RIDWAN HOSSAIN
100747897

2022/06/20

- Today will be focused on rehearsing and getting the bench ready for the demo on Wednesday.

- **Things to Talk About:**
    - Introduce myself, I'm an undergraduate research assistant working under Dr. Azim. So I'll be showing a little bit about this autonomous vehicle simulation bench. This system is using the open source software called Carla, which has been developed from the ground up to support development, training, and validation of autonomous driving systems. So this platform supports flexible specification of sensor suites, environmental conditions, full control of all static and dynamic actors, maps generation and much more.

    - In addition to autonomous vehicle simulation, the software also allows for the manual control of vehicles. Out of the box, the simulator can be manually controlled using the keyboard, but we felt that creates a gap between what one would experience in real life. In order to bridge that gap, multiple hardware peripherals such as a steering wheel, pedals, gauge cluster, shifter, and an infotainment system were added to form the bench we have here. The hardware peripherals connected to the simulator provide a much more familiar and cohesive interface to the software system, including elements and characteristics of a real vehicle. A user can sit down and just start driving using all the peripherals here. (**demo some driving around, then with traffic, and with autopilot as well**)

    - One promising aspect of the bench is the android head unit I mentioned before, which utilizes something called an Android debugging bridge, or ADB for short. The ADB interface allows the android head unit to communicate with the simulator software. Some possibilities the ADB allows for is that it can perform various vehicle control functions that occur in the simulation software from the head unit. I've mapped some simple functions such as opening certain applications using buttons on the steering wheel. **(Open google maps)**.

    - One highly useful aspect of this communication interface is that I could develop some code in my autopilot algorithm that obtains google maps data from the android head unit, and uses that information to guide the self driving agent. But beyond that, the ADB can also be used for application and software testing and development of all sorts. Let's say I develop an application that is used to easily access vehicle maintenance data. I can communicate from the simulator to the android application in order to retrieve that information.

- Another feature of the simulation bench is the integration of a CAN bus. For some background, a CAN (Control Area Network) bus is a vehicle bus designed to more easily allow microcontrollers and devices to communicate with each other's applications. The CAN bus which can be used to communicate with the software and send data to the vehicle control subsystems. CAN buses are especially prevalent in Autonomous vehicle systems, as they have even more subsystems that all need to be communicating with one another. The CAN was devised to fill this need. One key advantage is that interconnection between different vehicle systems can allow many features to be implemented using software alone, saving time and money on creating complex, hard-wired systems to handle that communication. That same kind of network can be simulated here.

- The CAN bus on this bench can be used to retrieve some data from the simulator, such as individual wheel speeds, the steering angle, and so on. (**show candump**)

- That data can also be communicated to the vehicle to be interpreted as commands, which leads me to my next topic. A gleaming weak spoty of CAN networks is security vulnerabilities. Those aspects can be simulated and tested using the CAN bus integrated in the bench **(Show demo of CAN attack)**

- So this is how this addition to the software can be used to develop upon and strengthen a CAN network's integrity.

- A very important feature added thus far in regards to supporting development and testing is the data logging pipeline. This feature can be used to collect a slew of data, everything from vehicle telemetry to the behavior of autonomous functions. **(Show run from dataset1 in Town06 [lane swerving], the data logs recorded, and some of the visualized data) (Do the same for dataset8 in Town10HD_Opt [obstacle detection])**.

- This feature can be used to help development and testing when creating autonomous vehicle algorithms, or how certain software features affect the physical behavior of the vehicle, etc.

- There is also a test suite which is being created, and can be used to ensure software aspects of the system perform to standard, and can be easily modified and expanded. **(Go over test folder briefly and discuss some tests I made)**

- One of the future goals is to automate the testing suite to adhere to better continuous integration and testing. We are going to look to build a framework which upon making changes to the software system and pushing any new features, the framework can run a suite of tests in an automated manner to further streamline development on the bench.

- The last topic I'd like to briefly discuss is, as it stands, the maps on this bench are limited to the maps the software is packaged with. However, the ability to develop custom maps is here and the software exists to design those maps and export them to the simulation bench. One of the things we want to do is reproduce Oshawa roads using that software in conjunction with google earth provided 3d models and export them to the bench in order to perform all forms of development and testing activities in real local roads.