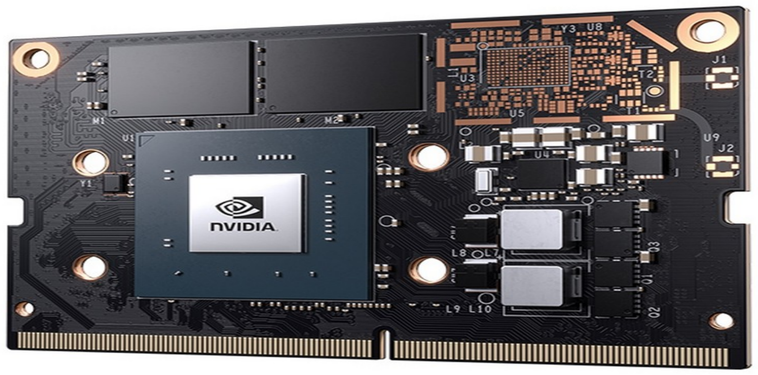
deck_donkey_notes.txt

Up front you will notice dates are not in order.
I put these notes because of importance to me.
Will try to group pictures with text.

4 Nov 2021



Got greedy to install cuda on Nano emmc module. It corrupted the emmc memory and
can't seem to flash a new OS Linux for Tegra. Can't get vendor ID and part ID
when I lsusb
Will search for fix

5 Nov 2021

Fix was to use

nvidia_sdk/nvidia_sdk/JetPack_4.5.1_Linux_JETSON_NANO folder

1) jumper forced recovery to ground
2) use short 3 foot USB cable (long extension won't allow lsusb to work)
    from host to Jetson Nano micro USB
3) power up with 5 volt barrel set up

4) remove recovery to ground jumper

5) lsusb ... look for NVidia name
    if missing ck 1-5 again if present go to step 6)
6) go to
cd nvidia_sdk/nvidia_sdk/JetPack_4.5.1_Linux_JETSON_NANO

7) sudo ./flash.sh jetson-nano-emmc mmcblk0p1

8) will take a while to complete so be patient until you
    see prompt

added Tue 19 Oct 2021 12:50:42 PM PDT

============ notes for donkey and deck donkey ===============================

Sat 28 Aug 2021 03:17:00 PM PDT

jetson-nano-emmc-module-notes.txt

Reasons to use Jetson Nano production module:

1) Uses emmc memory which is shock tolerant and faster then SD card

2) The production module runs at lower temps -25C versus 0C

Have two modules now:

1) andy@donkey-emmc

The id on this first one is 0995:7f21 and the OS (Linux for Tegra) was
installed using SDK Manager for jetson dev kits.

5 Nov 2021 ignore writing with //
go to procedure above with same date 5 Nov 2021
name changed to andy@donkey-emmc2

//2)andy@andy-emmc2

//The id on this one is 0995:7118 purchased from Arrow. This one WOULD NOT
//install from the SDK Manager. After a month search finally got lucky and
//installed using:

//Need to download: Jetson-210_Linux_R32.5.1_aarch64.tbz2

//Then extract to folder: /home/.../Jetson-210_Linux_R32.5.1_aarch64

//cd /home/...../Jetson-210_Linux_R32.5.1_aarch64/Linux_for_Tegra

//sudo ./flash.sh jetson-nano-emmc mmcblk0p1

==
For both modules

```
sudo apt install qtcreator qt5-default
```

```
sudo apt install synaptic
```

sudo apt install libqt5serialport5-dev or install using synaptic

Since you use serialports the user needs to be in group dialout:

```
sudo usermod -a -G dialout <user>
```

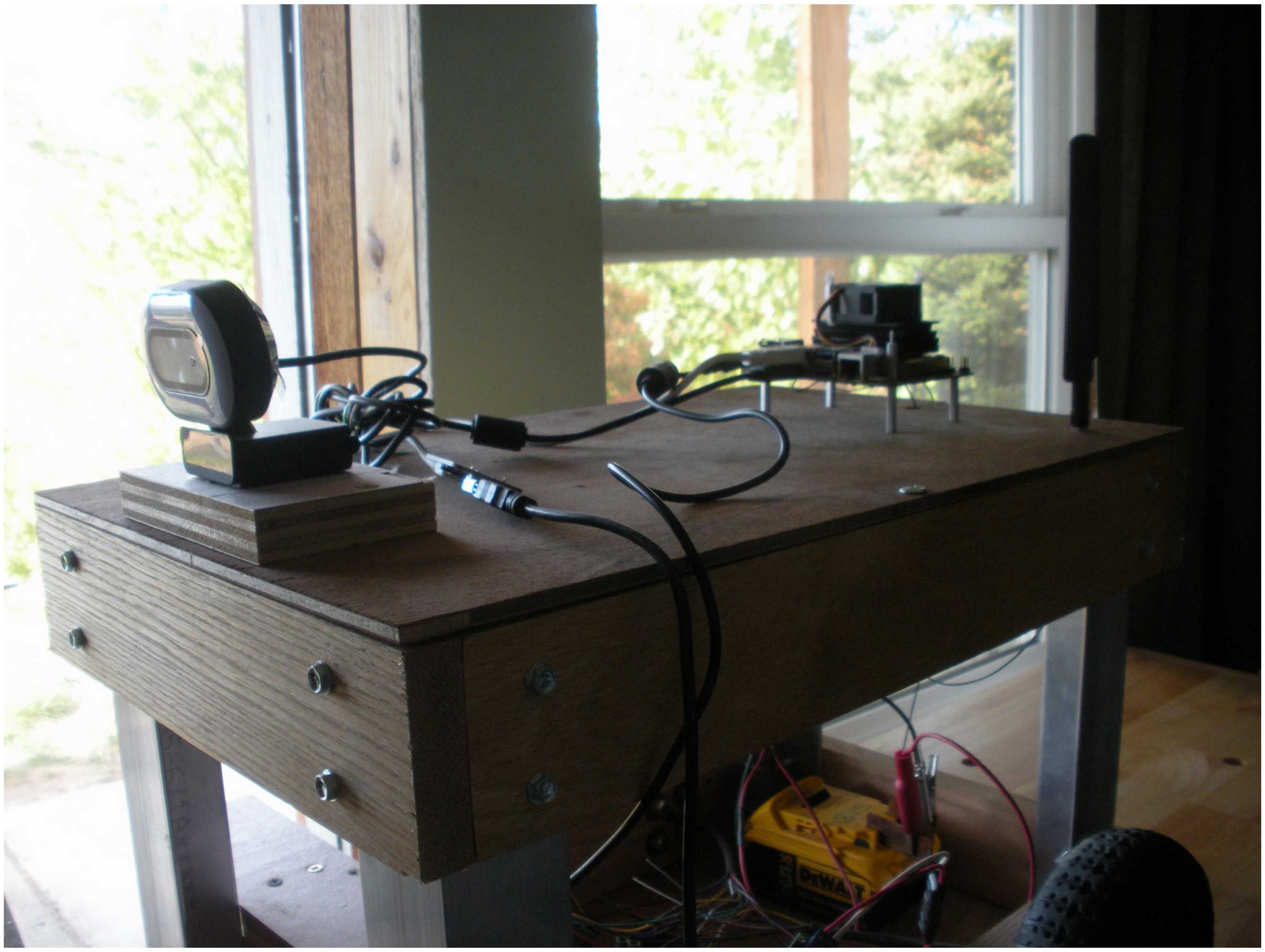
If using wifi the module needs to be connected to WiFi and mine required a password when connecting.

Most of the these are accessible via direct display although some can be done over ssh -X accept the wifi password

for use with USB web cam -----vvvvvvvvvvvvvvvvvvvvvvvvvvvvv ----------- below

```
 gst-launch-1.0   v4l2src device=/dev/video0 ! image/jpeg, width=320, height=240, framrate=10/1 !
jpegdec ! videoconvert ! videoscale ! ximagesink sync = false
```

A_Deck_Donkey_Theory



Sun 06 Jun 2021 09:02:07 AM PDT

This is the first writings on my Deck Donkey which will be referred to as DD
I have done some work already...
This project is so I can get use to BLDC motors and their controllers.

The motors I am using are the lower speed hoverboard motors ...
2-RoboWheel RW170 purchased from SkysEdge, who did not have a controller at
the time so I purchased an ODrive controller from ODriverobotics

I am in the process of writing some control software that will reside on my
Jetson Nano called ddod(x) the x will be a version number...this software is
for the odrive controller and is using the ASCII protocol...using the Nano
will allow me to run DD headless from my desktop

there is some development using the Native protocol but it is just starting
...I will keep abreast of it's development as it looks like it will be a more
roboust environment... the ODrive community looks to be a very knowledgable
group

Mon 14 Jun 2021 09:00:36 AM PDT

Was debating weather to use 48v lithium iron phosphate batteries for my Donkey projects; but have found out that the absolute peak voltage voltage that the odrive 56v model can handle is 56 volts. I have decided to use the 12.4V lithium iron phosphate batteries instead and build up to voltage 12, 24 or 36. Debating between 24 or 36 volt LiFePO4 batteries or could build one up from hoverboard.

Sat 03 Jul 2021 09:28:24 AM PDT

will start some notes on the deck donkey. Thought had some notes before if I find some , will add to these.

Got a production module with emmc and a Jetson Nano B01 board
This module (0995 7118) was not the same as previous (0995 7f21) so it would not load the Nano kernel from the SDK manager which is bitch. I will endeavor to find a work around; in the mean time I will revert back to the SD card. The SD card is slower, will not work below 0-F and is prone to shock. SOLVED: see note above dated 19 Oct 2021

I had issues with getting my code to work with serial interface. Finally notice some serial code would work with sudo! come to find out again I had to add user to dialout group (dialout is for serial comm):

```
sudo usermod -a -G dialout andy
```

andy is my user. After reboot serialport programs would work without sudo.

Next have to get gstreamer to work for my cameras

```
 gst-launch-1.0   v4l2src device=/dev/video0 ! image/jpeg, width=320, height=240, framrate=10/1 !
jpegdec ! videoconvert ! videoscale ! ximagesink sync = false
```

the above works with webcams over wifi, some latency but usable. I do need to work with cameras.

Tue 19 Oct 2021 01:22:30 PM PDT

Have finally got Odrive too work reliably with optical encoders on the Robowheels using this configuration setup:

```
odrv0.config.enable_brake_resistor = True
odrv0.axis0.motor.config.calibration_current = 2
odrv0.axis0.motor.config.pole_pairs = 15
odrv0.axis0.motor.config.resistance_calib_max_voltage = 6
odrv0.axis0.motor.config.requested_current_range = 35
odrv0.axis0.motor.config.current_control_bandwidth = 100.0
```

```
odrv0.axis0.motor.config.torque_constant = 8.27 / 10
odrv0.axis1.motor.config.calibration_current = 2
odrv0.axis1.motor.config.pole_pairs = 15
odrv0.axis1.motor.config.resistance_calib_max_voltage = 6
odrv0.axis1.motor.config.requested_current_range = 35
odrv0.axis1.motor.config.current_control_bandwidth = 100.0
odrv0.axis1.motor.config.torque_constant = 8.27 / 10

# encoder
odrv0.axis0.encoder.config.mode = ENCODER_MODE_INCREMENTAL
odrv0.axis0.encoder.config.cpr = 3200
odrv0.axis0.encoder.config.bandwidth = 1000
odrv0.axis1.encoder.config.mode = ENCODER_MODE_INCREMENTAL
odrv0.axis1.encoder.config.cpr = 3200
odrv0.axis1.encoder.config.bandwidth = 1000

odrv0.axis0.controller.config.input_mode = INPUT_MODE_VEL_RAMP
odrv0.axis0.controller.config.control_mode = CONTROL_MODE_VELOCITY_CONTROL
odrv0.axis0.controller.config.pos_gain = 1
odrv0.axis0.controller.config.vel_gain = 0.002 * odrv0.axis0.motor.config.torque_constant *
odrv0.axis0.encoder.config.cpr
odrv0.axis0.controller.config.vel_integrator_gain = 0.01 * odrv0.axis0.motor.config.torque_constant *
odrv0.axis0.encoder.config.cpr
odrv0.axis0.controller.config.vel_limit = 5
odrv0.axis1.controller.config.input_mode = INPUT_MODE_VEL_RAMP
odrv0.axis1.controller.config.control_mode = CONTROL_MODE_VELOCITY_CONTROL
odrv0.axis1.controller.config.pos_gain = 1
odrv0.axis1.controller.config.vel_gain = 0.002 * odrv0.axis1.motor.config.torque_constant *
odrv0.axis1.encoder.config.cpr
odrv0.axis1.controller.config.vel_integrator_gain = 0.01 * odrv0.axis1.motor.config.torque_constant *
odrv0.axis1.encoder.config.cpr
odrv0.axis1.controller.config.vel_limit = 5

# BREAK HERE
odrv0.save_configuration()
odrv0.reboot()


odrv0.axis0.requested_state = AXIS_STATE_MOTOR_CALIBRATION

dump_errors(odrv0)

odrv0.axis1.requested_state = AXIS_STATE_MOTOR_CALIBRATION

odrv0.axis0.motor.config.pre_calibrated = True
odrv0.axis1.motor.config.pre_calibrated = True

BREAK HERE
odrv0.save_configuration()
```

```
odrv0.reboot()
```

Encoder

```
odrv0.axis0.encoder.config.calib_range = 0.05
odrv0.axis0.requested_state = AXIS_STATE_ENCODER_OFFSET_CALIBRATION
odrv0.axis1.encoder.config.calib_range = 0.05
odrv0.axis1.requested_state = AXIS_STATE_ENCODER_OFFSET_CALIBRATION
```

Movement

```
odrv0.axis0.requested_state = AXIS_STATE_CLOSED_LOOP_CONTROL
odrv0.axis1.requested_state = AXIS_STATE_CLOSED_LOOP_CONTROL
odrv0.axis0.controller.input_vel = 0.5
odrv0.axis1.controller.input_vel = 0.5
```

Your motor should spin here

```
odrv0.axis0.controller.input_vel = 0
odrv0.axis1.controller.input_vel = 0
odrv0.axis0.requested_state = AXIS_STATE_IDLE
odrv0.axis1.requested_state = AXIS_STATE_IDLE
```

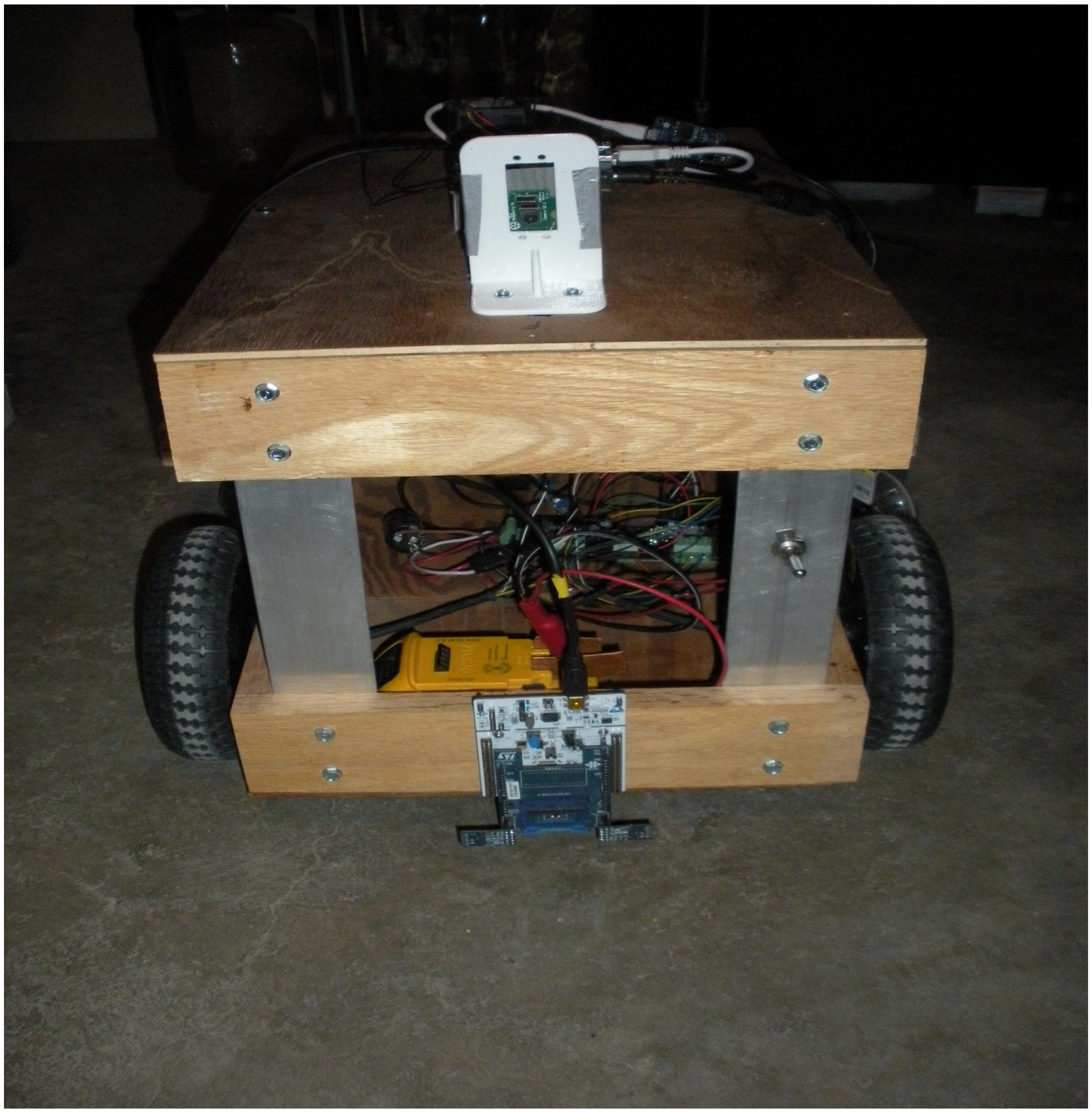
When I was using the hall sensors I had to be gentle or I would lose control to a motor or both; with the encoder I can bang into objects and do high speed pivots "ddod5_test1_encoder" video, go over gravel terrain "ddod5_test2_encoder" video and running on deck planks "ddod5_test3_encoder" video.

Wed 20 Oct 2021 06:06:46 PM PDT

Working with DD, am thinking about making the DD either front wheel or rear wheel drive. Could do this with a check box in the GUI and a camera remount. If Jetson Nano were mounted centrally could mount two cameras and just pay attention to which camera is forward. Speaking of cameras need to purchase some that work in the slots on the Nano that have replacable lenses.

27 Oct 2021

Working with RPI camera v2.1 finally found a script that would work. Latency was very long but using different parameters got it to be acceptable and usable.



```
gst-launch-1.0 nvarguscamerasrc ! 'video/x-raw(memory:NVMM),width=640, height=480,
framerate=21/1, format=NV12' ! nvvidconv flip-method=0 ! 'video/x-raw,width=960, height=616' !
nvvidconv ! ximagesink -e
```

With the above latency is a second or worse

```
gst-launch-1.0 nvarguscamerasrc ! 'video/x-raw(memory:NVMM),width=320, height=240,
framerate=12/1, format=NV12' ! nvvidconv flip-method=0 ! 'video/x-raw,width=320, height=240' !
nvvidconv ! ximagesink -e
```

With these parameters latency is usable for my purpose on jetson Nano

I have a definite issue with ssh communications

29 Oct 2021

Having a lot of hang ups with video stream and ddod program. I believe it is a communications problem with outside wifi extender.
I shut the POE to the outside unit shutting it off leaving me with just the inside Netgear router.

Video latency improved right away but need to run ddod test upstairs and down.
Video recoding 'ddod_encoder_ver10_5' shows the proof of having good communications is necessary for glitch free control
of DD using ssh

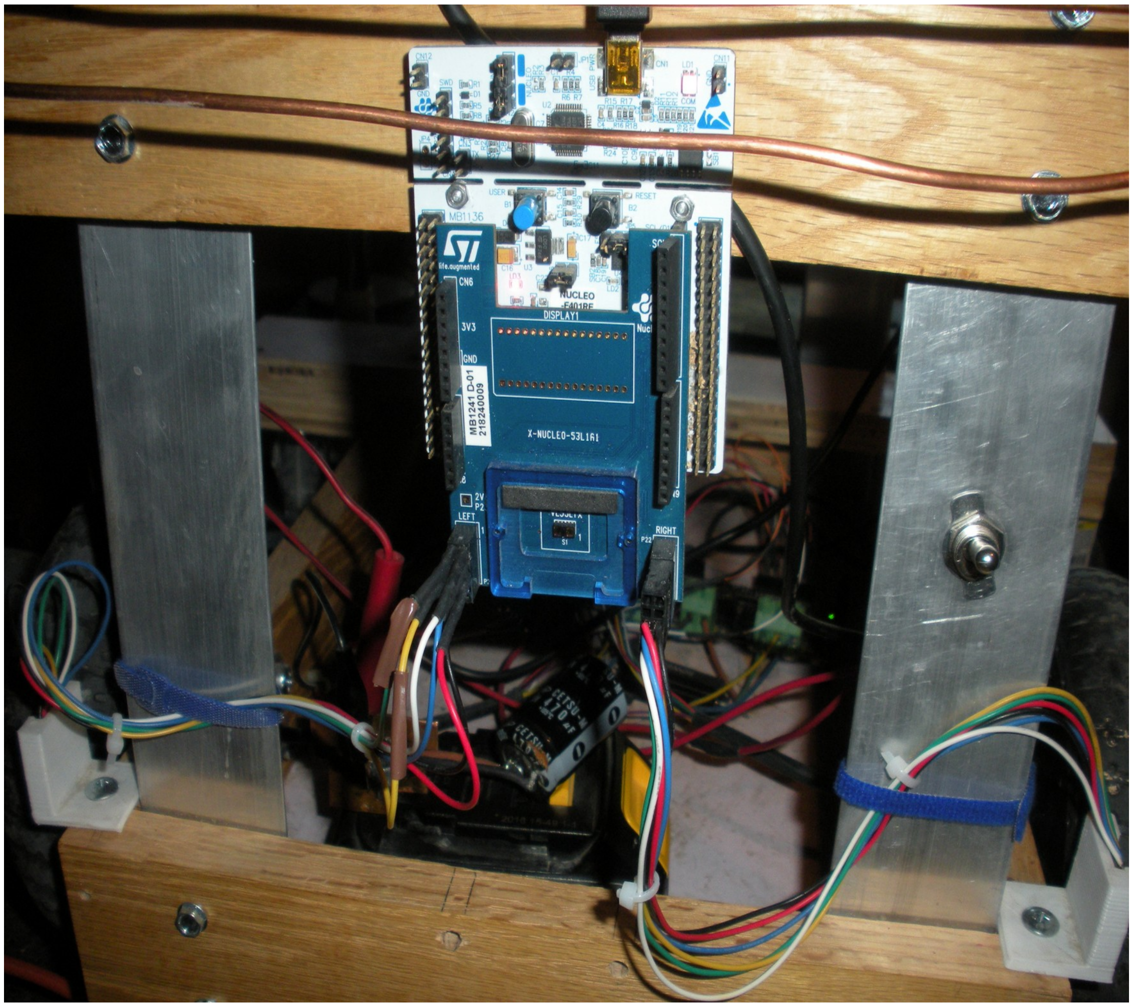
Need to invest in communications research so that maintain control of DD from anywhere. Might need to look at a communications mesh network?

31 Oct 2021

Try to reenter ssh -XC andy@andy-emmc2.local and could not; I knew the address 10.0.0.7 and was successful in pinging it...
after that I could access andy-emmc2.local via ssh .... go figure!

11 Nov 2021

installed tof sensor board to DD and wrote collision supporting software to stop DD...need to work on slam or at least
guidance system



6 Dec 2021

color code for the TOF sensors to the STM evaluation board is as follows.

1 white	2 green
3 blue	4 yellow
5 red	6 black

#1 is the square printed circuit pad

Sun 19 Jun 2022 07:53:21 AM PDT

Some comments I have posted to Odrive forum
==
posted 22apr 2022
I have a two wheel drive robot that uses 170mm wheels with encoders (Deck Donkey Project). I have to calibrate the encoders each time I power up. The robot which weighs about 30 lbs does so on a level floor without any problem…
exception is if one of my encoder wires comes loose …

When I calibrate I do both motors at once and the robot pivots 90 degrees and then back to original position.

With 4 wheels going at once there may be friction. Some creative coding might be possible.

jun 17 2022 I posted this:

Since the above post I have changed the wiring set up on one of my motors.
I had both axis's M0 and M1 wired (A)Blu-(B)Yel-(C)Grn. This caused my DD to pivot during encoder calibration.
Now I have changed just M0 to (A)Grn-(B)Yel-(C)Blu. This causes my DD to just back-up and return to original position during encoder calibration

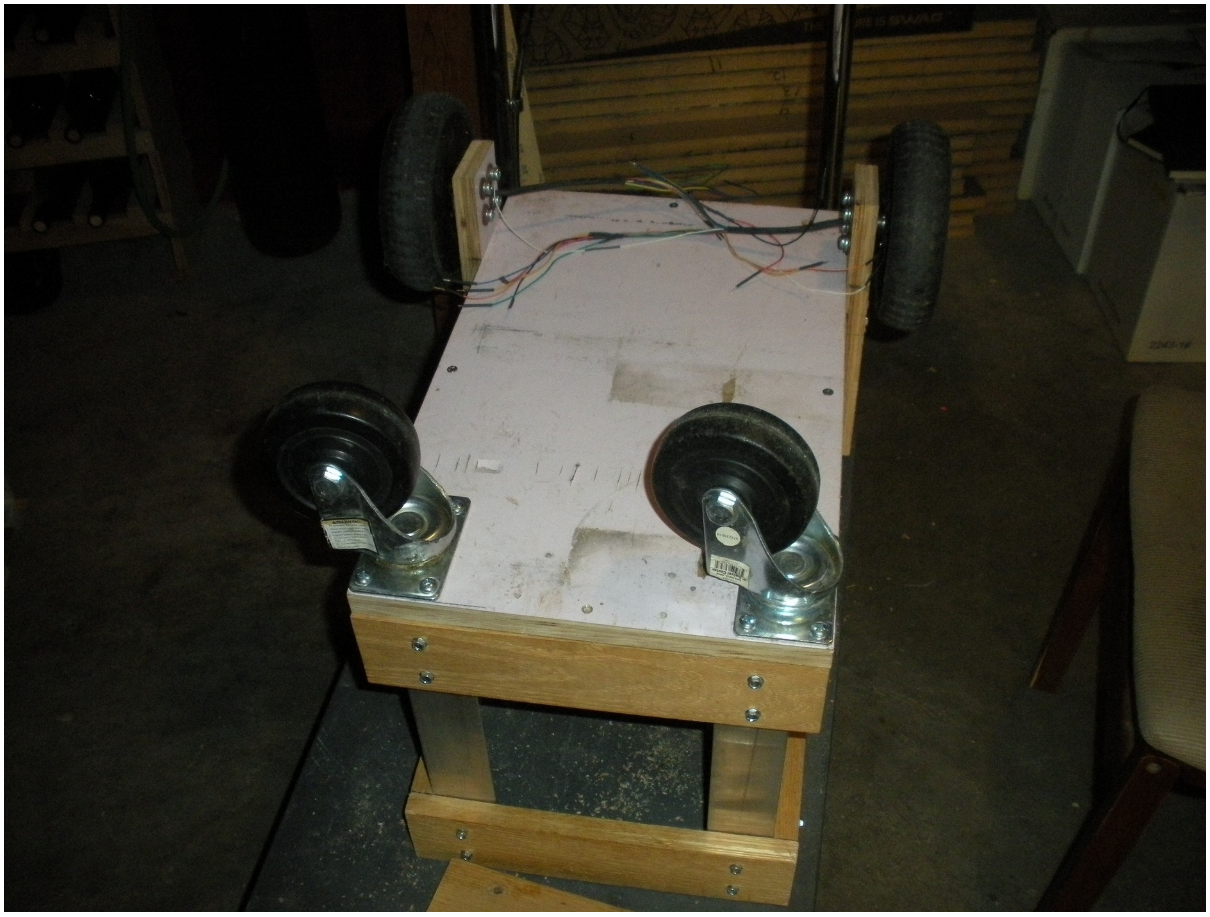
I see this as a plus for 4 wheel drive calibration.

jun 19 2022 I installed without power two more hoverboard motors as out riggers.
original wheels calibrated OK....but
could not pivot or turn as I did with swivel dolly wheels

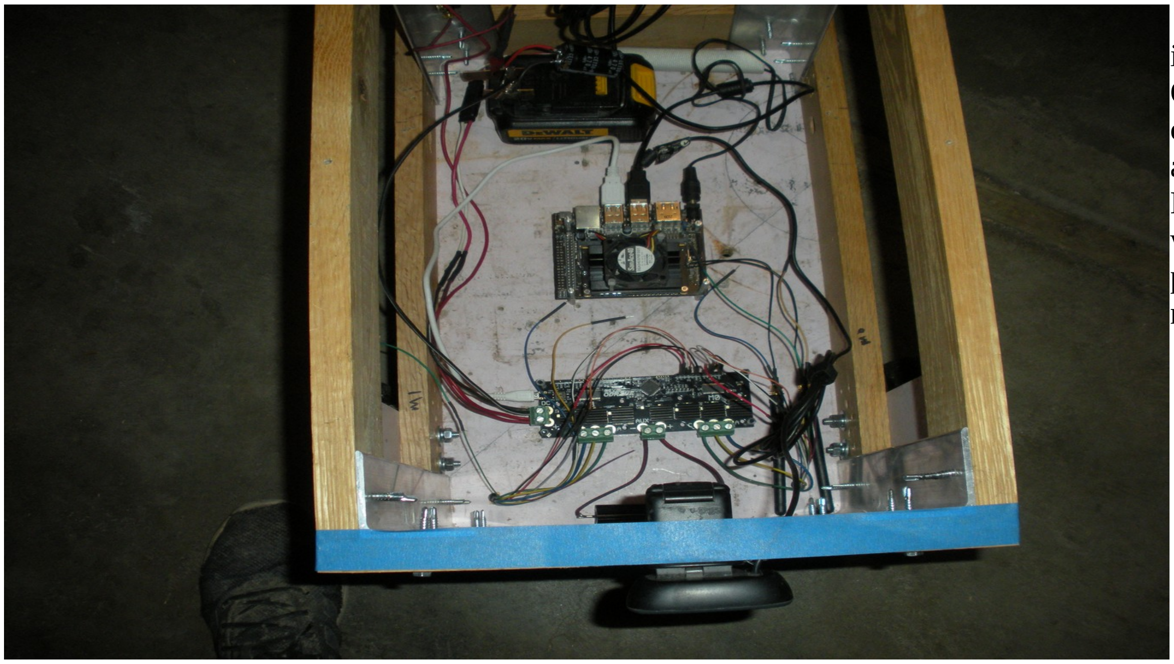
==

Sun 18 Sep 2022 09:11:59 AM PDT

At the end of Aug- 2022 - until date above have been rebuilding DD

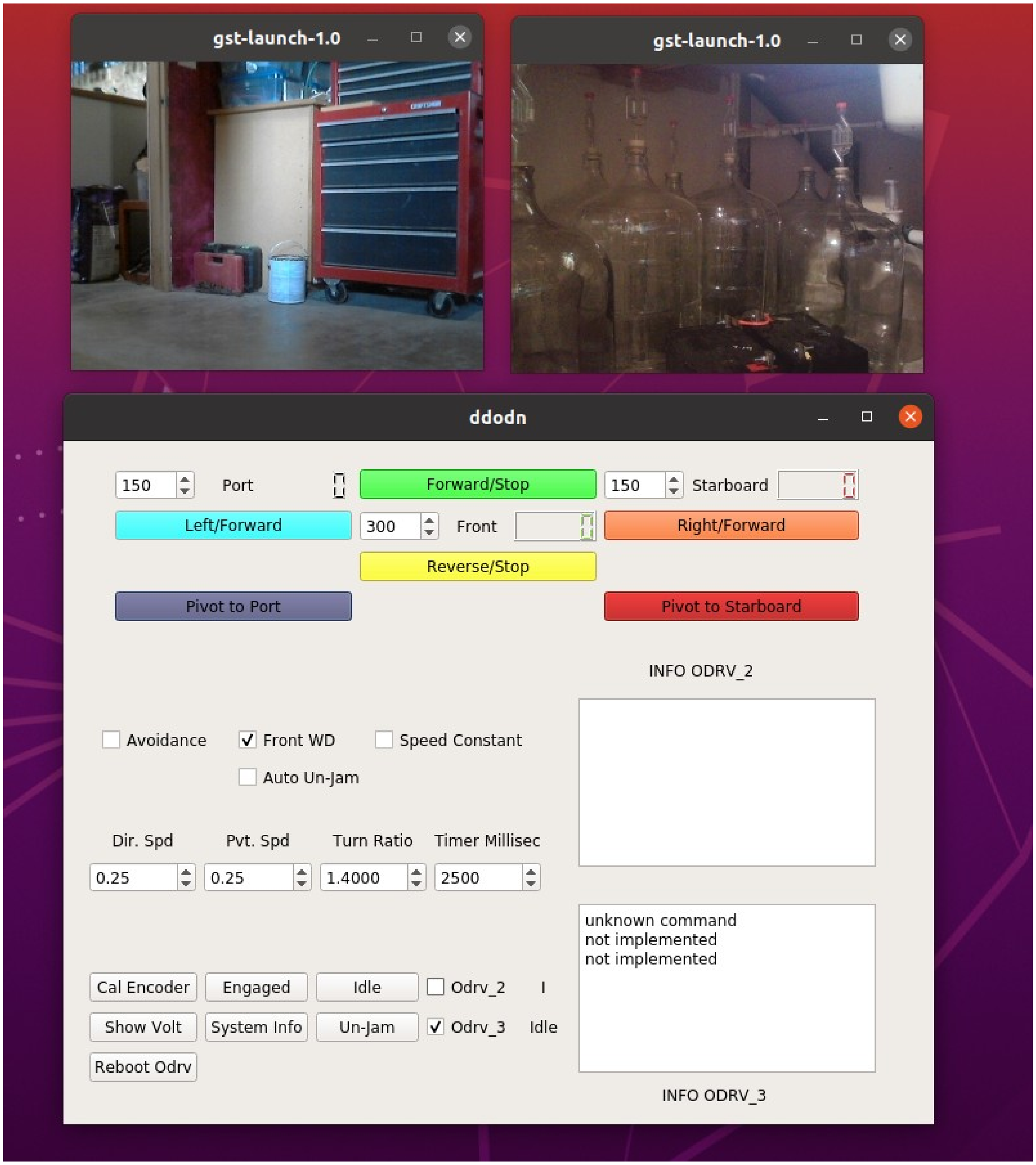


have mounted wheels to give around 5" ground clearance



install Odrive controller and Jetson Nano B01 with production module

...at present running program ddodn and two



Port: 150; Starboard: 150; Front: 300

[ ] Avoidance [✓ Front WD] [ ] Speed Constant
[ ] Auto Un-Jam

Dir. Spd	Pvt. Spd	Turn Ratio	Timer Millisec
0.25	0.25	1.4000	2500

[ ] Odrv_2 I
[✓ Odrv_3] Idle

```
unknown command
not implemented
not implemented
```

USB cameras using gstreamer:

```
gst-launch-1.0 -v v4l2src device=/dev/video0 ! video/x-raw, width=320, height=240, framerate=30/1,
format=YUY2 ! videoconvert ! xvimagesink
```

and

```
gst-launch-1.0 -v v4l2src device=/dev/video1 ! video/x-raw, width=320, height=240, framerate=30/1,
format=YUY2 ! videoconvert ! xvimagesink
```

I have run some outside tests and run the DD up my 38 percent grade ramp.

In the front wheel mode the DD would skew hard to right or left. In the rear wheel drive mode the DD would maintain its direction.
I'm guessing it's a weight distribution issue or tire tread issue.
When stopped the DD would stay in position as long as the current was engaged.