



TRAINEE SOFTWARE MODULE (2022-23)

Ubuntu 20.04 Dual Boot (Feb 18-20):

To start with this module, we want you to dual-boot **Ubuntu 20.04 (20.04 only)** first.

Step 1: If Mac - Buy a windows laptop :P. (Get in touch.)

Step 2: <https://youtu.be/Ptqy8aLi8aY>

The above video is fairly clear for the installation, in case of doubts, please reach out to us.

Now that you have Ubuntu (working preferably), no more hunting through menus or clicking through windows, you can manage files, and install software without leaving the command line.

Go through the following tutorial to get used to basic Linux commands:

<https://www.freecodecamp.org/news/the-linux-commands-handbook/>

Checkpoint:

After this, you should be comfortable using the following commands:

`cd , ls , pwd , sudo , mkdir , rm , mv , cp , cat , echo , chmod`

Github (Feb 21-22):

If you already have the knowledge of how *git* works, good!
Else, no issues, we have got you covered.

Follow this [tutorial](#) to know how to use git from your terminal.

You might also want to check out this cheat sheet:

<https://github.com/mikeizbicki/ucr-cs100/blob/2015winter/textbook/cheatsheets/git-cheatsheet.md>

Checkpoint:

Try out this fun git game that walks you through using git commands.
Doing it up to level 6 is recommended, above that is up to your enthusiasm.

<https://github.com/git-game/git-game>

FS Driverless Simulator (Feb 23-25):

You will be required to install the Formula Student Driverless Simulator, where we validate our software stack.


Official documentation:

<https://fs-driverless.github.io/Formula-Student-Driverless-Simulator/v2.1.0/>

Download and run the following bash file ([how you run a bash file](#)):
<https://github.com/dv-software-22-23/Installation/blob/main/Setup.sh>

Running FSDS alone:

1. `cd fsds-v2.2.0-linux`
2. `./FSDS.sh`

Now you see the simulator launched, click on run simulation and you can move the simulator with your  arrow keys.

Launching ROS bridge :

1. Run FSDS as above
2. `cd Formula-Student-Driverless-Simulator/ros`
3. `source devel/setup.bash`
4. `roslaunch fsds_ros_bridge fsds_ros_bridge.launch`


Checkpoint:

List the topics published and notice how sensor data is published.

Python & Required Libraries (Feb 26-28)

Python is the programming language we majorly use.

It is suggested to use IDEs like VSCode, you can follow this video for installing:

 How to Install and Use Visual Studio Code on Ubuntu 20.04 LTS Linu...

Those who are new to python can refer to -

<https://www.w3schools.com/python/>

Those familiar with python can refer to -

1. <https://www.w3schools.com/python/numpy/default.asp>: Numpy
2. <https://www.w3schools.com/python/pandas/default.asp>: Pandas
3. https://www.w3schools.com/python/matplotlib_intro.asp: Matplotlib

Checkpoint:

Python will be used extensively so keep continuing to practice. Reach out to us once you have done the basic exercises above for more.

ROS (March 1-4)

What is ROS? ROS (Robot Operating System) is a framework used in robotic applications to build and interconnect code. It provides a set of libraries and tools that allow developers to create software for robots, making it easier to build, test, and deploy new robotic systems. This is the standard resource we use for [ros tutorials](#), Follow the order as it is.

Checkpoint:

A) Familiarity with -

- Catkin workspace (ROS workspace)
- Rospack
- Ros package (creating) (observe CMakeLists.txt and package.xml in a package when created, what are they for?)
- Sourcing the setup file (source devel/setup.bash)
- catkin_make
- What is a rosnode (what are the types?)(publisher and subscriber)
- What are rostopics and ros messages?

B) Handshake Assignment:

a. Make 2 nodes: VCU and AI.

Note: VCU represents the vehicle control unit, and AI is the a computer that processes our autonomous software.

b. These nodes will use 2 topics to communicate: Name these topics as – “VCU2AI” and “AI2VCU”.

c. Aim: VCU needs to check continuously if the computer (AI) is working or not. Devise a method to ensure this. You need to write blocks of codes in VCU and AI nodes to implement this method.

Final Assignment (March 5-6):

This is a mixed assignment testing your basic understanding on git, fsds, python, matplotlib and ros.

Problem Statement:

Various sensors are attached to the car in the Formula Student Driverless Simulator. Consider the LiDAR sensor.

(Let's say Lidar1 specifically, refer settings.json
"./Formula-Student-Driverless-Simulator/settings.json")

Now, we want to collect the LiDAR [point cloud](#) using ROS (recall rosbridge) and store it in a npy file and visualise the data.

You might want to think along these lines :

- To get view/access lidar data, which topic out of all the published topics do I publish or subscribe?
- Which node should I create? Publisher or subscriber?
- How to save the data into a npy file>
- How to load data from npy file and visualize it?

THE END
