**Lawn Tractor Startup Procedures - 2022:**

At the Radio Control Unit

* Ensure battery power is sufficient
* Turn the unit on

At the tractor

* Disengage transmission - pull lever at back
* Pull back cover
* Ensure wiring connections are complete
  + Steer potentiometer
  + Steer motor
  + Transmission
* Unplug the laptop and ensure you can log in remotely
* Disconnect the battery charger (i.e. shore power)
* Turn on master power
* Replace cover
* Start engine

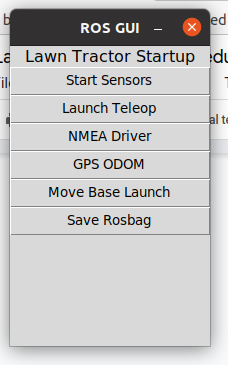
**Old Notes**

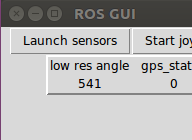
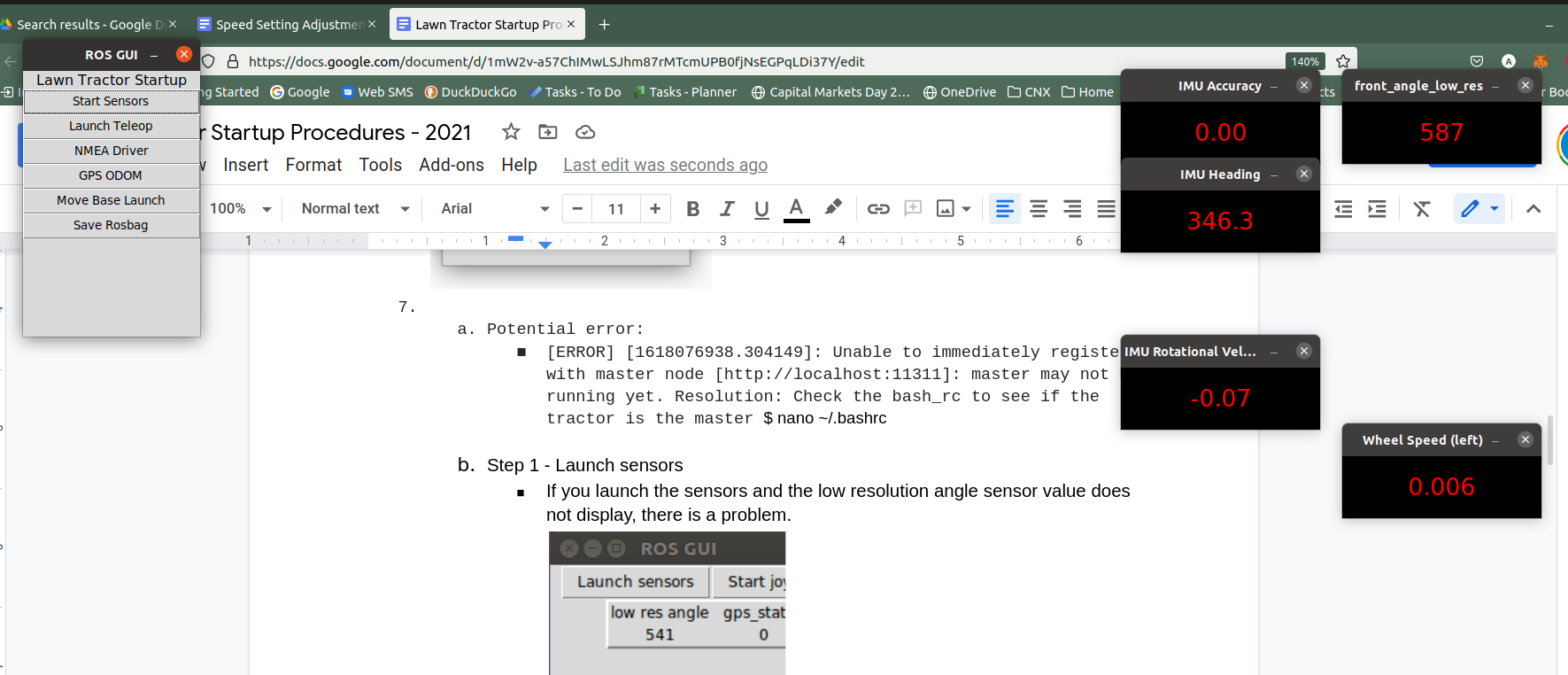
1. Check <https://192.168.10.1/cgi-bin/luci> to make sure the Tractor is getting an IP, root/Jackson69



From RPi, $ ping controller\_aj; From laptop, $ ping tractor\_aj - If these don’t work $ sudo nano /etc/hosts and make corrections

1. If you intend to run a pre-planned mission, follow steps at “**Creating a waypoint plan using path\_generator.py**” to create the mission. A more simple way to create a test mission is to use instructions at: CMD\_VEL Path Planning.
2. Place the base station in the yard; Give it power; Make sure the wi-fi is being provided; The 3DR radio needs 12 volt - there should be a solid green light if it is paired with the tractor.
3. Turn on the laptop, make sure the base station is presented and connected to the internet and you are able to get to the production code.
4. At the tractor:
   1. Confirm the physical transmission control arm is dis-engaged (i.e. in neutral)
   2. Clear away all loose material
   3. Confirm drive belt is clear
   4. Confirm all wires are connected and none are loose
   5. Pull the circuit breaker on transmission
   6. Disconnect power from RPi
   7. Remind yourself where the manual emergency shut off is - make sure it is clear
   8. Put the RF shut off in your top left shirt pocket
   9. Put in ear plugs
   10. Move tractor to starting position
   11. Engage main power from the battery
   12. Plug in the plastic key to the ignition
   13. Straighten front wheels
   14. Hold brake switch down
   15. Start the mower
   16. Re-insert the circuit breaker on transmission - note the transmission may be in reverse until the “launch sensors” is started
   17. Connect power for the RPi
   18. Wait 30-45 seconds and try to PuTTy into RPi
   19. When testing
       * Make sure transmission is in neutral; Roll out tractor; Connect Circuit breaker; Connect battery charger; Connect 12volt power to fire up RPi
       * Try to PuTTy into RPi
5. Ctrl\_Alt+T; $ python3 /home/al/python/ros\_gui/ROS\_GUI\_v7.py
6. ~~On Laptop, Ctrl\_Alt+T; $ python /home/al/python/ros\_gui/ROS\_GUI\_v6.py~~



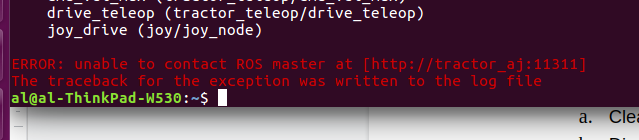
* 1. Potential error:
     + [ERROR] [1618076938.304149]: Unable to immediately register with master node [http://localhost:11311]: master may not be running yet. Resolution: Check the bash\_rc to see if the tractor is the master $ nano ~/.bashrc
  2. Step 1 - Launch sensors
     + If you launch the sensors and the low resolution angle sensor value does not display, there is a problem.
     + 
     + 
     + if (440 <= current\_position\_low\_res && current\_position\_low\_res <= 640) then the process will continue. If not, you will have to straighten the front wheels.
     + Check to make sure transmission is in neutral (TODO - add transmission reading to GUI)

The PID controller for speed control needs to be re-written. Until that is done the speed control is hard coded to use parameters set in /speed. [3] is the PWM for 0.5-1.0 linear.x; [4] is the PWM for 0.0-0.5 linear.x; - be careful - just pressing the deadman may trigger [4]

1550 = max; 1525 = ~ 1 m/s; 1490 = ~ 0.5 m/s

$ rosparam set /speed "[1000, 1290, 1550, 1525, 1490, 70, 5, 0, 20]"

* 1. Step 2 - Launch Teleop/Start joystick
     + Click Joystick icon
     + Turn power on 2.4GHz receiver gamepad on and off and confirm the red led comes on
     + Press deadman switch and move joystick steering and transmission control to confirm the tractor is under control. The cmd\_speed and angle values should change, the steering wheels should move and the low res angle value should move.
     + 
     + If you get this error:



the RPi on the tractor is not functioning correctly. See Debug RPi Network Issues.

IMU - serial\_to\_ros.py needs to be started - it publishes

pub\_imuacc = rospy.Publisher('bno085\_imuacc', Float32, queue\_size=1)

pub\_rotvel = rospy.Publisher('bno085\_rotvel', Float32, queue\_size=1)

pub\_hdg = rospy.Publisher('bno085\_heading', Float32, queue\_size=1)

pubList = rospy.Publisher('bno085\_array', Float32List, queue\_size=1)

* 1. Check joystick control with steering - if the joystick is slow to respond it may be due to the laptop being on one network and the tractor being on another network. You do not want network latency for sensitive steering control functions.

1. Step 3 - Start NMEA Driver; Confirm GPS status “2” appears on gui
2. Step 4 - Start GPS odom

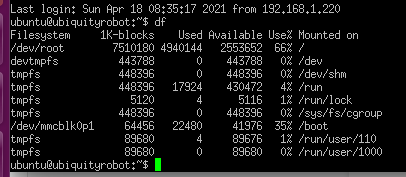
(At this point start the tractor and move to the starting position)

1. Step 5 - MoveBase
   * Adjust the starting position of the tractor in origin statement of the farm.yaml file: origin: [-23.0, -8.0, 0.0] so that it is physically on the correct spot on the map. Stop and restart the simulator as needed.
   * [ x, y, z] = x - left to right, smaller is closer to the left; y - top to bottom, smaller is closer to the bottom
2. Confirm steering is under joystick control
3. RVIS on the Laptop $ rosrun rviz rviz
4. Step 6 - Bagfile

**Engage transmission**

1. Drive with joystick or Tab 6 - RPi: # run a mission
   1. $ plink RPI\_on\_ZeroTier -pw ubuntu
   2. {Before next statement - update cmd\_vel\_input.txt with mission}
   3. {Don’t run a mission yet, - Step 9 below runs the mission}
   4. $ cd ~/catkin\_ws
   5. $ rosrun beginner\_tutorials meters\_travelled\_v1.py cmd\_vel\_input.txt

SHUT DOWN PROCESS:

* REPLACE BATTERY CHARGER WHEN DONE
* Disconnect Fuse to to TRANSMISSION SERVO
* Move bagfiles:
  + Open FileZilla
  + From RPI : /home/ubuntu/.ros/bagfiles
  + To Laptop : /home/al/bagfiles/
  + Delete old bagfiles from RPi
* Delete old log files:
  + From RPI: /home/ubuntu/.ros/log
  + Keep only latest
  + Copy anything you think you should retain
  + $ rosclean check or $ rosclean purge
* $ df # check available disk space - 6/21/20 used 66%
* 
* $ rosclean # another option
* Optional:
  + Turn off USB ports with Teensy’s connected
  + Shut down ROS processes

Things to test/watch as the system matures:

* 1. Tab 4 - Ubuntu, $ rosrun plotjuggler PlotJuggler
     + Check these data elements to plot
       - Course\_over\_ground
       - Front\_angle\_raw #actual
       - Front\_angle\_target #target
       - Heading
       - Left\_speed #actual
       - Left\_speed\_travelled\_msg #delete?
       - Satellite\_no
       - Satellite\_qual
       - Speed
       - Steer\_effort\_b4\_ra (for debugging)
       - Steer\_effort\_raw (for debugging)
       - Transmission\_PWM
     + File -> Recent Layout -> tractor1.xml # Load previously saved preferences. I have had issues where KLM values were not being published. The resolution was to unplug and reconnect the front\_angle
  2. Tab 7 - Ubuntu: # {update /steer or /speed as needed}
     + $ rosparam get /speed
     + $ rosparam set /speed "[1000.0, 1290.0, 1600.0, 135.0, -100.0, 70.0, 5.0, 0.0, 20.0]"
     + $ rosparam get /steering
     + $ rosparam set /steering "[-14000.0, 13500.0, 0.0, 60.0, 3.0]"

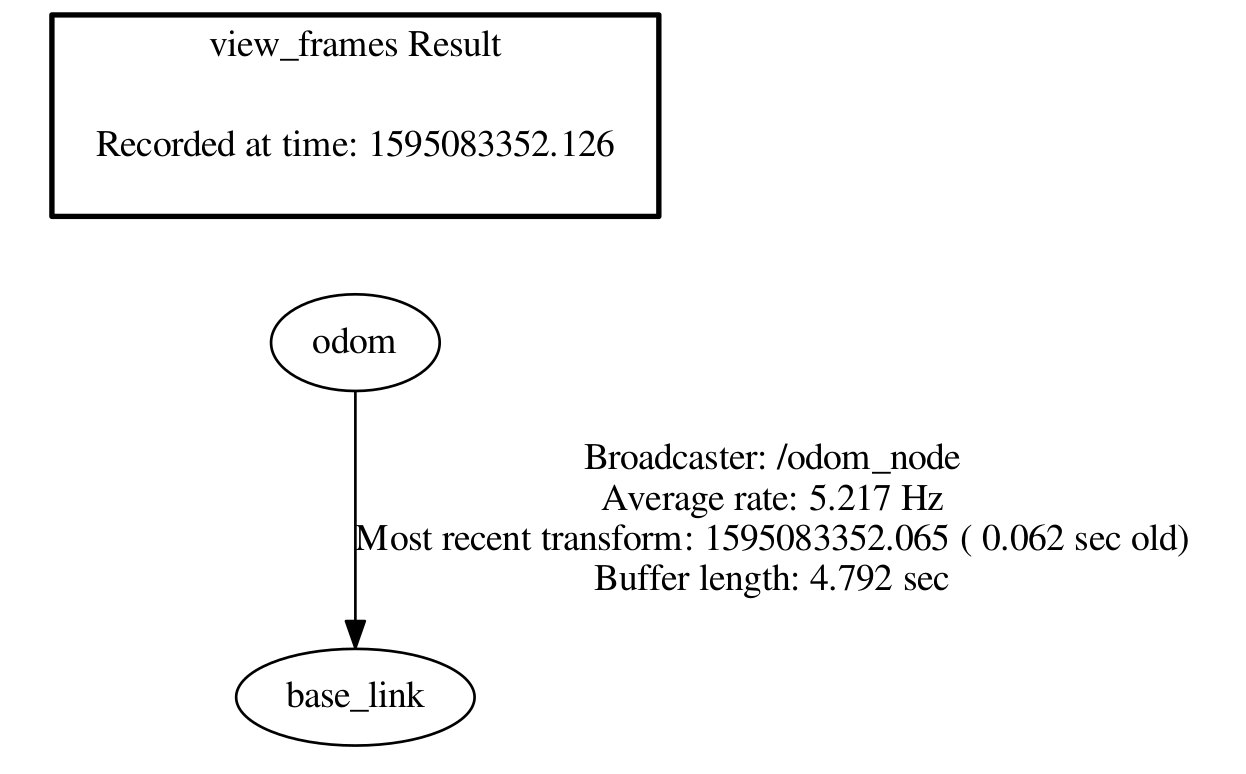
1. Tab 1, Checklist of processes:
   1. Teensy - check\_rpm publishes /left\_speed\_avg and prints “left\_speed sensor…..”
   2. Teensy - speed\_control publishes /transmission\_PWM and prints “speed\_control -...”
   3. Teensy - front\_anglel publishes /front\_angle\_klm and prints “front\_angle -...”
   4. Teensy - steer\_control publishes /front\_angle\_target and prints “steer cmd\_vel -...”; plus others
2. Tab 7, Check Rostopic are publishing
   1. $ rostopic echo /speed
   2. $ rostopic echo /nmea\_sentence
   3. $ rostopic echo /front\_angle\_raw

Monitor rvis and odom

$ rostopic echo /odom

Take a copy of the

* Tab 5 - $ rosrun tf view\_frames && evince frames.pdf (old)
* Try $ rosrun tf2\_tools view\_frames.py



* $ rosrun rqt\_graph rqt\_graph # save the file with a time stamp for debugging

