ECE 350/450 Intro to Robotics, Lab 1

**Get Familiar with F1/10 Race Car**

Nicholas LaRosa, Roland Nowak, and Nolan Sornson

Department of Electrical & Computer Engineering

Lehigh University

{nil318, rln220, nls221}@lehigh.edu

Date: 2/23/2021

**Abstract**

In this lab, our team was introduced to and configured our race car and associated equipment that will be used throughout the rest of the semester. All team members individually installed host machine software packages including Ubuntu 18.04 and Robot Operating System (ROS) and familiarized themselves with the basics of Linux, ROS, and Python. All team members collaborated on race car configuration including assembly, VESC motor tuning, and Jetson TX2 software installation. Most tasks were completed as designed, however the team encountered issues with flashing the Jetson TX2 that remain unresolved.

**Introduction**

Over the past three weeks, our team has quickly gotten up to speed on the foundation of the F1/10 autonomous race car platform. The main features and tools that make up this foundation are as follows:

* F1/10 - an open source learning platform developed by University of Pennsylvania that is designed to teach students about Perception, Planning, Control and design of autonomous systems.
* F1/10 car - a 1/10th scale Ford Fiesta ST Rally Car RC car with a custom control platform consisting of a NVidia Jetson TX2, Orbitty Carrier Board, and VESC. It has also been upgraded with a 3S Lithium Ion Battery and a brushless motor. The Lithium Ion battery should be charged whenever a cell voltage drops below 3.4V. Brushed motors are inexpensive and can make high torque, but require maintenance as the brushes require physical contact with the rotating components within the motor and make sparks during operation. Brushless motors have a longer life and have electronic commutators that allow for precise control over the speed. These motors also create less heat than their brushed counterparts and utilize less energy from the battery.
* Pitch, Roll, Yaw - the three dimensions that define motion in the axes of rotation. Pitch is nose up or down, roll is tilting side to side, and yaw is changing the direction of motion left and right.
* Differential Drive System - a differential drive allows the wheels on opposite sides of a car to rotate at different speeds. This enables easier turning without understeer.
* Ackerman Steering System - this steering system uses linkages to move the wheels uneven amounts so that the inside tire is turned sharper into the turn than the outside wheel. This prevents wheel slip during the turn.
* Ubuntu 18.04 Bionic Beaver - a Long Term Support (LTS) release of the Ubuntu Linux distro. This open source operating system is required for installing ROS Melodic Morenia
* Robot Operating System (ROS) Melodic Morenia - This release of ROS is compatible with Ubuntu 18.04 and includes software tools that allow for the development of software packages for the F1/10 car using catkin\_make.
* VESC - a full featured electronic speed control module developed by Benjamin Vedder. Allows for control of multiple motors and servos and has a software package that allows for custom tuning.
* Python - a high level, dynamically typed, multiparadigm programming language. Python is easy to write and read, and will allow the team to quickly design programs for the F1/10 car. ROS Melodic Morenia uses Python 2.7.
* NVidia Jetson TX2 - an embedded computer that has 256 CUDA codes to support Artificial Intelligence programs while remaining power efficient.
* NVidia Jetpack - a software package designed to support autonomous vehicles that are powered by the NVidia Jetson embedded computer platform.
* Orbitty Carrier Board - a custom motherboard that breaks out the essential I/O for a NVidia Jetson TX2.
* PID control - PID stands for Proportional, Integral, Derivative. When we try to set the speed of the motor, a control signal is sent to the VESC that defines our target speed. The VESC then outputs a pulse width modulated signal that drives the motor and reports back how far away from the target the current motor speed is. The control input is then adjusted in relationship to the current error using the P, I and D values. Using the PID transfer function we are able to predict the growth of the error and adjust smoothly to a new target speed.

**Procedures**

1. The team began the semester virtually and therefore opted to install Ubuntu 18.04, ROS, and NVidia Jetpack, before starting on F1/10 car software. Most team members attempted to use a Virtualbox Virtual Machine to run Ubuntu within Windows, however when installing NVidia Jetpack, it became clear that the default hard drive size of 20 GB in Virtualbox would not allow for the installation of NVidia Jetpack 4.4.1 which requires downloading 25 GB of files. Some team members adjusted their hard drive size, while the team member who will be interfacing regularly with the F1/10 car and NVidia Jetson, opted to dual boot Ubuntu 18.04 alongside Windows 10. A tutorial was followed to ensure proper installation [1]
   1. Dual booting is a fun process, but can appear daunting to many people who are unfamiliar with Linux. While it is possible to fully erase an existing Windows installation if one is not careful during the installation process, the graphical installer makes it pretty easy to select “install alongside” to prevent that issue. I would encourage more people to dual boot to take advantage of the full power of their computer if they are able to make a backup of their hard drive and are confident in their ability to follow the instructions carefully.
2. The team reviewed the car assembly and found it to be assembled properly and well. However, one note to make car use easier in the future, the wires for the USB hub and the LIDAR unit should be zip tied on the underside of the custom top plate to make accessing the screws for the top plate easier to allow for faster removal of the battery.
3. The team continued on to PID tuning using VESC-tool. The firmware on the VESC was already updated, so that step was skipped. The team moved on to PID tuning on the VESC. We achieved PID control with the VESC by first starting with the UPenn configuration XML file. This file was supposed to set reasonable starting values for the VESC, however we found that all of the values were far too large and were causing instability. We reduced Ki and Kd to very close to 0 and adjusted Kp until we saw corrections of < 500 rpm for our targeted 500 rpm steps. Then we increased Ki until we achieved exactly 500 rpm steps. Finally to combat overshoot and provide better damping, we increased Kd until just before we saw the initial overshoot start to grow again.
4. The first attempt was made to flash the Jetson TX2 using the CTI-L4T tool provided by Connect Tech, the manufacturer of the Orbitty Carrier board. The steps in Module A, Part 3 were followed exactly, installing Jetpack 4.4.1 and extracting the CTI-L4T tool was completed without issue, however, running the ./flash.sh never seemed to complete without error. The errors we always got were: “BootRom not running” and “CPU Bootloader is not running”. Googling these errors did not yield any immediate fixes.
   1. A forum post [2] pointed to a bad install of Jetpack as a potential source of error, so we uninstalled and reinstalled Jetpack to attempt again with no success
5. A final attempt was made to flash the Jetson TX2. NVidia SDK manager and all of the previous steps were repeated, but the flash failed again. The team's conclusion is that there is something wrong with the host computer’s Jetpack installation and will attempt another flash with a separate computer as this is only a step that needs to be completed once.
   1. Because the source of error is unknown, no advice can be provided yet about how to prevent this in the future.
6. The last remaining task that is incomplete is setting up the wireless networking of the Jetson TX2. This will be completed after a successful flash.
7. Next, we performed the Linux tutorials listed. Since all team members already have a working knowledge of Linux and its different commands and utilities, these tutorials were mainly a review. After completing the tutorials, we were able to answer the following questions:
   1. A new terminal can be opened by pressing Ctrl+Alt+T on the keyboard, or by clicking the app drawer in the lower left corner and selecting “Terminal”.
   2. Different text editing tools are better suited for different applications, but the primary text editor that I use is nano for general tasks. Additionally, for more in-depth edits or where syntactical color coding is helpful I will sometimes use gedit, and I occasionally use vim when debugging source code due to its tools in seeking to problematic code entries.
   3. Linux contains a permission system to protect certain privileged parts of the operating system that an adversary could potentially use as an attack vector. The “sudo” command allows the command following it to be run at the highest level of privilege. For example, “sudo ls” can list the contents and access properties of any directory on the system, including directories that are owned by other users.
   4. To check the access properties of a file, the command “ls -l” is used. For example, when “ls -l” is run from the home directory, the following is returned:

total 60

drwxr-xr-x 5 nick nick 4096 Feb 16 16:01 catkin\_ws

drwxr-xr-x 2 nick nick 4096 Feb 11 04:31 Desktop

drwxr-xr-x 2 nick nick 4096 Feb 10 19:45 Documents

drwxr-xr-x 3 nick nick 4096 Feb 10 21:34 Downloads

-rw-r--r-- 1 nick nick 8980 Feb 10 19:30 examples.desktop

drwxr-xr-x 2 nick nick 4096 Feb 10 19:45 Music

drwxr-xr-x 3 nick nick 4096 Feb 10 21:34 nvidia

drwxr-xr-x 2 nick nick 4096 Feb 10 19:45 Pictures

drwxr-xr-x 2 nick nick 4096 Feb 10 19:45 Public

drwxr-xr-x 3 nick nick 4096 Feb 23 15:40 snap

drwxr-xr-x 5 nick nick 4096 Feb 23 15:34 team8\_ws

drwxr-xr-x 2 nick nick 4096 Feb 10 19:45 Templates

drwxr-xr-x 2 nick nick 4096 Feb 10 19:45 Videos

In the second line, there is a “d” in the first character indicating that “catkin\_ws” is a directory. The following nine characters indicate the access properties. The first three characters “rwx” indicate that the owner of the file (“nick” in this case as indicated by the owner username after the access permissions) has read, write, and execute permissions on the directory. The second group of three letters “r-x” indicates that the group of the file (“nick” in this case as indicated by the second group name listed after the owner name) has read and execute permissions on the directory, but not write permissions. The third group of three letters “r-x” indicates that any other user or group on the system has read and execute permissions. Note that, for directories, the access properties apply only to the directory itself and not any of its contents. These will have their own access properties.

To change the access properties, we use the command “chmod xxx <somefile>”. The “xxx” can be one of two things. It can either be three numbers indicating the octal representation of the nine access property bits discussed earlier, or it can take the form of:

<o=owner, g=group, a=all><+ or -><r=read, w=write, x=execute>

to add or revoke a specific permission to the owner, group, or everyone. For example, “chmod o+x <somefile>” adds execute permissions to the owner of that file. If you do not have write permission for the file or folder you are attempting to change, “sudo” will need to be used.

* 1. A shell is the command syntax that is used for the terminal input. For the case of Ubuntu, the “bash” shell is used, but other shells also exist and can be installed if desired. A shell script is a set of shell commands within a file that is executed. It can contain any of the shell commands normally available, including conditional statements, loops, and console feedback.

1. In addition to the Linux tutorials, the team also completed ROS tutorials 1-6. For some of us, this was the first time using ROS, so it was an introduction to the interface and the filesystem. Completing these tutorials enabled us to answer the following questions:
   1. The three layers of the ROS hierarchy are nodes, packages, and workspaces. Nodes are programs that run as a single process and have a particular purpose. Packages are groups on nodes. Workspaces are directories. They are called catkin workspaces in ROS. (Referenced Module A-4 slides)
   2. Compared to Linux, navigating the ROS filesystem uses similar but different commands. For example, instead of ‘ls’ and ‘cd’ as in Linux, to navigate the ROS system ‘rosls’ and ‘roscd’ are used. There is another tool called ‘rospack’ which can be used to find information about ROS packages. In the tutorial we used the ‘find’ function of ‘rospack.’
   3. According to the Module A-4 slides, the four items are defined as follows. A topic is a channel where messages can be passed between nodes. A message is the data passed. A publisher sends messages, and a subscriber receives messages. Both publishers and subscribers are nodes. In the turtle example, the topic allows the nodes ‘turtlesim\_node’ and ‘turtle\_teleop\_key’ to communicate. The ‘turtle\_teleop\_key’ is the publisher because it sends data containing keystrokes to move the turtle. The ‘turtlesim’ is the subscriber which receives this data. The keystrokes themselves are the message.
   4. The example uses 8 terminals. We found it helpful to open new tabs in one Ubuntu terminal window, instead of opening new terminal windows. It is easier to keep track of them as tabs. The terminals are: roscore, running the turtlesim\_node, running turtle\_teleop\_key to drive the turtle, starting the graph (rqt\_graph), using rostopic echo, using rostopic functions (list, type, pub, hz), using rostopic echo again to see published data, and starting the plot (rqt\_plot). Pressing Ctrl+C will exit the process without closing the terminal.
   5. The command catkin\_make is run to create a catkin workspace. In the tutorial, we created the directory callede catkin\_ws and ran catkin\_make in it. The command catkin\_create\_pkg creates the beginner\_tutorials which contains package.xml and CMakeLists.txt. CMakeLists.txt contains instructions for CMake to build packages. It contains the version, name, dependency requirements, python support option, message generators, packages/libraries/executables to build, tests to build, and rules for installation. The CMakeLists.txt instructs CMake and can generate build systems, while the C++ makefile is the actual build system.
   6. The $source commands configure the current ROS session to set up the environment variables correctly [4]. It essentially ensures that ROS knows where to find all the required resources. Without sourcing, there may be issues with finding certain resources. The first $source command in the tutorial is $source /opt/ros/kinetic/setup.bash.
   7. We had trouble getting the plot to work using the command given in the tutorial, but found a solution in the ROS forums [3]. We did not watch the tutorial video but found that working through the tutorials and reading the information was helpful in understanding the basics of how to use ROS.

**Conclusion**

This lab was an introduction to the race car and associated equipment and software that we will use throughout the rest of the course. While installing host machine software packages including Ubuntu 18.04 and Robot Operating System (ROS), the team became comfortable with the bash command line and the ROS filesystem. By collaborating on race car configuration and VESC motor tuning, the team discovered how to ensure optimal performance going forward. While the Jetson is not currently flashed, the team will work hard to catch back up to the rest of the class. Since we were already somewhat familiar with Linux and its operation, the main things we learned in completing this lab were the structure and usage of the ROS operating system and the NVidia and VESC tools used to work with the Jetson and motor controller. We also gained a much thorough understanding of the car hardware and how we will be working with it for the rest of the course.

**References**

1. <https://itsfoss.com/install-ubuntu-1404-dual-boot-mode-windows-8-81-uefi/>
2. <https://forums.developer.nvidia.com/t/flash-sh-failing-with-cpu-bootloader-is-not-running-on-device-message-with-jetpack-4-2/72762/10>
3. <https://answers.ros.org/question/59251/rqt_plot-not-plotting-data/>
4. <https://answers.ros.org/question/252715/what-is-the-point-of-sourcing-setupbash-file-during-installation-of-ros/>