrade**

2022.04.21

Software Updater offered 29mb of update. OK. Reboot.

Changed code below according to advice to slow speed:

```
left_link = traitlets.dlink((controller.axes[1], 'value'), (robot.left_motor,
'value'), transform=lambda x: - 0.5 * x)

right_link = traitlets.dlink((controller.axes[5], 'value'),
(robot.right_motor, 'value'), transform=lambda x: - 0.5 * x)
```

but now get error

```
Error loading module `ublox_gps`: No module named 'serial'
```

Changing back didn't fix it. But motors run anyway!

Speed reduction works best at **x: -0.3*x)**

```
from jetbot.camera.zmq_camera import ZmqCamera

camera = ZmqCamera()
```

Still no image, nor with

```
from jetbot import Camera

camera = Camera.instance()
```

2022.04.23

USAi Labs meeting. D' advised that Dockers require JetPack 4.4 (Not 4.3 or 4.5) & will send a link to working images for JetBot and Issac Sim.

JetBot Fan

Note that fan doesn't run on JetBot. Info elsewhere says fan mode not defined for Jetson, just Xavier.

Google "**waveshare jetbot fan not working**":

https://www.waveshare.com/wiki/Fan-4020-PWM-5V

Running fan: you can use the command below to run the fan in full speed

jetbot@jetson-4-3:~$ **sudo sh -c 'echo 255 > /sys/devices/pwm-fan/target_pwm'**

It runs!

Note You can change the 255 to other value (0~255) to change the speed of fan

You can also set the fan ran when booting by creating and modifying rc.local file:

sudo ~~vi~~ nano /etc/rc.local

Add statements below to rc.local file and save

```
#!/bin/bash
sleep 10
sudo /usr/bin/jetson_clocks
sudo sh -c 'echo 255 > /sys/devices/pwm-fan/target_pwm'
```

Modify file permission

```
sudo chmod u+x rc.local
```

Then reboot and testing

```
sudo reboot
```

Question:

What functions do the four wires of Fan-4020-PWM-5V separately?

Answer:

Black -> GND; Red -> Vcc; Blue -> Tach; Yellow -> PWM

The color of Tach and PWM may be different among different batch, it is all to the actual products.

See also
https://forums.developer.nvidia.com/t/jetson-nano-fan/72469

Link from Saturday Meeting

G Brown, James Phelan

6222-ITAI-2374-Robot Operating Sys & Platform-RT-22680

April 23, 2022 at 2:57pm

Hi, Jim.

This page is the most helpful and I would consider it your true page for the Jetbot Process

https://github.com/NVIDIA-AI-IOT/jetbot/wiki (and go to software setup) These five steps are crucial.

I am certain you already know this.

https://github.com/NVIDIA-AI-IOT/jetbot/wiki/Examples (I am also sure you have already been here as well) - these examples are very helpful.

This next link shows the different files available for 4GB

https://jetbot.org/master/software_setup/sd_card which shows if you scroll down the 4.4 for the 4GB

At the top you will see the 4.5 for the 4GB Nano which will run everything just fine but not docker which you can assemble from the source if you like.

There is one more link: https://github.com/dusty-nv/jetbot_ros (this i very helpful to see how he calls on the containers.

To be complete: Jetson hacks makes good reference in their Video Software setup on Videos 2 and 3

https://www.youtube.com/watch?v=MY_Fe7EN6ro

Sorry, it is becoming clear code is changing all the time and these videos and githubs can barely keep it all straight. While this is a lot of data - this seems to be the best. One more tidbit is the use of the EdiMax WifFi dongle (please, please, please use the EdiMax WiFi Dongle for best results as well as Putty for SSH into your bot for this set of Jupyter Notebook Operations. Hope this helps. Holler if you need anything else.

Most especially, Very carefully read those five steps - there is a lot of information packed into those five steps. Take care. Thanks for all you do and I hope in some small way this helps you.

kindest regards.

-GRB-

2022.04.24

Following above from R&D:

https://github.com/NVIDIA-AI-IOT/jetbot/wiki

> Software Setup

Step 1 - Flash JetBot image onto SD card

Download the expandable JetBot SD card image
jetbot_image_v0p4p0.zip *[different from other 3 failed images!]*

[Don't get fooled by reference to older image v0p3 and download that my mistake!]

Using Etcher, select the jetbot_image_v0p4p0.zip image and flash it onto the SD card

Step 2 - Boot Jetson Nano

Step 3 - Connect JetBot to WiFi

Log in using the user **jetbot** and password **jetbot**

Connect to a WiFi network using the Ubuntu desktop GUI

Your Jetson Nano should now automatically connect to the WiFi at boot and display it's IP address on the piOLED display. ✔

Step 4 - Connect to JetBot from web browser

Per D' advice, connect from browser on JetBot using localhost:

Navigate to **http://localshost:8888** from your ~~desktop's~~ JetBot's web browser.

Step 5 - Install latest software ~~(optional)~~

Click the + icon to open the Jupyter Lab launcher

Launch a new terminal

Get and install the latest JetBot repository from GitHub by entering the following commands

```
git clone https://github.com/NVIDIA-AI-IOT/jetbot
cd jetbot
sudo python3 setup.py install
```

jetbot@jetson-4-3:~$ **git clone https://github.com/NVIDIA-AI-IOT/jetbot**

done

jetbot@jetson-4-3:~$ **cd jetbot**

jetbot@jetson-4-3:~/jetbot$ **sudo python3 setup.py install**

done

Replace the old notebooks with the new notebooks by entering

```
sudo apt-get install rsync
rsync jetbot/notebooks ~/Notebooks
```

jetbot@jetson-4-3:~/jetbot$ **sudo apt-get install rsync**

done

jetbot@jetson-4-3:~/jetbot$ **rsync jetbot/notebooks ~/Notebooks**

```
rsync: link_stat "/home/jetbot/jetbot/jetbot/notebooks" failed: No such file or directory (2)
rsync error: some files/attrs were not transferred (see previous errors) (code 23) at main.c(1196) [sender=3.1.2]
```

Try:

jetbot@jetson-4-3:~/jetbot$ **cd ..**

jetbot@jetson-4-3:~$ **rsync jetbot/notebooks ~/Notebooks**

```
skipping directory notebooks
```

jetbot@jetson-4-3:~$ **rsync jetbot/ ~/Notebooks**

```
skipping directory .
```

jetbot@jetson-4-3:~$ **rsync jetbot ~/Notebooks**

```
skipping directory jetbot
```

jetbot@jetson-4-3:~$ **rsync ~/Notebooks**

```
drwxr-xr-x          4,096 2020/01/15 19:19:46 Notebooks
```

Sent text to R&D about this

Step 6 - Configure power mode

This can be done easier by selecting NVIDIA symbol on top bar, dropping menu, Power mode > and selecting 1:5W

Select 5W power mode

```
sudo nvpmodel -m1
```

Verify the Jetson Nano is in 5W power mode

```
sudo nvpmodel -q
```

Also, apparently there's no fan mode for the Nano (just Xavier). See yesterday's notes p138 or try

jetbot@jetson-4-3:~$ **sudo /usr/bin/jetson_clocks**

fan started!

Next

Follow the examples:

https://github.com/NVIDIA-AI-IOT/jetbot/wiki/examples

Again, per D's advice, run this from http://localhost:8888 on the **JetBot**.

Example 1 - Basic Motion

in section that says:

The ``link`` function that we created above actually creates a bi-directional link! That means,

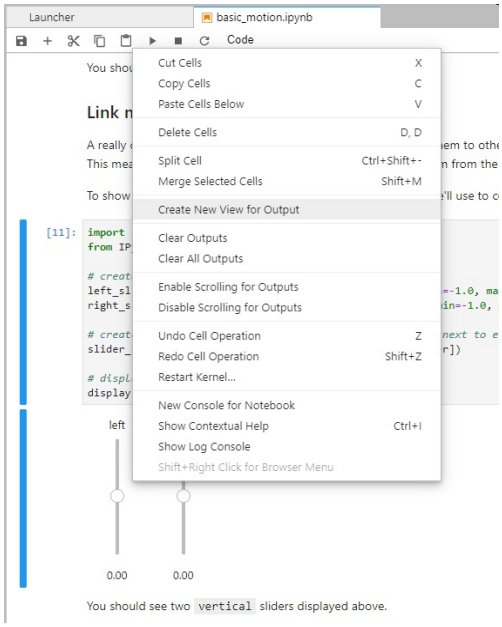
if we set the motor values elsewhere, the sliders will update! Try executing the code block below

You can't see the sliders from the part of the notebook that executes the code. You need to do a trick, described elsewhere n the notebook:

right-click the slider section and choose

"Create New View for Output"

This will put the sliders in a new window that won't move with the notebook.



After fooling around with Basic Motion for a while from both the JetBot and a PC browser (to help document), I ran into this error when restarting the notebook from the JetBot (but not the PC):

```
Error loading module 'ublox_gps': No module named 'serial'
```

Google not helpful in this.

I've run into this before with this notebook. I suspect a conflict created with changing, then restarting code, or with 2 versions of the same notebook open at once. The system freq asked whether I wanted to overwright or revert to changes on disk. I always chose "cancel". Perhaps rsync will fix this?

jetbot@jetson-4-3:~$ **rsync ~/Notebooks**

```
drwxr-xr-x         4,096 2020/01/15 19:19:46 Notebooks
```

Nope!

Need to reinstall from git-clone?

Need to delete destination path 'jetbot'

$ **sudo rm -rf jetbot**

jetbot@jetson-4-3:~$ **git clone https://github.com/NVIDIA-AI-IOT/jetbot**

jetbot@jetson-4-3:~$ **cd jetbot**

jetbot@jetson-4-3:~/jetbot$ **sudo python3 setup.py install**

jetbot@jetson-4-3:~/jetbot$ **cd**

jetbot@jetson-4-3:~$ **rsync ~/Notebooks**

Close & reopen browser / noteboot to be sure...

That seemed to fix it!

Now on to teleoperation....

Example 2 - Teleoperation

https://gamepad-tester.com/

May have to hold down HOME button 7 seconds to get into Xbox mode that displays 2 LEDs. *This is not working now!* Turning off, removing batteries doesn't restore this function. But is DOES if I plug the receiver dongle into the PC!

Anyway, single LED ShanWan USB Wireless Gamepad mode works ok.

If I assign left-link to axes[**1**] and right-link to axes[**0**] then, if I turn it 45o to the right, it acts like a regular joystick up/down right/left!

Get `Error loading module 'ublox_gps': No module named 'serial'`

but this doesn't apparently harm the motion control.

"Create and display Image widget" fails to produce error or image frame, just a placeholder for the frame.

"Create camera instance" no errors

"Connect Camera to Image widget" fails to display video

Google: "waveshare jetbot test camera"

https://www.waveshare.com/wiki/IMX219-160_Camera

$ **nvgstcapture-1.0 --sensor-id=0**

does ok til end, then:

```
Error generated: /dvs/git/dirty/git-master_linux/multimedia/nvgtstreamer/gst-nvarguscamera/gstnvarguscamerasrc.cpp, execute:543 Failed to create CaptureSession
```

$ **nvgstcapture-1.0**

same

$ **nvgstcapture-1.0 --sensor-id=1**

same

Waveshare CSI camera failed but picamera succeeded in hello_camera

Going back to JetBot SD card, basic_motion worked ok but still get

`Error loading module 'ublox_gps': No module named 'serial'`

Teleoperation

widgets.Controller(index=**1**)

from jetbot import Robot...

Error loading module `ublox_gps`: No module named 'serial'

image = widgets.Image(...

only creates placeholder icon, no image box

from jetbot import Camera

causes JetBot to reboot after about a minute withour error message

GIVE UP ON THIS FOR NOW

2022.04.25 HCC ITAI class

Not much accomplished on JetBot.

Instead a *FASTINATING* visit from 2 NVIDIA representatives:

Reynaldo Gomez, Energy Partner Ecosystem Manager and

Kenneth Hester, Solutions Architect Manager Gloval Energy Sector

regarding interesting AI challenges and solutions experienced by NVIDIA and advice on job seeking.

2022.04.29 HCC West Houston Institute

“Converence on Artificial Intelligence & The Future of Houston”

2022.04.30 [passing on Industry day at conference]

Download HCC ITAI class slideshow ppt:

ITAI 2374 Robot OS & Platforms 09 & 10 B Spr 22 Final .pdf

Slide 2/55 links to Dusty’s Inference Learning github:

[I guess this mean we build on this older SD image?]

NO! SKIP DOWN FEW PAGES TO GREEN

https://github.com/dusty-nv/jetson-inference#hello-ai-world

Slide 8/55 Why ROS2 links to ROS2 documentation:

http://docs.ros.org/en/rolling/

Slide 9/55 JetBot ROS (jetbot_ros) *[NOW we’re getting down to it!]*

https://github.com/dusty-nv/jetbot_ros

Download **jetson-nano-jp461-sd-card-image.zip** from links in 2/55

Flash to formatted SD card using Balena Etcher

Boot on Nano

Accept licenses

Local configuration

computer name **JetBotROS**

username **ubuntu**

password **ubuntu**

log in automatically

default partition size max

Nvpmodel Mode = **5V**

System reboots /p config

configure local **WiFi**

disable **lock screen**

Software Updater: **install 692 kB**

ubuntu@JetBotROS:~$ **sudo apt-get update** *[to cover all bases]*

ubuntu@JetBotROS:~$ **sudo apt-get upgrade**

finds add'l stuff to update/upgrade

Now from https://github.com/dusty-nv/jetbot_ros

jetbot_ros

Start the JetBot ROS2 Foxy container

git clone https://github.com/dusty-nv/jetbot_ros

cd jetbot_ros

docker/run.sh

ubuntu@JetBotROS:~$ **git clone https://github.com/dusty-nv/jetbot_ros**

```
Cloning into 'jetbot_ros'...
remote: Enumerating objects: 683, done.
remote: Counting objects: 100% (146/146), done.
remote: Compressing objects: 100% (71/71), done.
remote: Total 683 (delta 85), reused 81 (delta 75), pack-reused 537
Receiving objects: 100% (683/683), 11.19 MiB | 7.75 MiB/s, done.
Resolving deltas: 100% (389/389), done.
```

ubuntu@JetBotROS:~$ **cd jetbot_ros/**

ubuntu@JetBotROS:~/jetbot_ros$ **docker/run.sh**

```
reading L4T version from /etc/nv_tegra_release
L4T BSP Version:  L4T R32.7.1
[sudo] password for ubuntu:
CONTAINER:     dustynv/jetbot_ros:foxy-r32.7.1
DEV_VOLUME:
DATA_VOLUME:   --volume /home/ubuntu/jetbot_ros/data:/workspace/src/jetbot_ros/data
USER_VOLUME:
USER_COMMAND:
```

V4L2_DEVICES: --device /dev/video0

xhost: unable to open display ""   [can’t open from PuTTY window]

xauth: file /tmp/.docker.xauth does not exist

xauth: file /home/ubuntu/.Xauthority does not exist

xauth: (argv):1: unable to read any entries from file "(stdin)"

*chmod: cannot access '/tmp/.docker.xauth': **No such file or directory***

*Unable to find image '**dustynv/jetbot_ros:foxy-r32.7.1**' locally*

docker: Error response from daemon: manifest for dustynv/jetbot_ros:foxy-r32.7.1 not found: manifest unknown: manifest unknown.

See 'docker run --help'.

L4T-README

version

nvidia-l4t-core.dpkg-s.txt

Version: 32.7.1-20220219090432

$ **sudo find / -name foxy-r32.7.1**

negative results

2022.05.01

Let's start again from here: **[Dead end. Skip down to green]**

https://github.com/dusty-nv/jetbot_ros

clearly "you can't get there from here, you have to go somewhere else first" so back to the beginning where, hopefully, I just skipped something somewhere:

https://github.com/dusty-nv/jetson-inference#hello-ai-world

then on to

https://github.com/dusty-nv/jetson-inference/blob/master/docs/jetpack-setup-2.md

which points to

https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit

and step through to

https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit#write

to download the SD card image

https://developer.nvidia.com/jetson-nano-sd-card-image

flash the card and boot the Nano in the usual fashion. Then

https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit#setup

Do initial configuration then on to the JetBot link:

https://github.com/NVIDIA-AI-IOT/jetbot

It seems to dead-end here, but click on the second line docker

https://github.com/NVIDIA-AI-IOT/jetbot/tree/master/docker

This I believe is the missing link! [It isn't!]

JetBot Docker

Quick Start

Step 1 - Configure System

First, call the scripts/configure_jetson.sh script to configure the power mode and other parameters.

```
cd jetbot
./scripts/configure_jetson.sh
```

Well, here there's no jetbot, just jetbot_ros, which has a scrips directory. Try that.

ubuntu@JetBotROS:~$ **cd jetbot_ros/**

ubuntu@JetBotROS:~/jetbot_ros$ **cd scripts/**

ubuntu@JetBotROS:~/jetbot_ros/scripts$ **ls**

ros_entrypoint.sh setup_workspace.sh

No configure_jetson.sh. Think we're going off the rails here. Retreat!

Go back to earlier advice:

2022.04.23

USAi Labs meeting. D' advised that Dockers require JetPack 4.4 (Not 4.3 or 4.5) & will send a link to working images for JetBot and Issac Sim.

And reflash SD card with THIS image: **jetbot_image_v0p4p0.zip**

username: **jetbot** / password: **jetbot**

local config, WiFi, 5W power mode.

Decline 18.0 > 20.0 ubuntu upgrade.

Software Updater 730.9 MB download. Restart.

$ sudo update/upgrade

Slide 12/55:

https://github.com/dusty-nv/jetbot_ros

jetbot_ros / Start the JetBot ROS2 Foxy container

```
git clone https://github.com/dusty-nv/jetbot_ros
cd jetbot_ros
docker/run.sh
```

jetbot@jetson-4-3:~$ **git clone https://github.com/dusty-nv/jetbot_ros**

jetbot@jetson-4-3:~$ **cd jetbot_ros**

jetbot@jetson-4-3:~/jetbot_ros$ **docker/run.sh**

```
reading L4T version from /etc/nv_tegra_release
L4T BSP Version:  L4T R32.3.1
CONTAINER:     dustynv/jetbot_ros:foxy-r32.3.1
DEV_VOLUME:
DATA_VOLUME:   --volume /home/jetbot/jetbot_ros/data:/workspace/src/jetbot_ros/data
USER_VOLUME:
USER_COMMAND:
V4L2_DEVICES:    --device /dev/video0
xhost:  unable to open display ""
xauth:  file /tmp/.docker.xauth does not exist
Unable to find image 'dustynv/jetbot_ros:foxy-r32.3.1' locally
docker: Error response from daemon: manifest for dustynv/jetbot_ros:foxy-r32.3.1 not found: manifest unknown: manifest unknown.
See 'docker run --help'.
```

Try again from desktop terminal:

same errors x- `xhost:  unable to open display ""` not this time.

Google: **“Unable to find image 'dustynv/jetbot_ros:foxy-r32.3.1' locally”**

https://github.com/dusty-nv/jetbot_ros/issues/47

> “Setting REVISION to 5.0 in nv_tegra_release temporarily resolved the issue in my case.”
>
> # R32 (release), REVISION: 3.1, GCID: 18186506, BOARD: t210ref, EABT: aarch64, DATE: Tue Dec 10
>
> *Change REVISION 3.1 to 5.0*
>
> Can’t do this as is READ ONLY file system
>
> Try here:
>
> https://hub.docker.com/r/dustynv/ros
>
> click on Tags
>
> https://hub.docker.com/r/dustynv/ros/tags
>
> look for foxy and 32.7.1
>
> Click on docker pull copy command
>
> jetbot@jetson-4-3:~$ **docker pull dustynv/ros:foxy-ros-base-l4t-r32.7.1**
>
> Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post http://%2Fvar%2Frun%2Fdocker.sock/v1.24/images/create?fromImage=dustynv%2Fros&tag=foxy-ros-base-l4t-r32.7.1: dial unix /var/run/docker.sock: connect: permission denied
>
> FAIL

https://forums.developer.nvidia.com/t/unable-to-initiate-jetson-inference-container-after-update/194828

jetbot@jetson-4-3:~/jetbot_ros$ docker/run.sh

```
reading L4T version from /etc/nv_tegra_release

L4T BSP Version:  L4T R32.3.1

[sudo] password for jetbot:

CONTAINER:     dustynv/jetbot_ros:foxy-r32.3.1

DEV_VOLUME:

DATA_VOLUME:   --volume /home/jetbot/jetbot_ros/data:/workspace/src/jetbot_ros/data

USER_VOLUME:

USER_COMMAND:

V4L2_DEVICES:    --device /dev/video0

xhost:  unable to open display ""

xauth:  file /tmp/.docker.xauth does not exist

Unable to find image 'dustynv/jetbot_ros:foxy-r32.3.1' locally

docker: Error response from daemon: manifest for dustynv/jetbot_ros:foxy-r32.3.1 not found:
manifest unknown: manifest unknown.

See 'docker run --help'.
```

run.sh

is looking for r32.3.1

But my version is:

```
L4T-README

       version

              nvidia-l4t-core.dpkg-s.txt

                     Version: 32.7.1-20220219090432
```

jetbot@jetson-4-3:~/jetbot_ros$ **cd /etc/**

jetbot@jetson-4-3:/etc$ **cp nv_tegra_release nv_tegra_release_ORIGINAL**

jetbot@jetson-4-3:/etc$ **sudo nano nv_tegra_release**

change

R32 (release), REVISION: **3**.1, GCID: 18186506, BOARD: t210ref, EABI: aarch64, DATE: Tue Dec 10 06:58:34 UTC 2019

to

R32 (release), REVISION: **7**.1, GCID: 18186506, BOARD: t210ref, EABI: aarch64, DATE: Tue Dec 10 06:58:34 UTC 2019

jetbot@jetson-4-3:~/jetbot_ros$ **docker/run.sh**

```
Unable to find image 'dustynv/jetbot_ros:foxy-r32.7.1' locally
```

didn’t help! Change back.

2022.05.02 HCC ITAI class

Demonstrated above error messages to R&D. Speculation since it seems to be an authorization issue that it has something to do with a recent NVIDIA hack and file disruption. Calls to service are pending. Will have to wait for restoration as there’s no “from source” alternative to the Jupyter notebook.