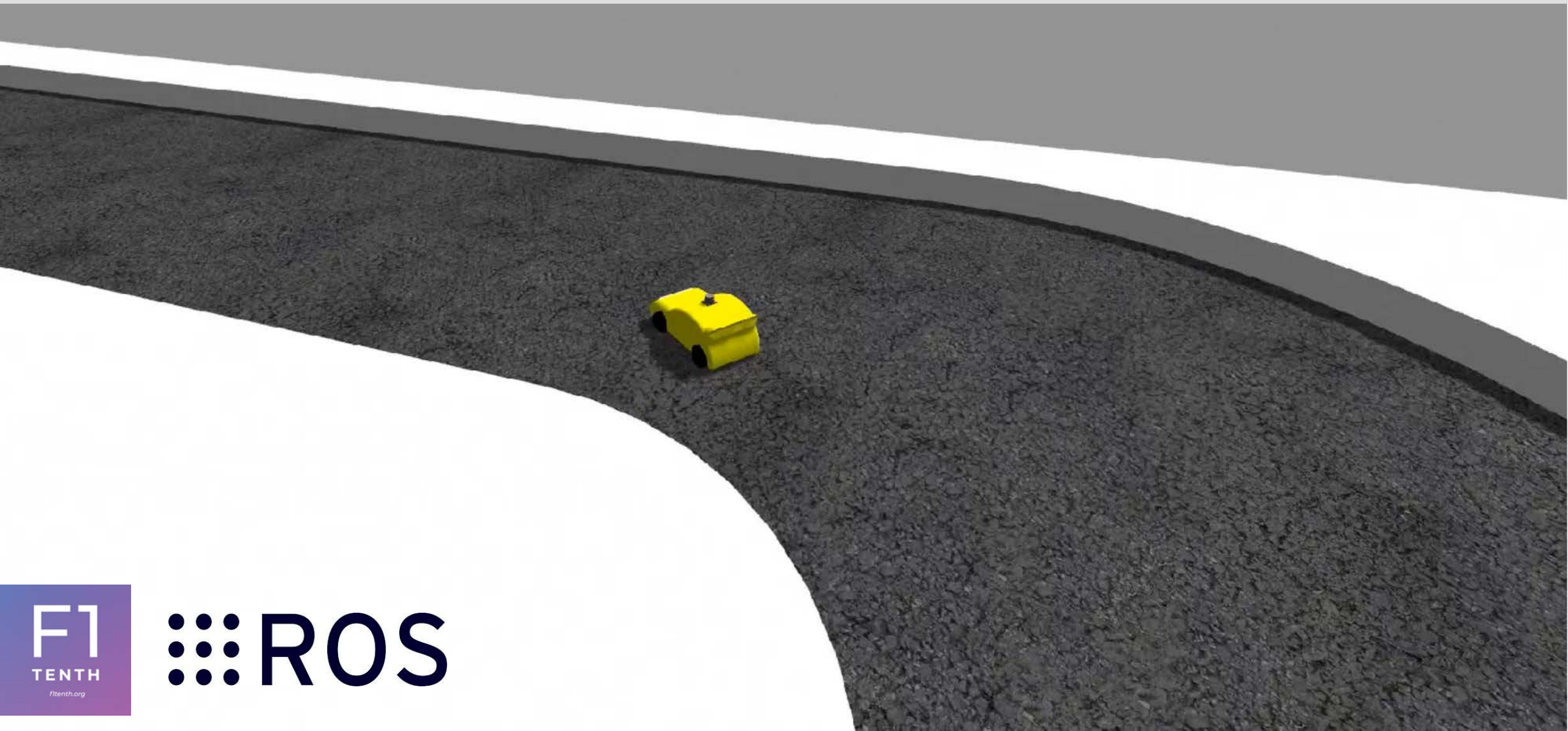
# ROS F1/10 Autonomous Racing Simulator



F1 TENTH
f1tenth.org

ROS

# Two ways to access

1) Standalone installation:
    1) Download and install from f1tenth.dev
    2) Tutorials available on the website

2) Access shared online version:
    1) This video


The ROS F1/10 Autonomous Racecar Simulator is a Gazebo based virtual racing environment which includes a realistic model of the F1/10 autonomous racecar and associated race controllers. Otimize autonomous racing algorithms and design and visualize head-to-head racing with multiple autonomous racecars.

Github: Click Here For Intallation And Tutorials

## Presentation from ROSCon 2019 [Macau].


If you use the simulator for your work, please cite the paper below:

# Installation

UVA F1/10 Group edited this page on Oct 21, 2019 · 1 revision

## Installation and Setup

The simulator has been tested on a variety of hardware configurations, and while Gazebo and ROS dependencies work with bare minimum hardware requirements, the essence of an autonomous racing simulator requires a real-time factor of at-least 0.85. For this reason, we suggest that your local machine meet the requirements stated in the following table.

| | Suggested Minimum Requirements |
|---|---|
| Operating System | Ubuntu 18.04 (Bionic) 64-bit |
| Memory | 16 GiB DDR4 |
| GPU | NVIDIA GTX-1660 |

The simulator itself depends on libraries, packages and objects from ROS and Gazebo. The current version of the simulator uses ROS Melodic and Gazebo 9; along with certain other libraries and their python bindings. Before proceeding with the installation steps in this document, check if your local machine already has the full-desktop version of ROS Melodic installed. If not, follow the instructions provided in this link. When you get to section 1.4 of the ROS installation tutorial, choose the first option: `ros-melodic-desktop-full`. Your local machine will now have the basic dependencies installed. The next steps of the installation tutorial are critical and must be followed in the shown sequence.

First, we begin by installing the ROS and Gazebo packages that are not installed by default using the above method. This includes SLAM packages, ROS navigation and the MIT GPU particle filter. Some of these packages have a Debian installation candidates and the others have to be downloaded as source codes and locally compiled. We install the Debian packages first; open a new terminal and enter the



Clone this wiki locally

https://github.com/f1te


Spring 2020

# Getting Started with F1/10 Online Simulator

**Madhur Behl** **Varundev Sukhil** **Sandesh Banskota** **Mert Karakas**

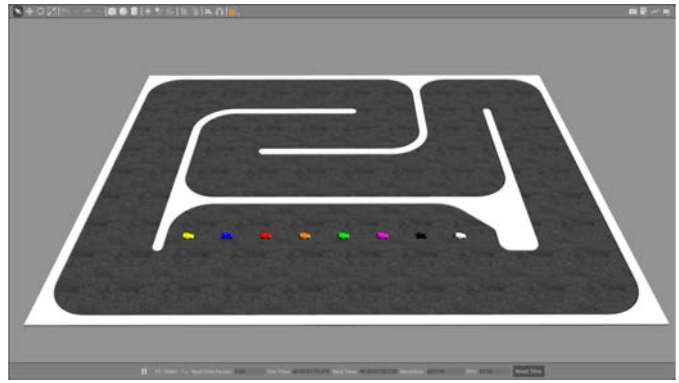
# Critical Steps (chronological)

1. Connect to UVA Anywhere VPN
2. Host/VM ROS network configuration
3. CS GPU server #5 & team workspaces
4. Host/VM remote visualization package
5. Remote tele-operation & best practices

FAA University of Virginia

# Critical Steps (chronological)

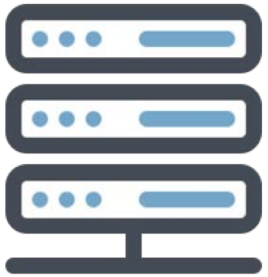
1. **Connect to UVA Anywhere VPN**
2. Host/VM ROS network configuration
3. CS GPU server #5 & team workspaces
4. Host/VM remote visualization package
5. Remote tele-operation & best practices



ROS over Network visualization

Remote Machine
VM Client

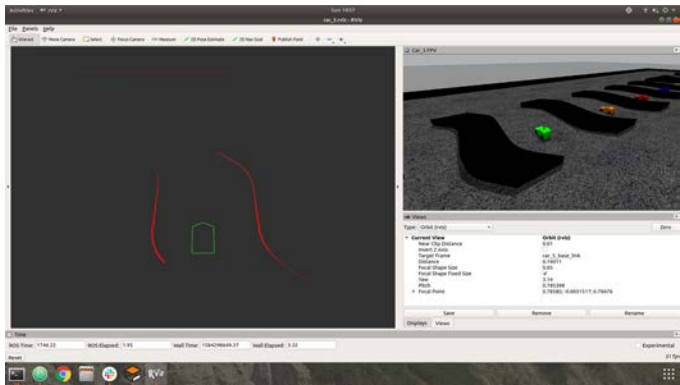

UVA VPN
Connection



Computer Science
gpusrv05
dedicated server.

Ubuntu 18.04
ROS Melodic
Gazebo 9

Secure ssh for interaction



# Connect to UVA Anywhere VPN

**Ubuntu Host or Dual Boot** (preferred method)
Follow steps in link - https://at.virginia.edu/39f3Xs6

**Windows/macOS**
Follow steps in link - https://bit.ly/2WBWdhi

For both Linux and Windows/macOS follow each step exactly as described.

# Connect to UVA Anywhere VPN

## Important Note

The IPv4 address assigned to each address via the UVA Anywhere VPN changes every time your computer connects to the network.

**TODO –** note the IPv4 address assigned every time, this becomes important later

FAA University of Virginia

# Critical Steps (chronological)

1. Connect to UVA Anywhere VPN
2. **Host/VM ROS network configuration**
3. CS GPU server #5 & team workspaces
4. Host/VM remote visualization package
5. Remote tele-operation & best practices

# Host/VM ROS network configuration

**For both Ubuntu Host and Ubuntu VM**

Edit the ~/.bashrc file to add/modify the following lines

```
export ROS_MASTER_URI=http://gpusrv05.cs.virginia.edu:11311
export ROS_IP=<your_UVA_Anywhere_VPN_IPv4_address>
```

The UVA Anywhere IPv4 address is the same obtained from the previous step, and has to be **changed every time you connect to the VPN**, otherwise you will not be able to connect to the F1/10 Online Simulator.

FAO University of Virginia

# Critical Steps (chronological)

1. Connect to UVA Anywhere VPN
2. Host/VM ROS network configuration
3. **CS GPU server #5 & team workspaces**
4. Host/VM remote visualization package
5. Remote tele-operation & best practices

# CS GPU server #5 & team workspaces

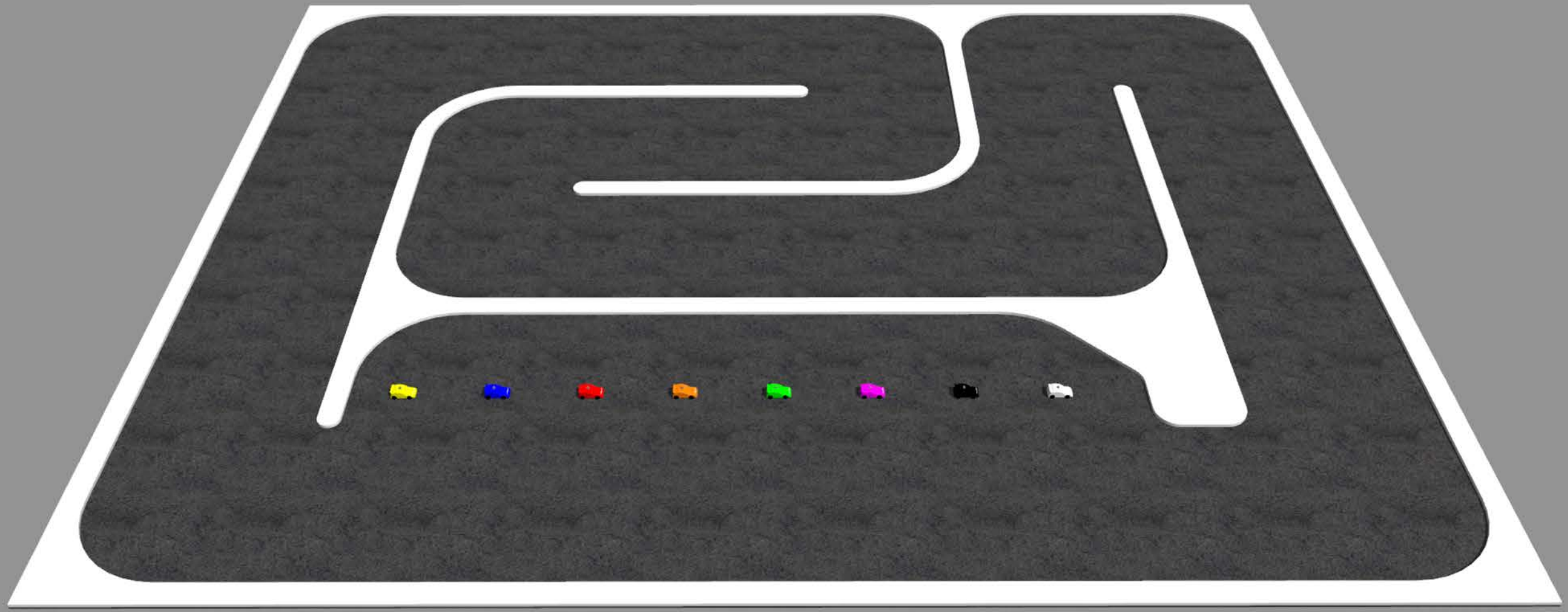
The F1/10 Online Simulator is hosted on one of the CS Dept. GPU Server and is made available for the class for the rest of the semester.

The simulator is **always active** and all your team cars are present on the **same track**.

Each team is assigned a **team workspace** where you are expected to place your assignment codes and project codes. You will not have access to other team's workspace.

The server is located at `gpusrv05.cs.virginia.edu` and can be accessed only by using the UVA Anywhere VPN – no exceptions can be made to this rule.

Each team is assigned one car under the `/car_<team_number>` namespace



FIAO University of Virginia

# Critical Steps (chronological)

1. Connect to UVA Anywhere VPN
2. Host/VM ROS network configuration
3. CS GPU server #5 & team workspaces
4. **Host/VM remote visualization package**
5. Remote tele-operation & best practices

# Host/VM remote visualization package

The Gazebo GUI cannot be shared with the teams due to UVA network restrictions.

A new F1/10 centric visualization package has been released to help every member of every team to visualize and control the virtual racecar. To build the package on your Ubuntu host/VM:

- `$ cd ~/catkin_ws/src/`
- `$ git clone https://github.com/linklab-uva/sim-tools`
- `$ cd ~/catkin_ws && catkin_make && source ~/.bashrc`

Confirm that your local `catkin_ws` workspace is sourced in `~/.bashrc`

# Host/VM remote visualization package

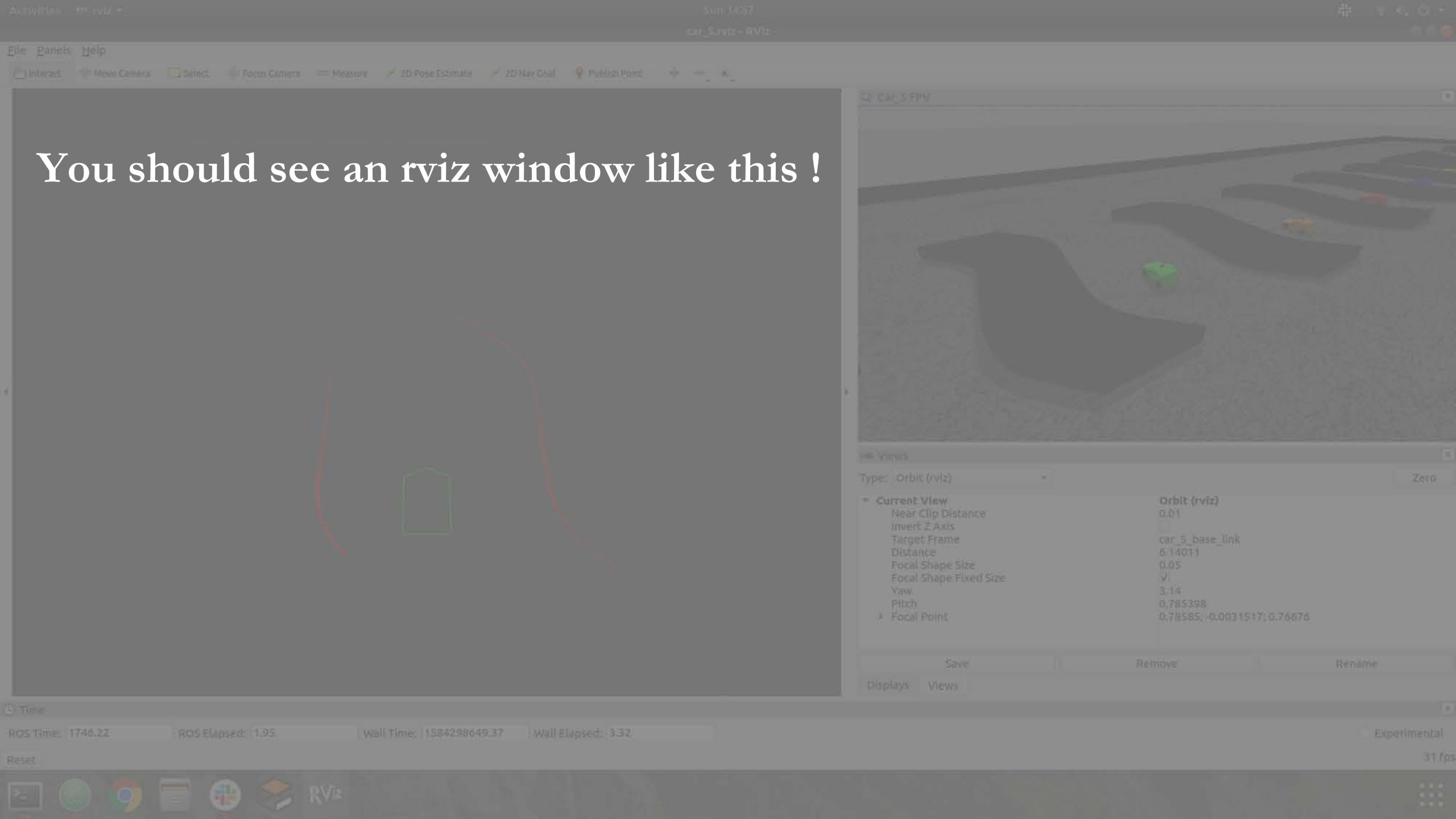
## Test and First Visualization of F1/10 Virtual Racecar

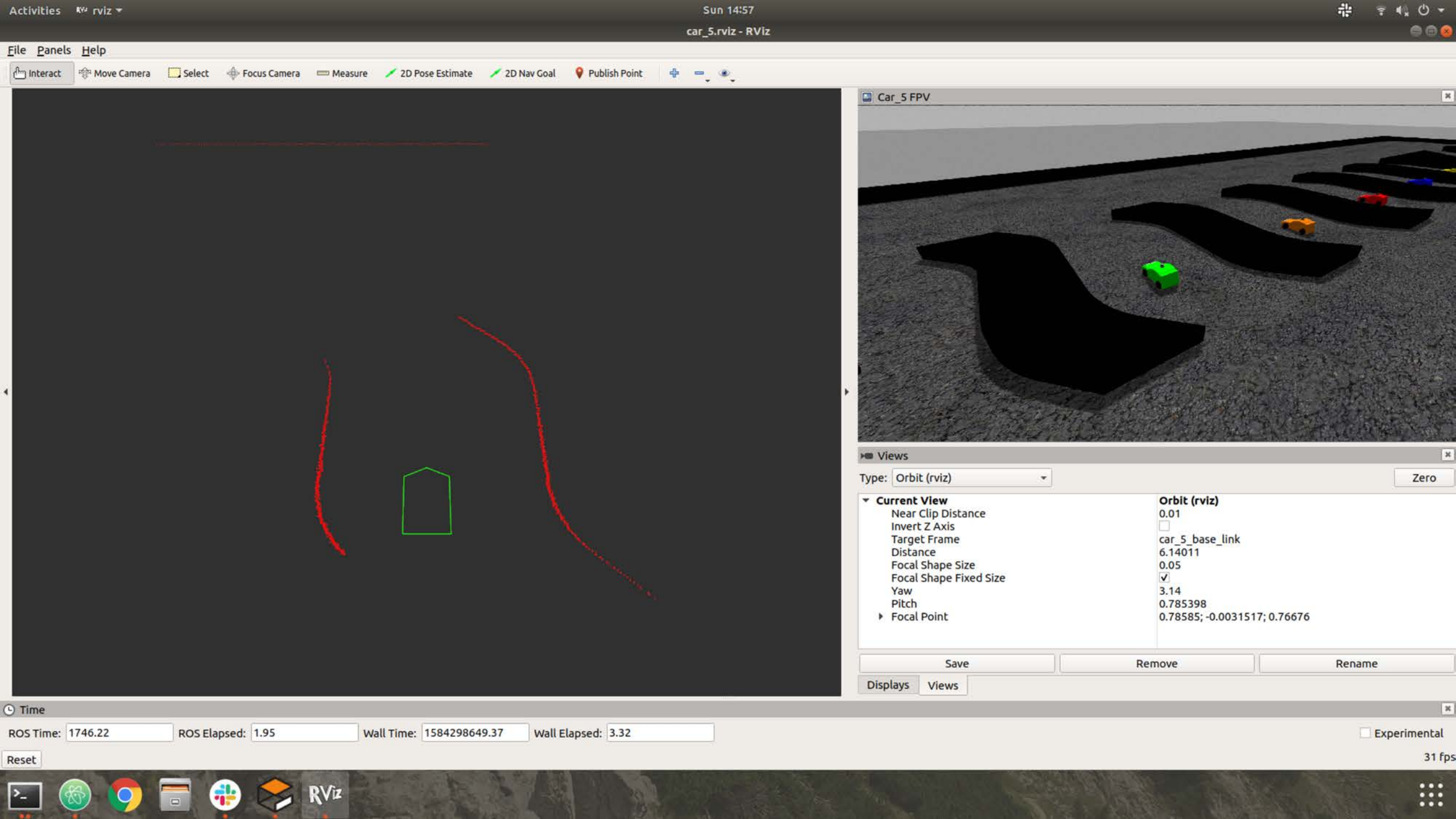
On your local Ubuntu host/VM, open a new terminal and enter the following command:

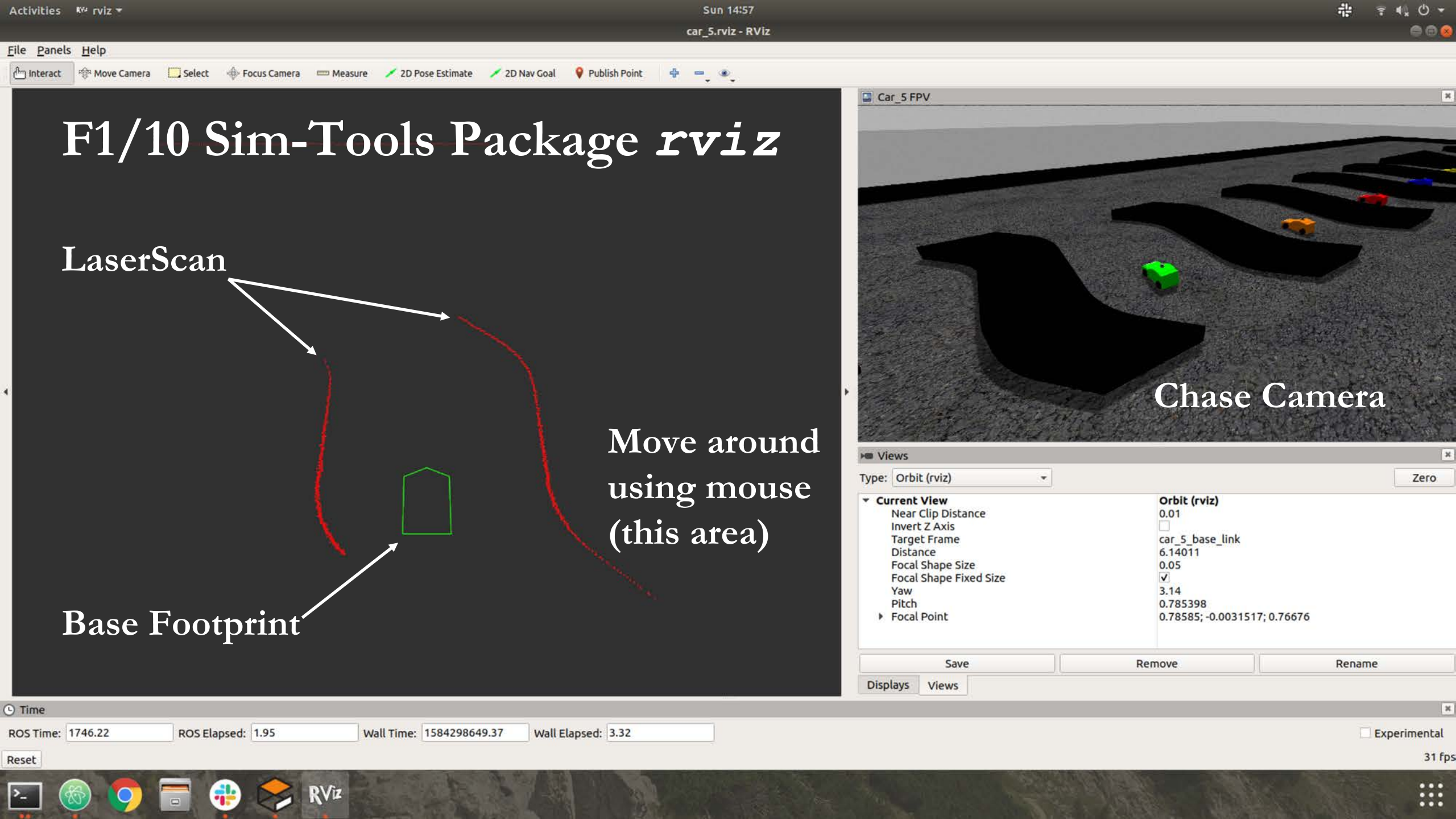
```
$ roslaunch sim-tools remote-access.launch
      uva_ID:=<your_UVA_ID>
      car_name:=car_<your_team_number>
      visualize:=true
      remote_teleop:=false
cr
```

**Example:**

```
$ roslaunch sim-tools remote-access.launch uva_ID:=abc5xyz
      car_name:=car_5 visualize:=true remote_teleop:=false
```


You should see an rviz window like this !
File Panels Help
Interact Move Camera Select Focus Camera Measure 2D Pose Estimate 2D Nav Goal Publish Point
car_5 FPV
Views
Type: Orbit (rviz)
Zero
Current View
Orbit (rviz)
Near Clip Distance
0.01
Invert Z Axis
Target Frame
car_5_base_link
Distance
6.14011
Focal Shape Size
0.05
Focal Shape Fixed Size
Yaw
3.14
Pitch
0.785398
Focal Point
0.78585; -0.0031517; 0.76676
Save
Remove
Rename
Displays
Views
Time
ROS Time: 1746.22
ROS Elapsed: 1.95
Wall Time: 1584298649.37
Wall Elapsed: 3.32
Experimental
Reset
31 fps
RViz


Activities
rviz
Sun 14:57
car_5.rviz - RViz
File
Panels
Help
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Car_5 FPV
Views
Type:
Orbit (rviz)
Zero
Current View
Orbit (rviz)
Near Clip Distance
0.01
Invert Z Axis
Target Frame
car_5_base_link
Distance
6.14011
Focal Shape Size
0.05
Focal Shape Fixed Size
Yaw
3.14
Pitch
0.785398
Focal Point
0.78585; -0.0031517; 0.76676
Save
Remove
Rename
Displays
Views
Time
ROS Time:
1746.22
ROS Elapsed:
1.95
Wall Time:
1584298649.37
Wall Elapsed:
3.32
Experimental
Reset
31 fps
RViz


Activities
rviz
Sun 14:57
car_5.rviz - RViz
File Panels Help
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
F1/10 Sim-Tools Package rviz
LaserScan
Move around using mouse (this area)
Base Footprint
Car_5 FPV
Chase Camera
Views
Type: Orbit (rviz)
Zero
Current View
Orbit (rviz)
Near Clip Distance
0.01
Invert Z Axis
Target Frame
car_5_base_link
Distance
6.14011
Focal Shape Size
0.05
Focal Shape Fixed Size
Yaw
3.14
Pitch
0.785398
Focal Point
0.78585; -0.0031517; 0.76676
Save
Remove
Rename
Displays
Views
Time
ROS Time: 1746.22
ROS Elapsed: 1.95
Wall Time: 1584298649.37
Wall Elapsed: 3.32
Experimental
Reset
31 fps

FAA University of Virginia

# Critical Steps (chronological)

1. Connect to UVA Anywhere VPN
2. Host/VM ROS network configuration
3. CS GPU server #5 & team workspaces
4. Host/VM remote visualization package
5. **Remote tele-operation & best practices**

# Remote tele-operation & best practices

## Remote tele-operation (using keyboard)

Note – only one person per team can remotely control the virtual racecar

Log into the team workspace

```
$ ssh team_<your_team_number>@gpusrv05.cs.virginia.edu
$ password: will be sent to each team
```

Launch the teleoperation node

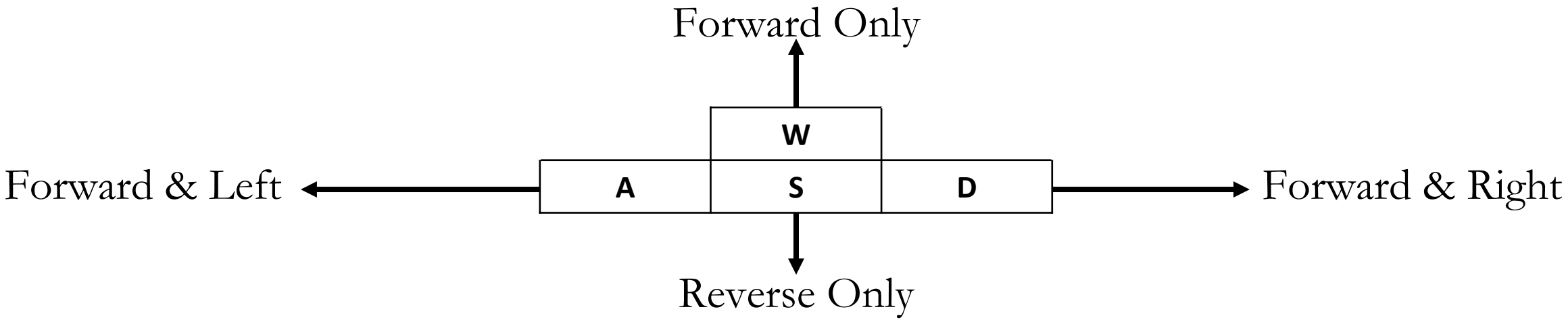
```
$ roslaunch sim-tools remote-access.launch
    uva_ID:=<your_UVA_ID>
    car_name:=car_<your_team_number>
    visualize:=false
    remote_teleop:=true listen_offboard:=false
```

# Remote tele-operation & best practices

**Remote tele-operation (using keyboard)**

Note – only one person per team can remotely control the virtual racecar

**Keep the terminal where the tele-operation node was launch active on the screen**




**Press 'Q' for emergency brakes**

# Remote tele-operation & best practices

## Best Practices

- Use the team workspace on the server to control the virtual racecar
- Use Ubuntu host/VM only to visualize the virtual racecar
- When in doubt, ask the instructors (this will save time for everyone involved)
- For new packages, ask TAs – absolutely no `sudo` access to students (CS policy)
- Play nicely with other teams (please, no intentional collisions)