**Sensor Noise introduce (IMU, gps, LIDar) and Validation**

This document contains the information of Setup and Installation process for running the modified LG Simulator binary in Linux/Unix and using the Application Programming Interface (API) in python for IMU, GPS, LiDAR sensors.

With the completion of these steps the user shall be able to use the API in Python user scripts to introduce noise in senor data frame by frame on ROS2 bridge.

**Note:** Since we are using LG Simulator it will require a high-end server with Graphics card to run it smoothly. The Simulator performance will be very slow if run on a lower machine/server configuration.

Below is the machine configuration for the smooth run:

**PC Configuration:**

* CPU: Intel i7 10700
* GPU: Nvidia RTX 2070 (8GB memory)
* OS: Ubuntu 64-bit

**Prerequisite:**

* Ubuntu
* Nodejs 12.16
* Python3.5 or later
* m\_ lgsvlsimulator-linux64-2020.06 binary (Modified package- available on polyVerif repo)
* PythonAPI’s (Modified package- available on polyVerif repo)
* Test Scripts

**Setup:**

There are two version of lgsvl simulator binary on the polyVerif repo.

**lgsvlsimulator-linux64-2020.06/simulator:** This default binary of simulator taken from the lgsvl repo.

**m\_lgsvlsimulator-linux64-2020.06/simulator:** This is the modified binary of the lgsvl simulator for adding the noise in sensors.

Below changes required to run the modified lgsvl binary in polyVerif framework-

* Please open and modified the file adehome/Poly\_Suite/support\_Files/lgsvl.sh.
* At line number three please change string to **“. /lg\_sim\_06/m\_lgsvlsimulator-linux64-2020.06/simulator”** from **“. /lg\_sim\_06/lgsvlsimulator-linux64-2020.06/simulator”.**
* Now run the polyVerif framework (To use the polyVerif framework please follow this document **PolyVerification\_Suite\_UserGuide**).
* We have added the functionality introduce noise to the sensors like IMU, GPS, LiDAR point cloud data of LG Simulator. We have provided an API interface to add noise to sensor in Python.
* Below is the control flow in the code:

***NoisePythonAPI 🡪 ApplyNoiseToData 🡪 SetTotheLGSimulator 🡪 PublishToROS2Bridge 🡪 AD\_Stack***

**Test Cases:**

We have added few test cases for adding noise in the sensor data of IMU, GPS, LiDAR. You can find the test cases in **adehome/Test\_Cases** directory of respective map/scene, Below are the test case you will find and test in **Sensor\_Validation\_Test** folder -

* **Test\_IMU.py**
* **Test\_GPS.py**
* **Test\_LiDAR.py**

**Python Script for end user:**

Any user can write their own test cases to add noise in sensor data, below are the API’s user can use to add noise-

**APIs for noise addition to GPS –**

This APIS allow user to add noise to GPS sensor using the two variables which is Longitude and Latitude. Below are the ways/possibilities to apply noise to the GPS sensor data.

**a=sim.add\_agent(env.str("LGSVL\_\_VEHICLE\_0", "AVPCar"), lgsvl.AgentType.EGO, state)**

**s=lgsvl.GPSContext() # Var for GPS noise API**

**displacement = .50 # in meters**

**disp\_mtr2mm = displacement \* 1000 # 1 meter = 1000mm**

|  |
| --- |
|  |

**#Possibility-1**

**lat\_disp = disp\_mtr2mm + (0.00000001/1.1132) # 0.00000001 = 1.1132mm**

**long\_disp = 0.0**

**#Possibility-2**

**# lat\_disp = 0.0**

**# long\_disp = disp\_mtr2mm \* (0.00000001/1.1132) # 0.00000001 = 1.1132mm**

**#Possibility-3**

**# lat\_disp = disp\_mtr2mm \* (0.00000001/1.1132) #0.00000001 = 1.1132mm**

**# long\_disp = disp\_mtr2mm \* (0.00000001/1.1132) #0.00000001 = 1.1132mm**

**s.Latitude = lat\_disp # Set Noise to GPS Latitude var**

**s.Longitude = long\_disp # Set Noise to GPS Longitude var**

**a.apply\_gps\_context(s) # Apply the Noise to the sensor**

User can add the noise as displacement in latitude and longitude to the GPS as per their requirement.

**APIs for noise addition to IMU –**

This APIS allow user to add noise to IMU sensor in term of %. It will calculate the value using the % and apply to the IMU sensor data.

**a=sim.add\_agent(env.str("LGSVL\_\_VEHICLE\_0", "AVPCar"), lgsvl.AgentType.EGO, state)**

**s=lgsvl.IMUContext()**

**s.Percentage = 5 # setting % to introduce noise to IMU**

**a.apply\_imu\_context(s) # apply the % noise to the sensor**

**APIs for noise addition to LiDAR –**

This APIS allow user to add noise to LiDAR sensor in term of %. It will calculate the value using the % and apply to the LiDAR sensor point cloud data which will publish to the ROS2 bridge.

**a=sim.add\_agent(env.str("LGSVL\_\_VEHICLE\_0", "AVPCar"), lgsvl.AgentType.EGO, state)**

**s = lgsvl.LidarContext()**

**s.NoisePercentage = 30 # added the error % to the lidar points**

**a.apply\_lidar\_context(s) # apply noise to the sensor**