Poly-verification User Guide

This document contains the information of how to run the simulation through the Poly-Verification Suit. Once the installation and setup completed you are ready to use the polyVerif framework.

If you have not setup and install the pre-requisite then please follow **Setup\_And\_Installation** document for the reference.

This framework uses predefined metrics for the validation of the stacks which will be calculated after running the test cases. On the basis of these metrices threshold the success/failure of the stack validated. Any user can set their own values by modifying the config.ini file in **adehome/Poly\_Suite/config.ini**. Below is the config.ini file parameters-

**[autonomous\_stack\_config]**

**detection\_max\_threshold=40 #MAX value for detection validation**

**detection\_min\_threshold=30 #MIN value for detection validation**

**control\_collion\_count=0 #Collision count**

**localize\_max\_threshold=5 #MAX value for localization validation**

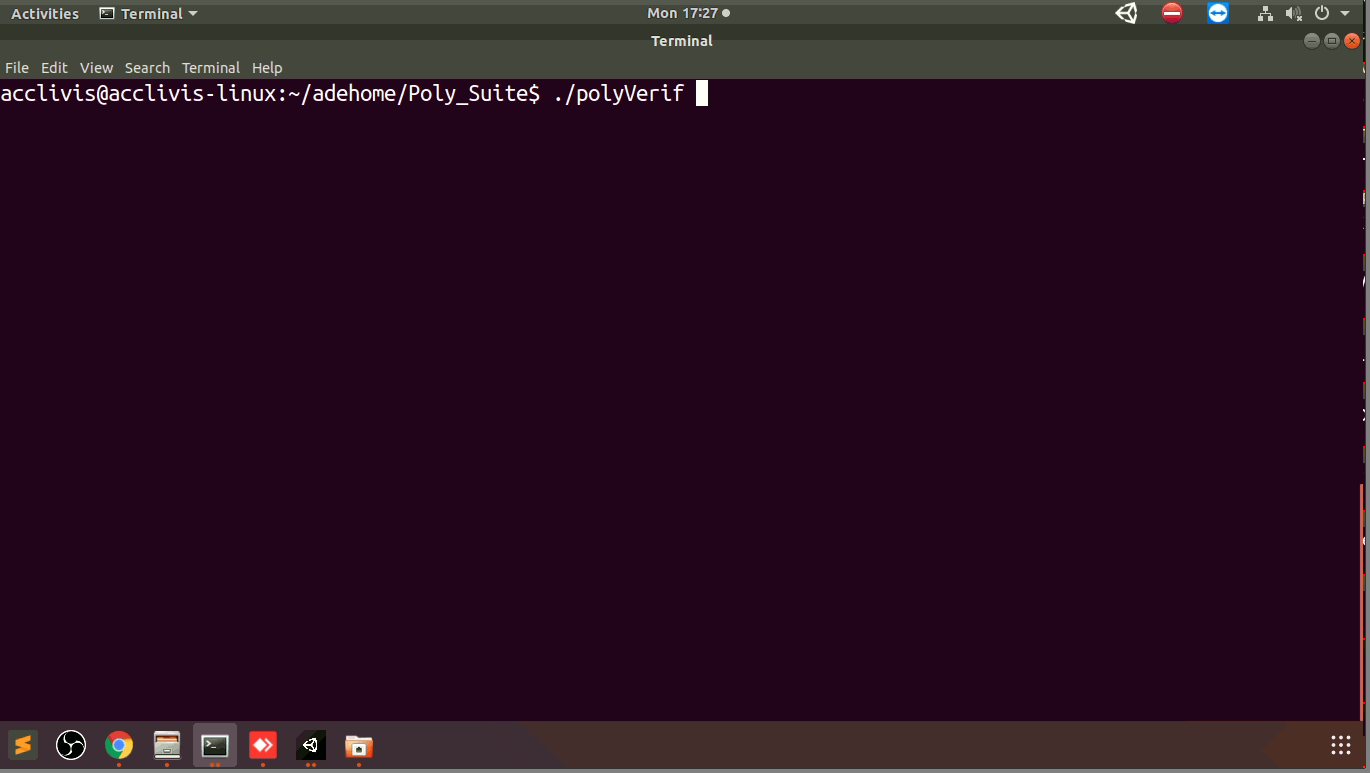
**localize\_min\_threshold=1 #MAX value for localization validation**

**planner\_goalpose\_max\_deviation=5 #MAX value of goal position deviation**

**planner\_goalpose\_min\_deviation=2 #MIN value of goal position deviation**

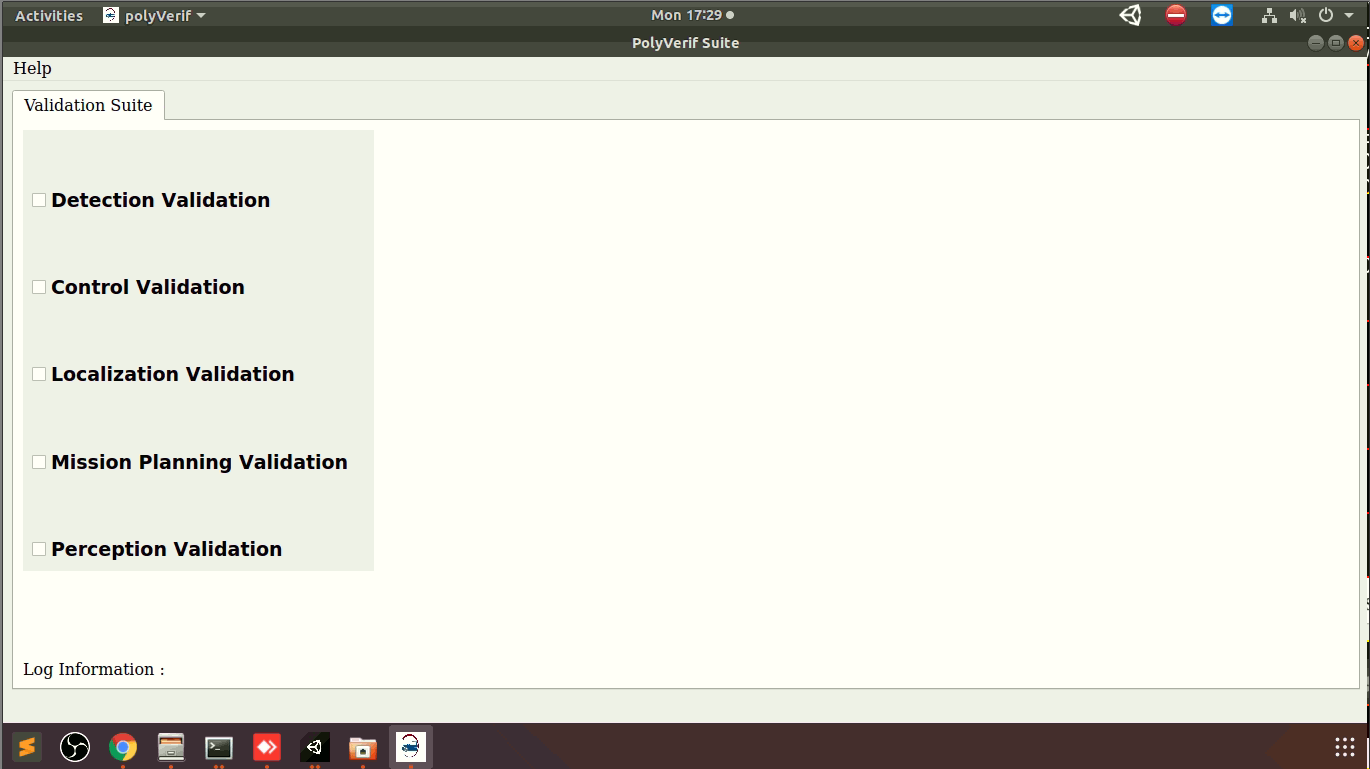
1. Go to the Poly\_Suite directory and run

**$ ./polyVerif**

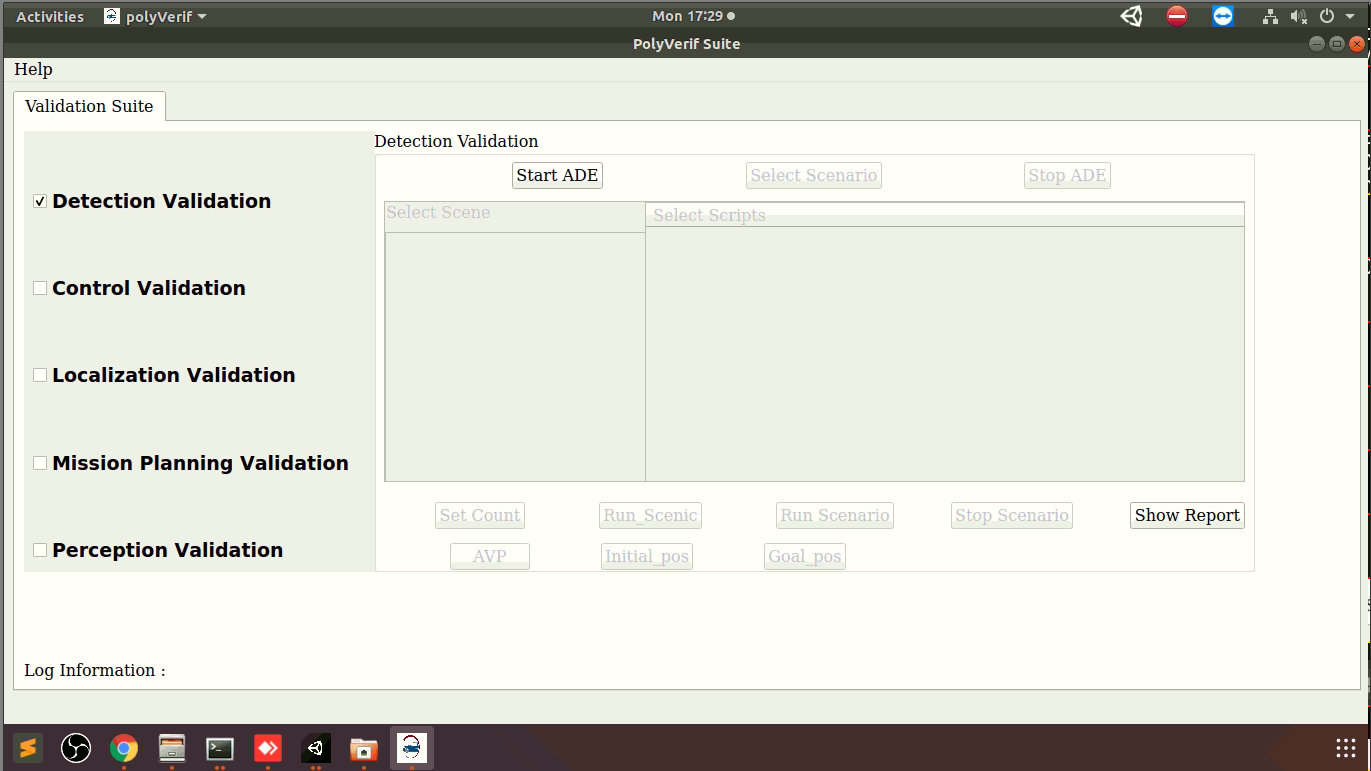


1. It will start the polyVerif framework
   1. As of now four validation is working-
      * Detection Validation
      * Control Validation
      * Localization Validation
      * Mission Planning Validation

Click any of them-



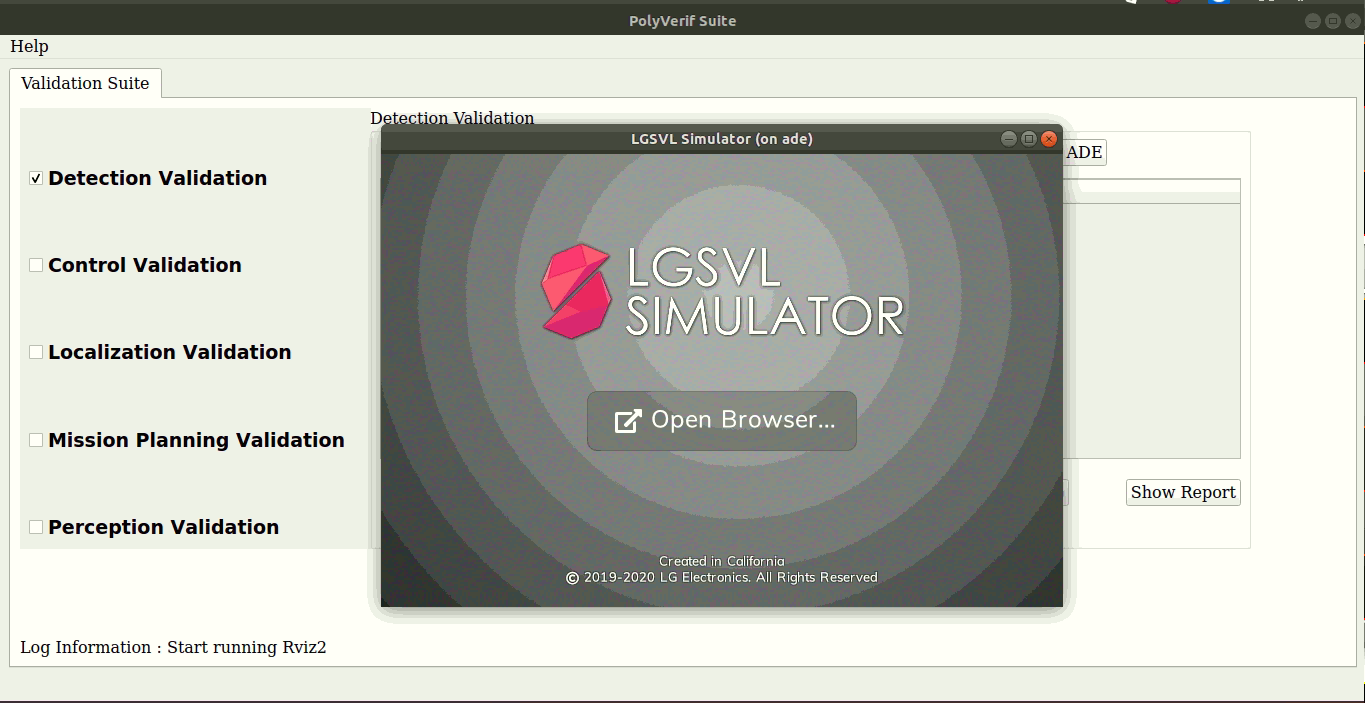
1. Detection Validation is for detecting the vehicle from perception stack.
   1. Click on the Start ADE button, it will start all the required module for example-
      * AutowareAuto
      * Perception Stack
      * Lgsvl simulator
      * Rviz
      * Ros2-lgsvl-bridge



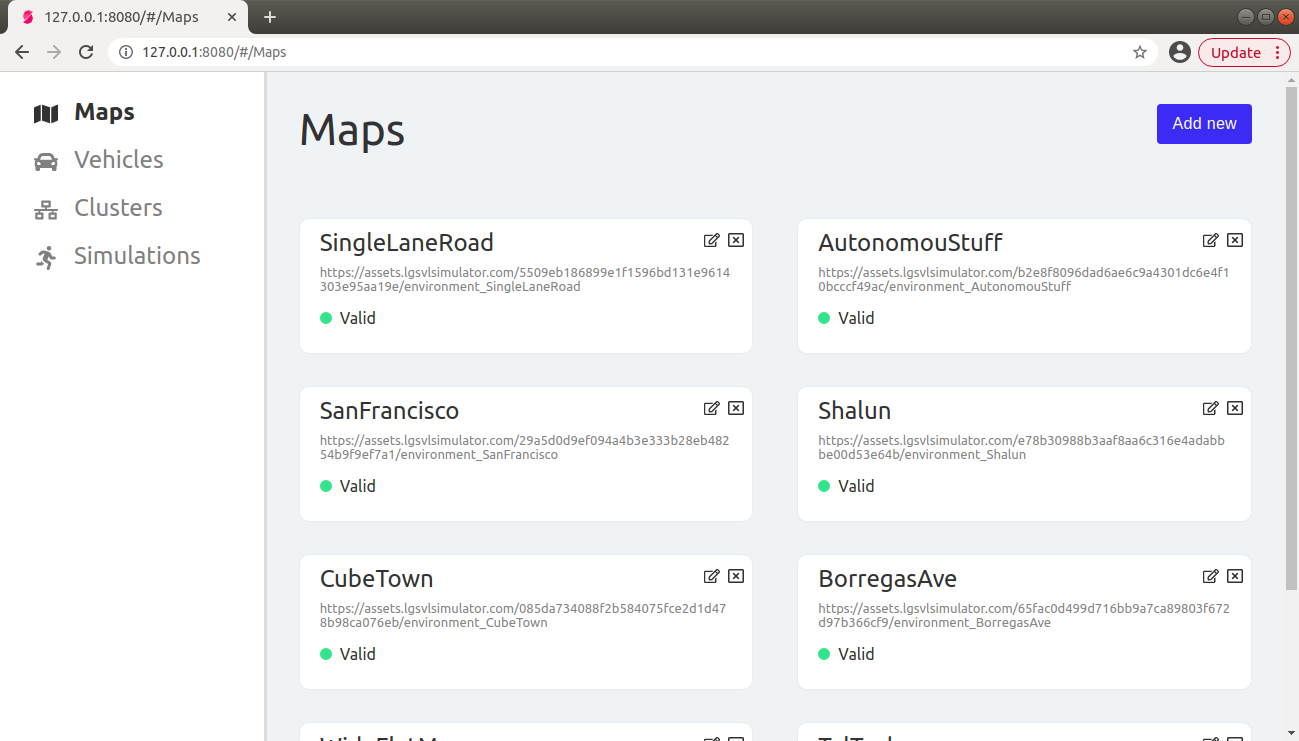
1. It will take some time to start the ADE docker and other components please wait for while



1. Once lgsvl simulator and rviz started, go to the any web browser

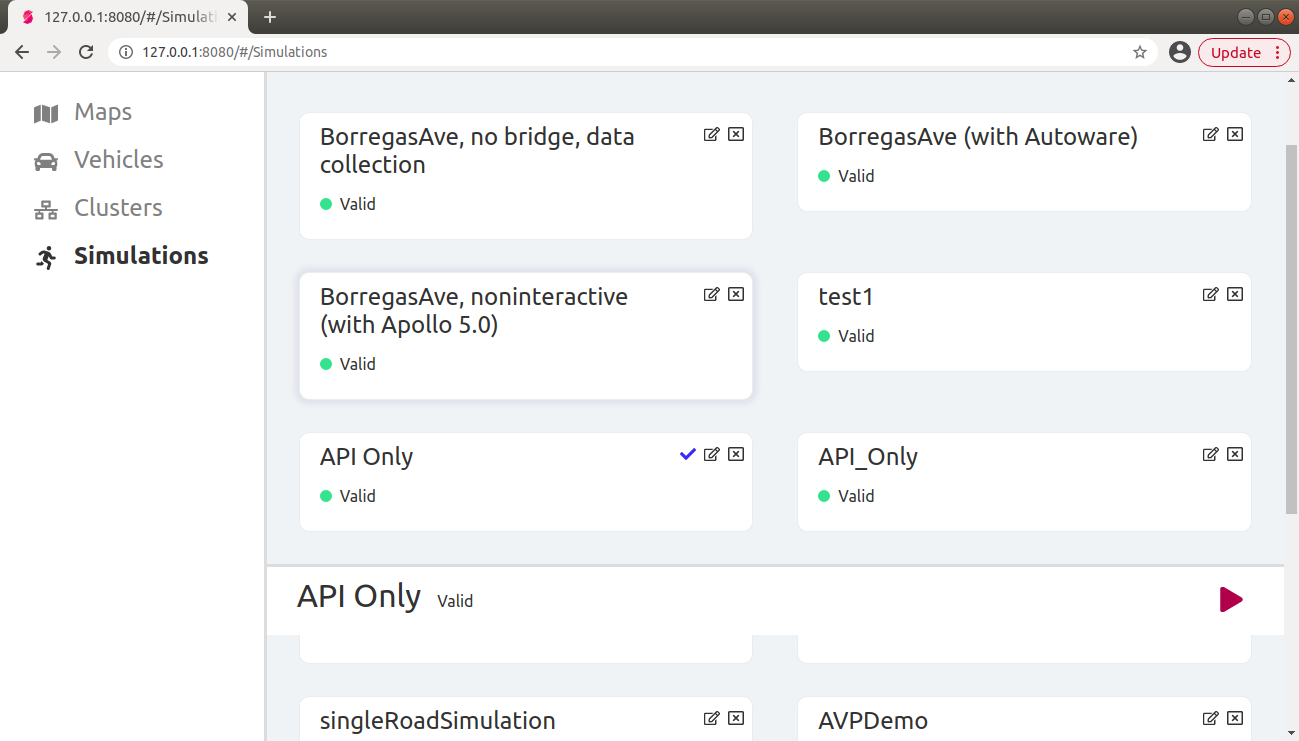


1. Enter localhost:8080 on address bar and hit enter.

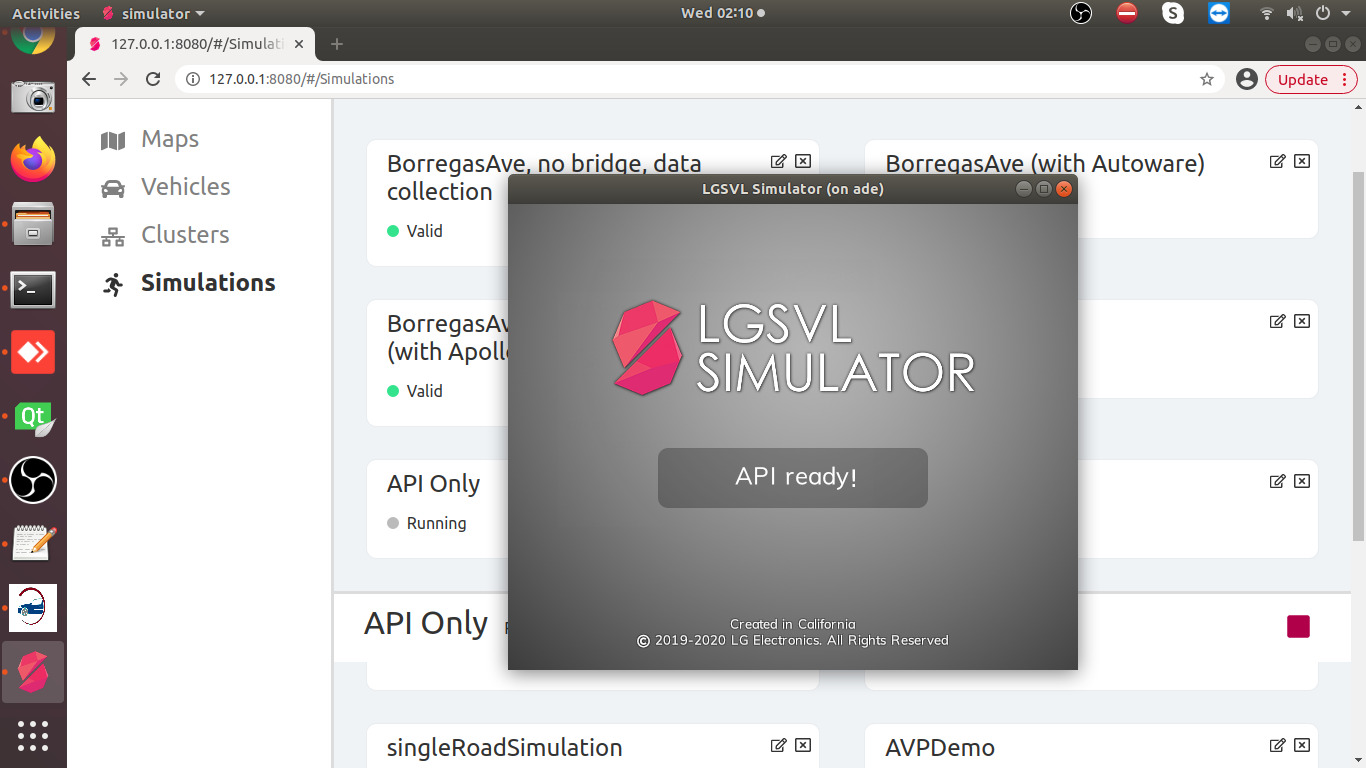


**Note: -** If the lg web page is not open the try below address also on address bar- 127.0.0.1:8080

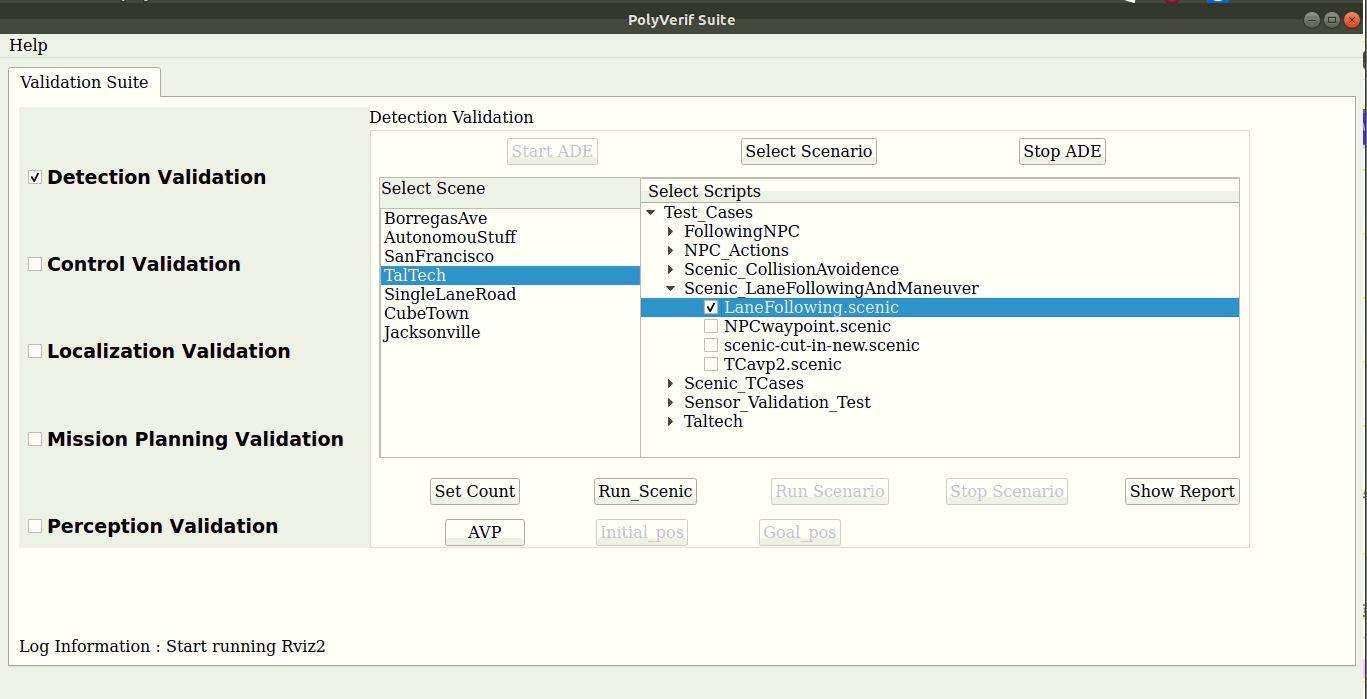
1. Click on the Simulations tab and select the API\_Only.
   1. Click on the Play button



1. Now the simulator is in the **API\_Mode**



1. Now click on the **Select Scenario** button
   * Select TalTech map from the Select Scene
   * Select any scenario from the Select Script list
   * The selected script in the below screenshot is written in scenic which will add an ego vehicle in the TalTech map which will controlled by the AutowareAuto.



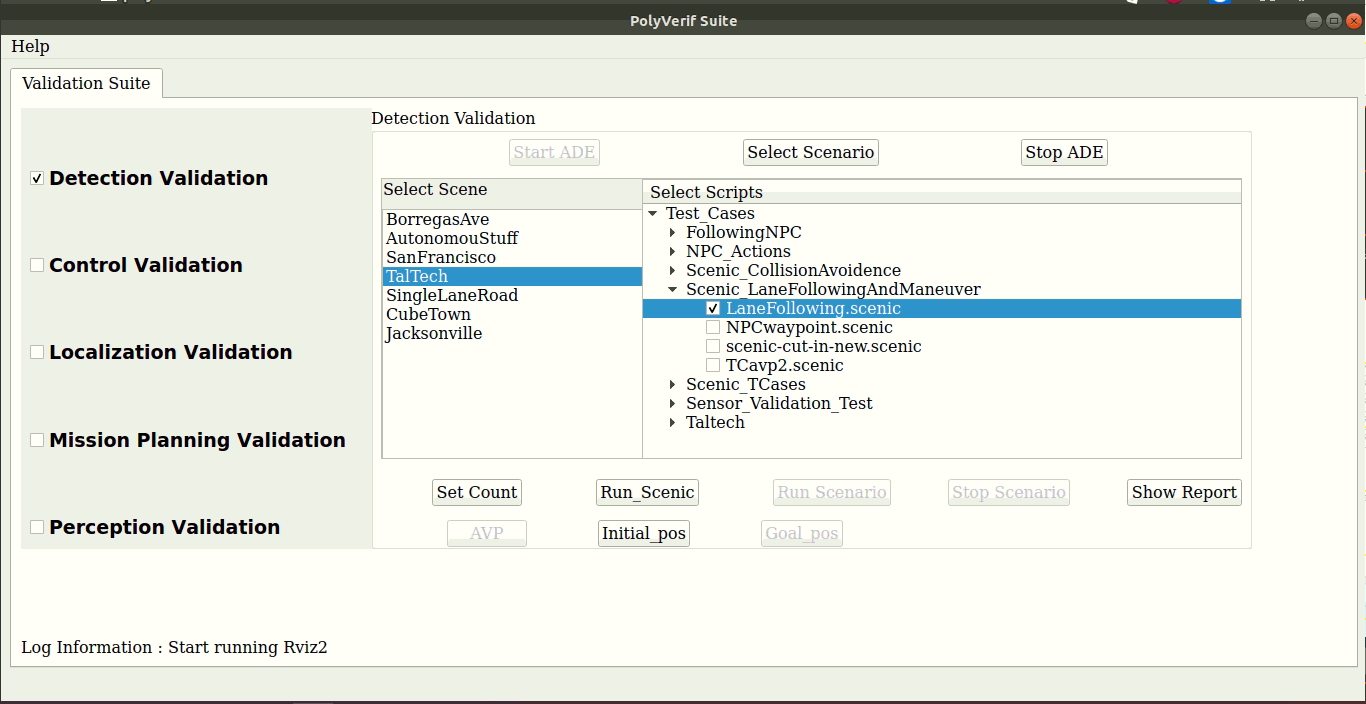
1. Once you select any script form the list it will enables the respective button to run the script as there are two way to run the script -
   * Using Scenic
   * Using PythonAPI

I have selected the scenic script to run so the **Run Scenic** button is enabled.

Also, you can set the number of random iterations for simulation through scenic using the **Set Count** button.

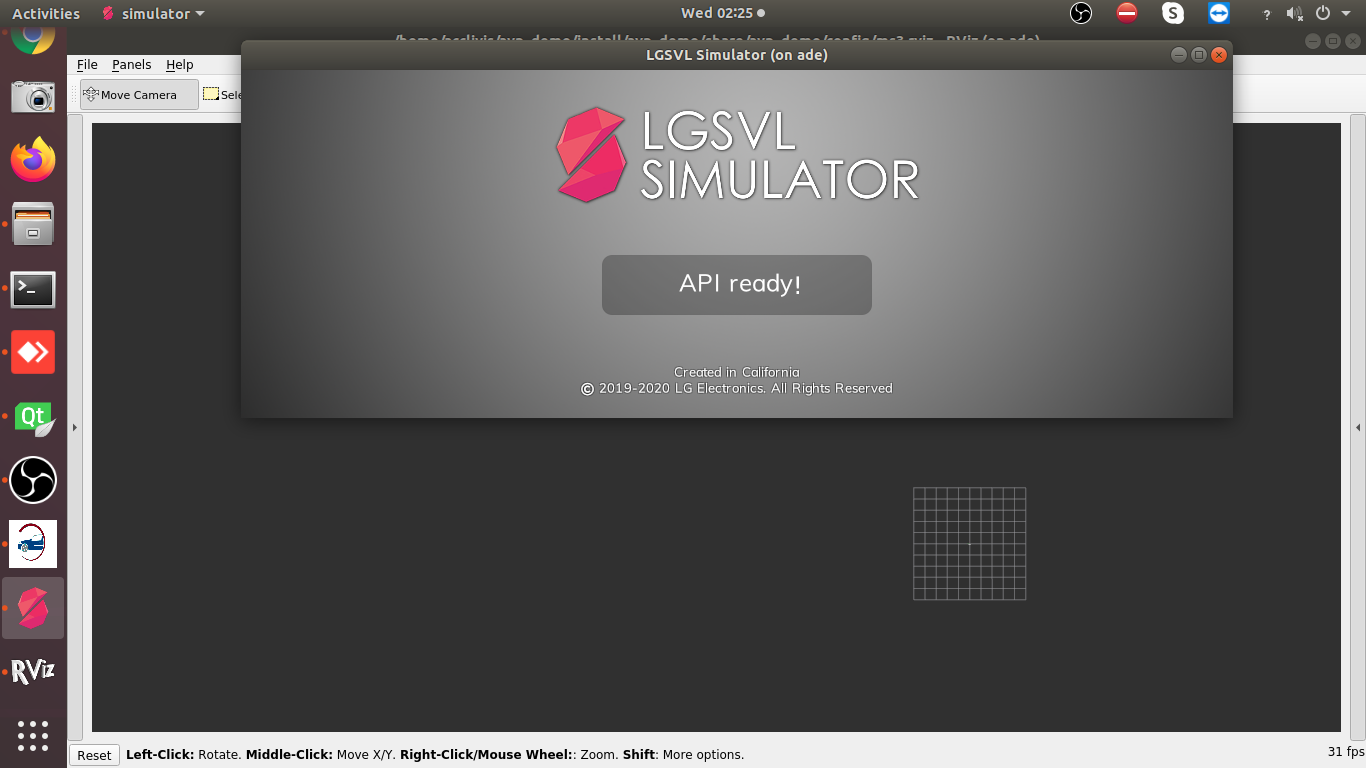
1. Select on **LaneFollowing.scenic** from the Scenic\_LaneFollowingAndManeuver category.

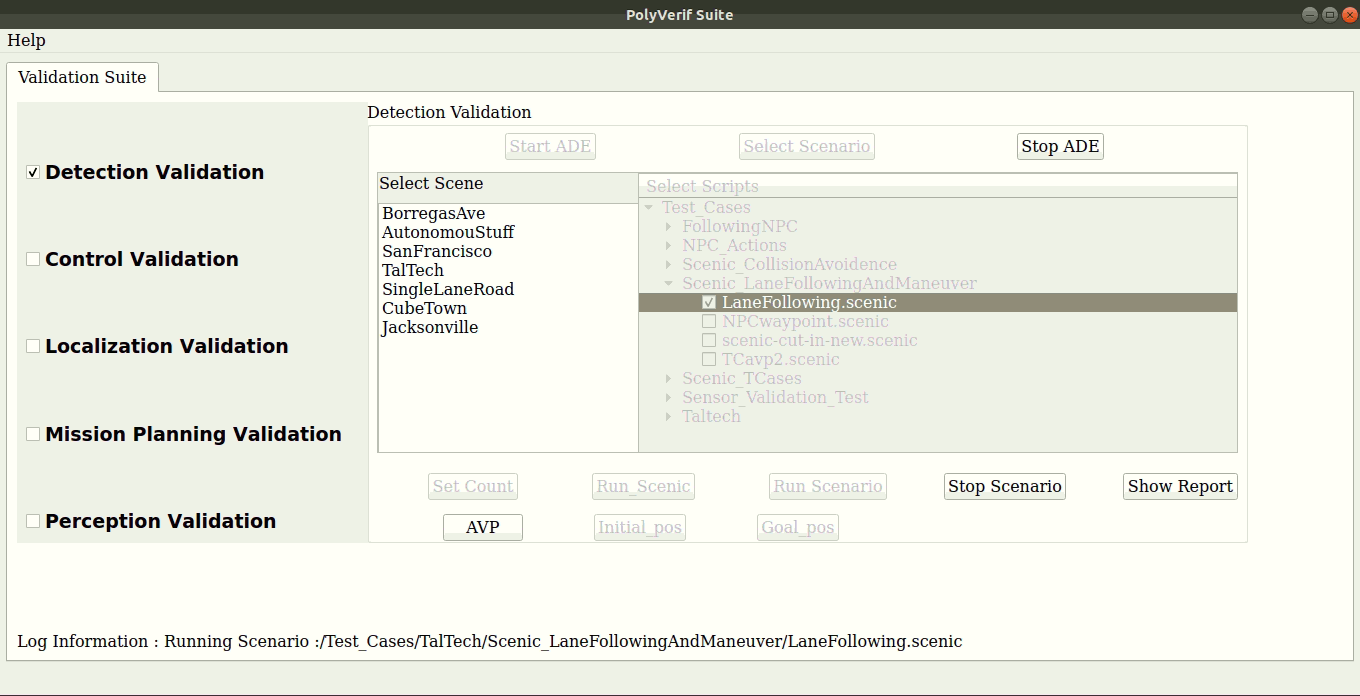
Now click on the AVP button which will start the autoware\_auto\_avp demo. I have modified that demo to run with TalTech map. Below is link of the demo of AutowareAuto – <https://autowarefoundation.gitlab.io/autoware.auto/AutowareAuto/avpdemo.html>



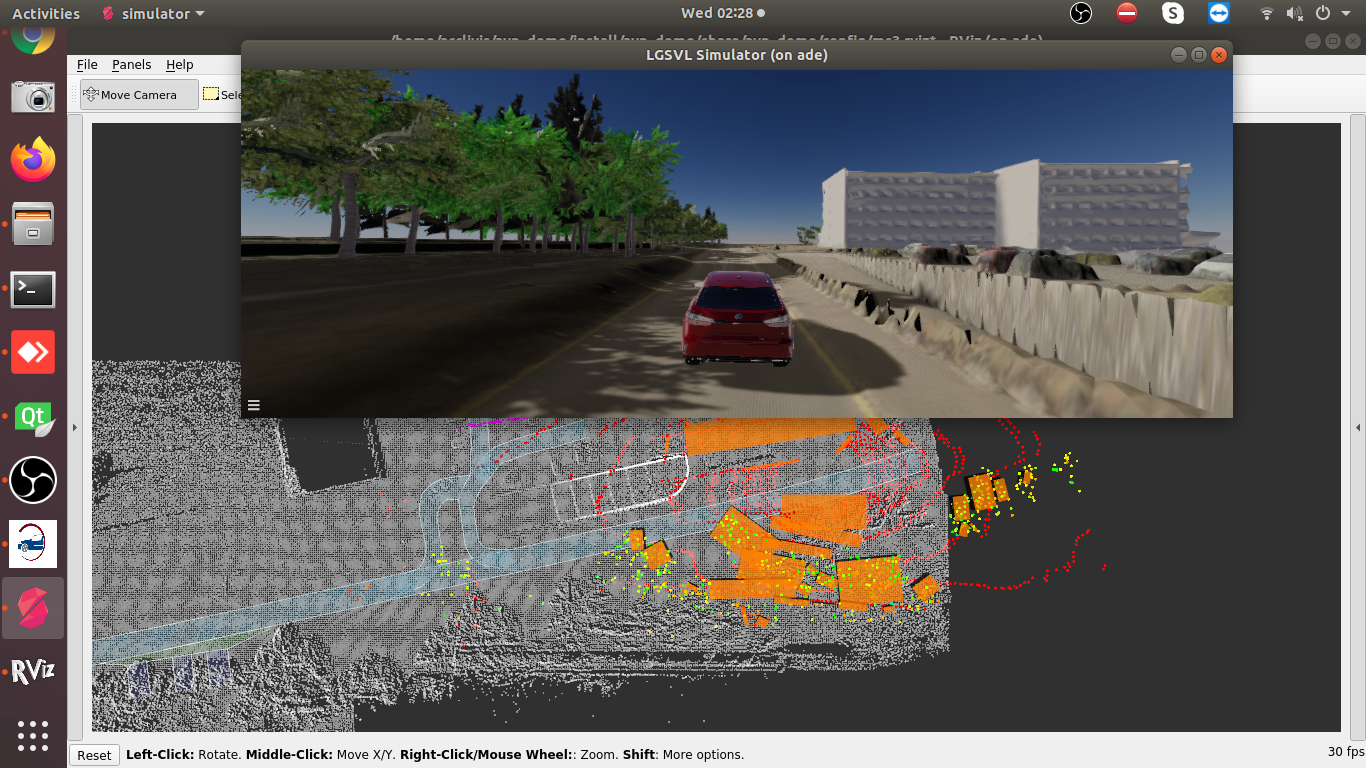
1. Once the rviz is started then click on the Run Scenic button followed by the Initial pos and Goal pos button, it will start the simulation in lgsvl simulator which will controlled by the AutowareAuto decisions.

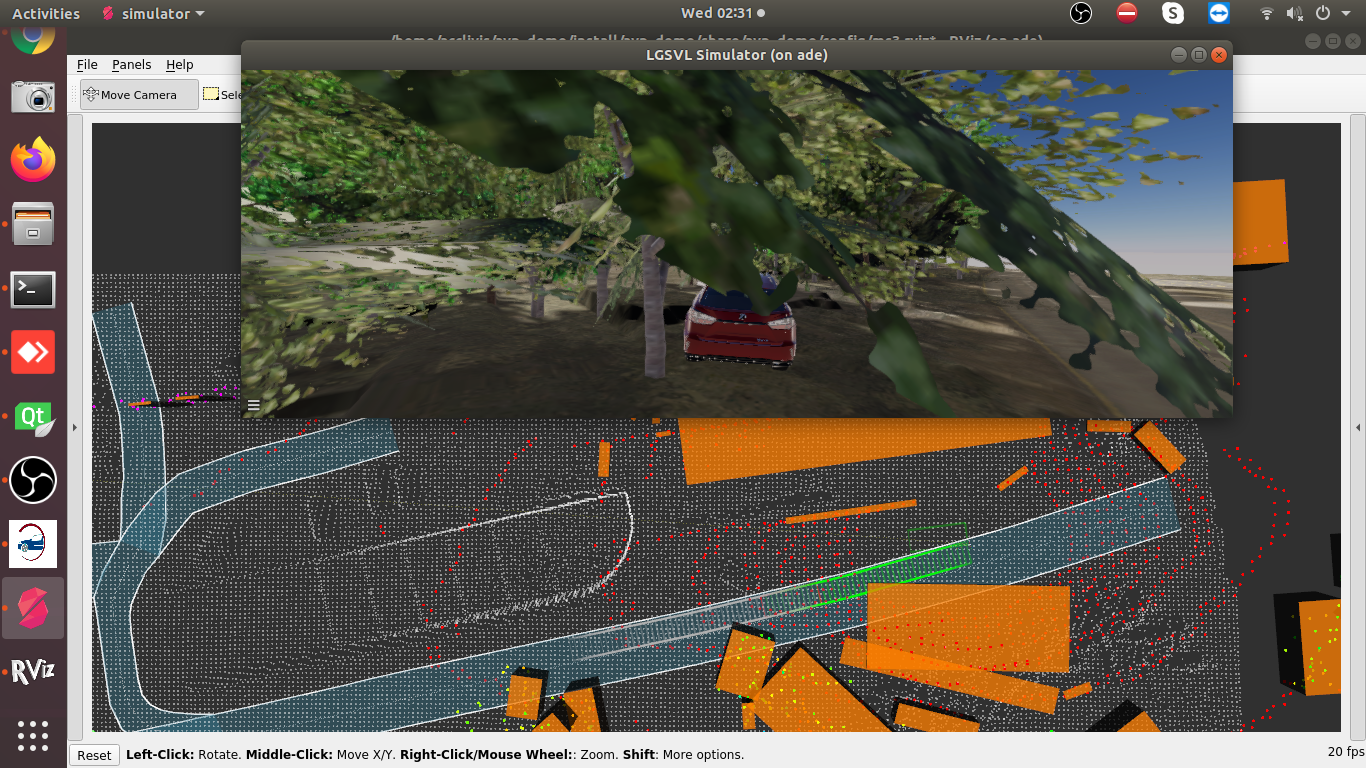
**Note: - Initial pos and goal pos value is a static value which is set specifically for the TalTech map. If you try these values in different map then it will not be accurate.**



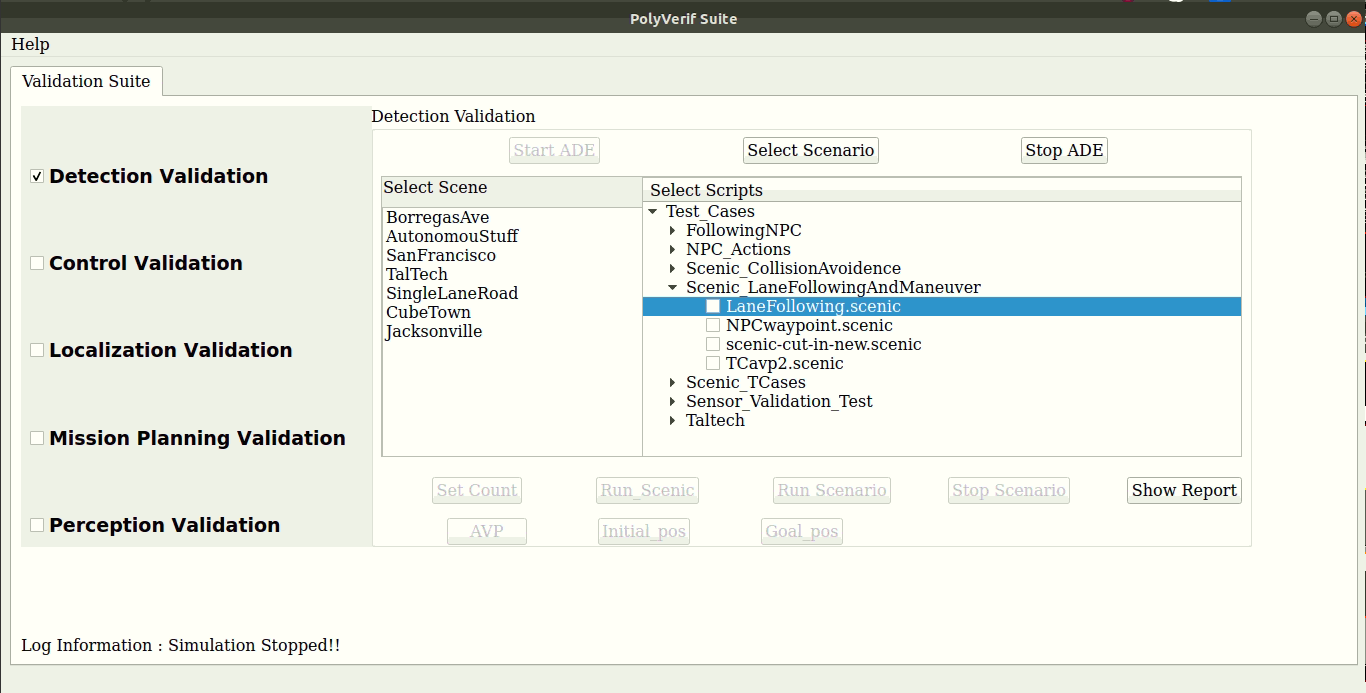


1. Below are the screen shot of trajectory following from the AutowareAuto.



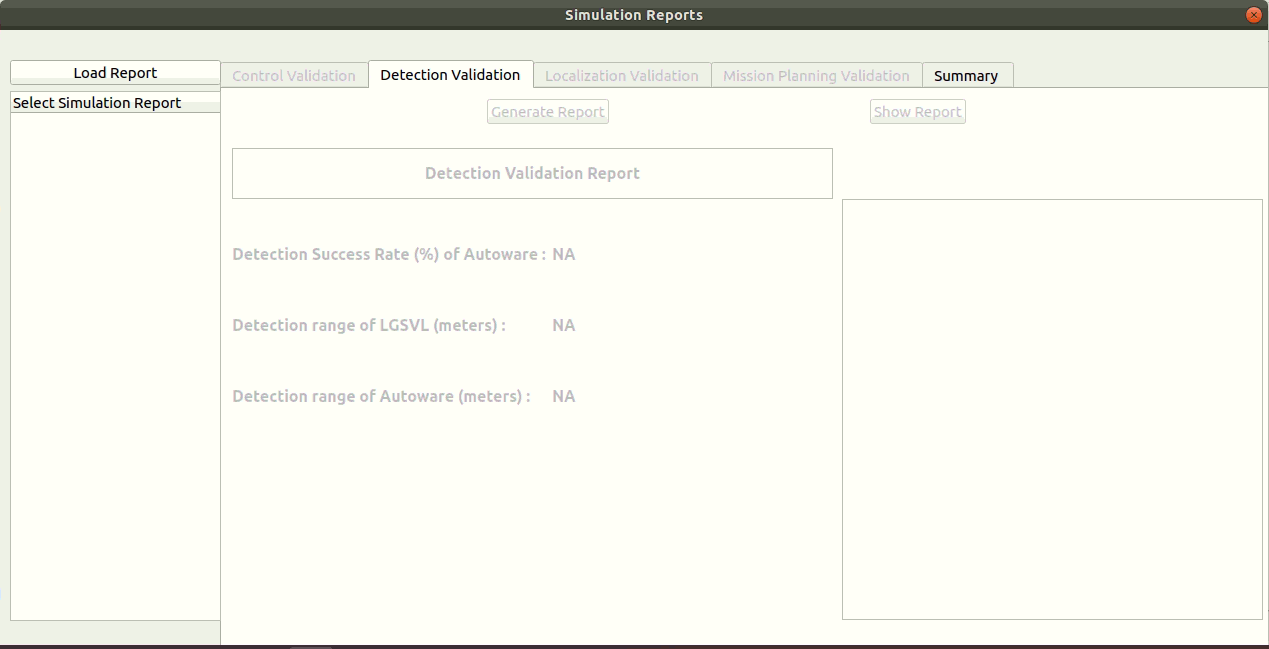


1. Once the simulation is completed or if you want to stop the Click on the Stop Scenario button and click on the Show Report button. It will redirect to another form where you can see the respective simulation reports.

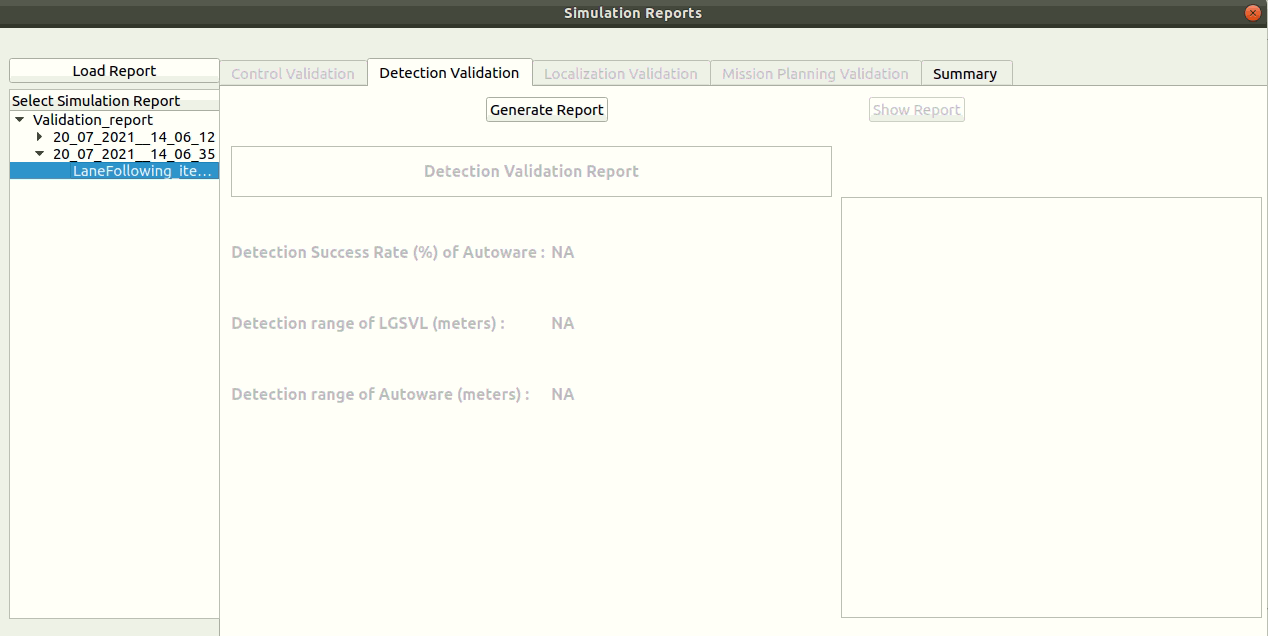


1. This form contains reports information of the respective simulation. As I have run for Perception Validation so it will generate and show the perception report.

Click on the Load Report button, it will list all the simulation with date and time.

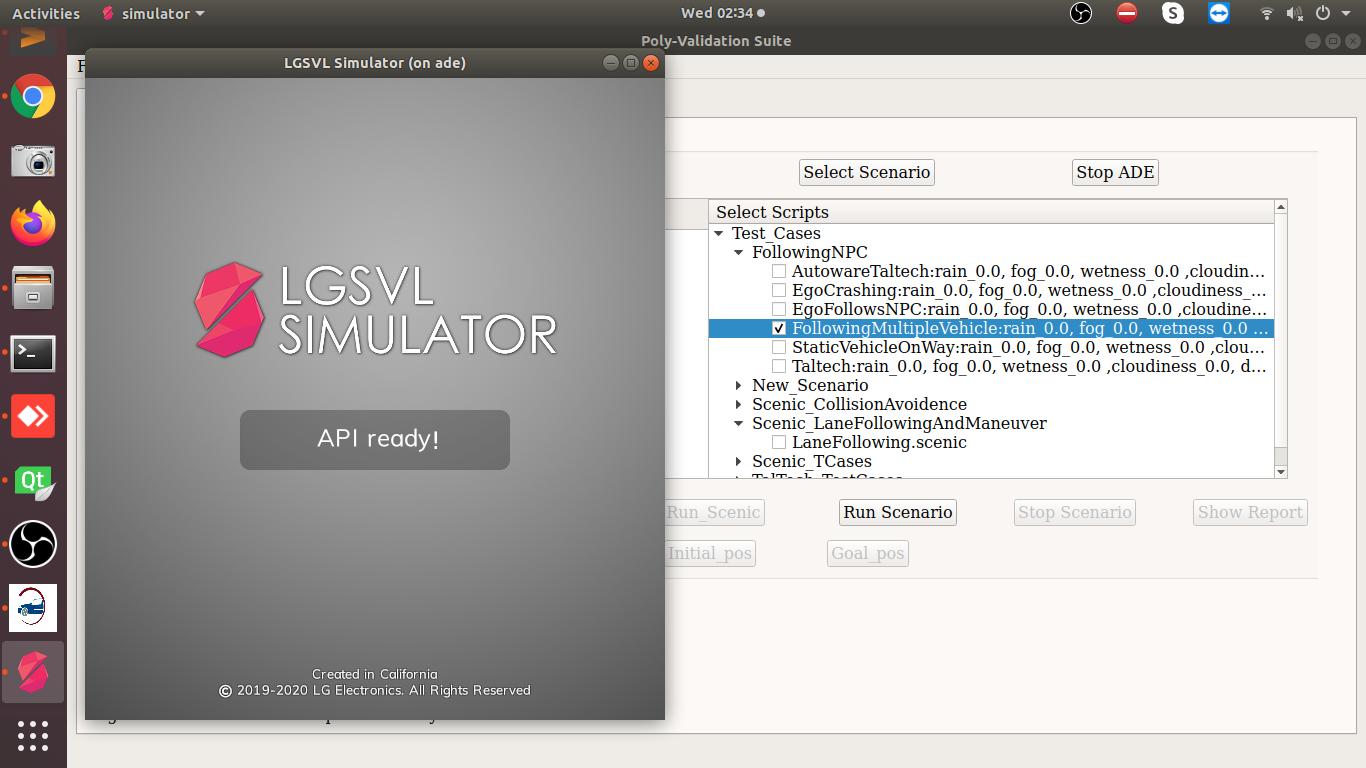


1. The last completed simulation report set at the end of the list. Select and click on the **Generate Report** button. It will take some time to generate the report. Once report is generated the It will enable the show report button, Click on **Show Report** button. You will see few respective parameters on the UI.

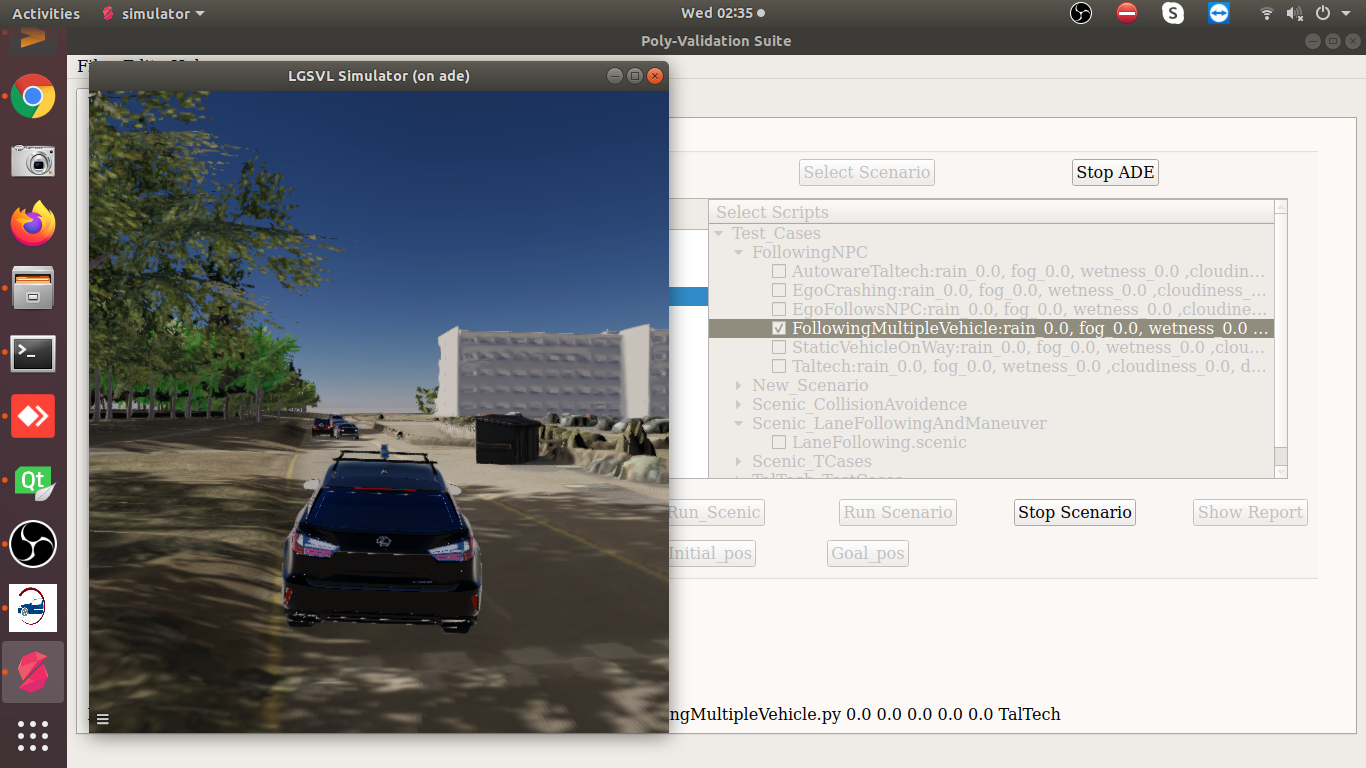


**Python Scenario running –**

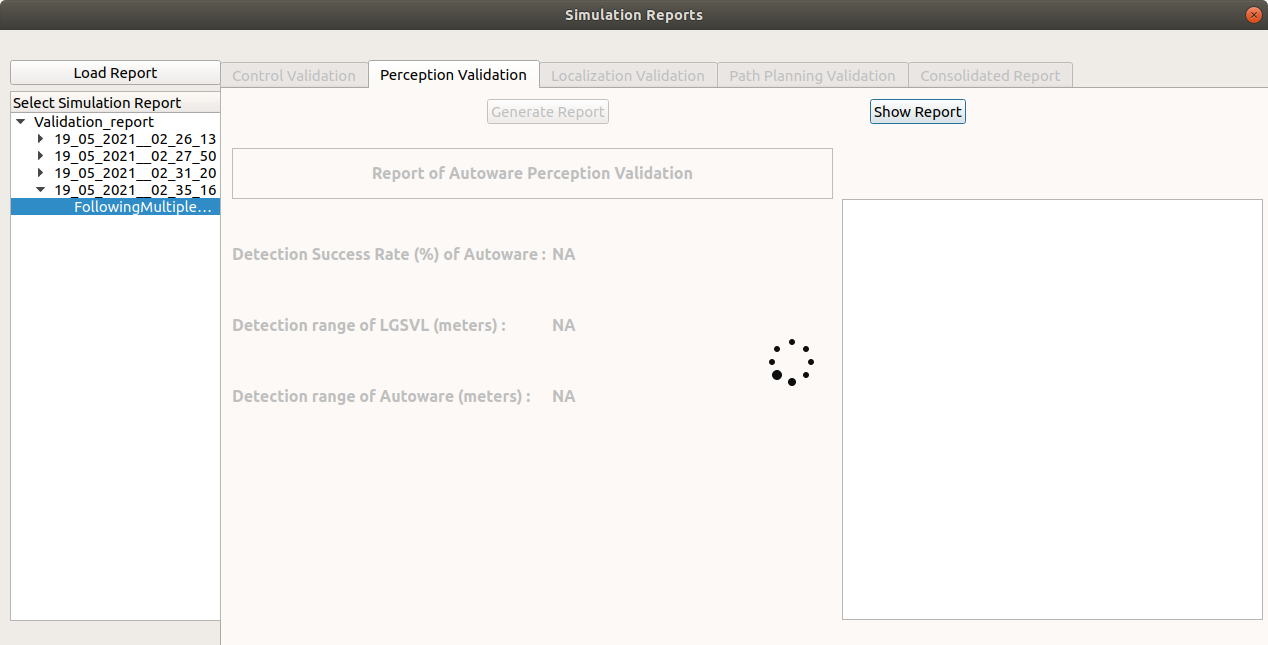
1. Select **Taltech map** and any python scenario form the list.



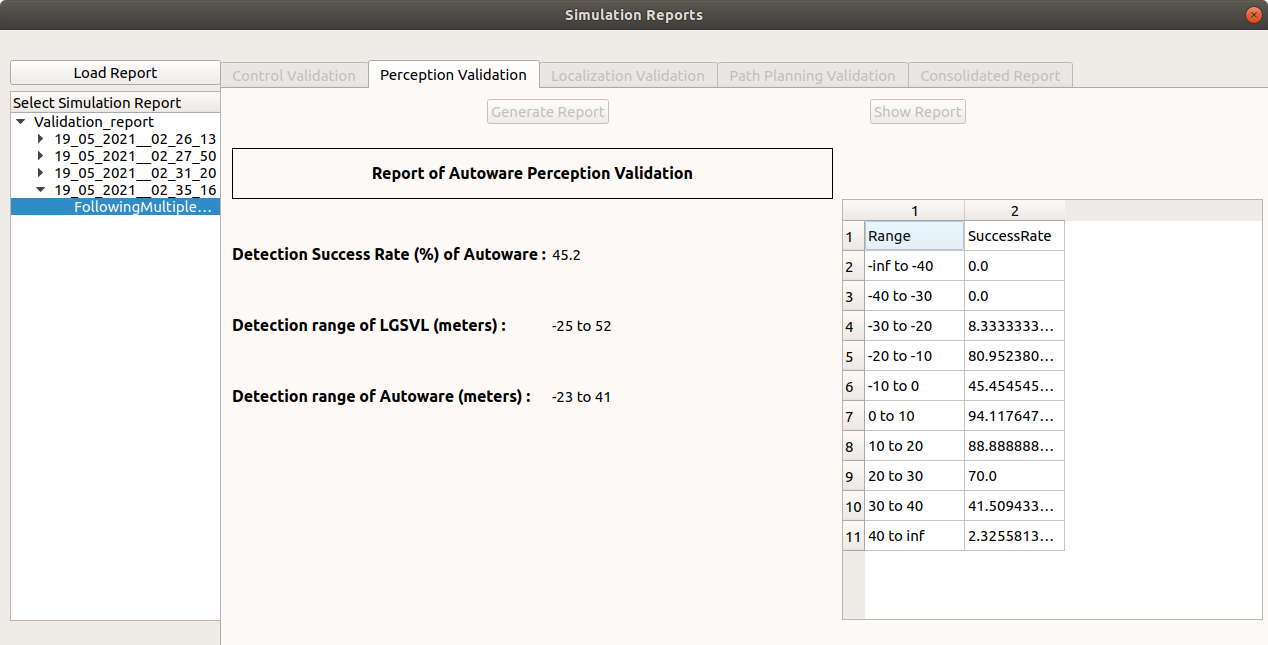
1. Now click on the Run Scenario button to run the selected scenario.



1. Once the simulation is completed, the click on the **Stop scenario** button and click on the Show Report.
2. Now click on the **Load Report** and select the respective scenario in the list.
3. Now click on the **Generate Report** button, it will take some time to process.



1. Once generated data processed, click on the **Show Report** button to see the final results.



**Note: - You can run Control Test suite using the same process. Need to stop the ADE by clicking the Stop ADE button and Select the Control Test Suite.**

**Note: - Some time rviz crashed but the perception stack is running in the back ground.**

**Assumptions/Issues:**

* **Run and tested the test cases for the Taltech map only.**
* **If there is only ego vehicle in the simulation then data will not compute.**
* **Sometime rviz crashed but the perception stack is running in the back ground**
* **While scenario run using the scenic, it got hanged while connecting to ros2 bridge. So need to forcefully terminate using Ctrl+C and start again.**

**References:**

* **Lgsvl Simulator -** [**https://github.com/lgsvl/simulator/releases/tag/2020.06**](https://github.com/lgsvl/simulator/releases/tag/2020.06)
* **Scenic -** [**https://scenic-lang.readthedocs.io/en/latest/syntax\_guide.html?highlight=facing#specifiers**](https://scenic-lang.readthedocs.io/en/latest/syntax_guide.html?highlight=facing#specifiers)
* **PythonAPI -** [**https://www.svlsimulator.com/docs/python-api/python-api/**](https://www.svlsimulator.com/docs/python-api/python-api/)
* **AutowareAuto avp demo -** [**https://autowarefoundation.gitlab.io/autoware.auto/AutowareAuto/avpdemo.html**](https://autowarefoundation.gitlab.io/autoware.auto/AutowareAuto/avpdemo.html)